GW 信奥 CSP 初赛习题集 1-1

计算机基础知识——计算机文化

1. 计算机基本概念

1.单选题 [NOIP 提高组 2005/NOIP 普及组 2005] 下列设备中没有计算功能的是( )。

A. 笔记本电脑

B. 掌上电脑

C. 智能手机

D. 电子计算器

E. 液晶显示器

2.单选题 [NOIP 普及组 2007] IT 的含义是（ ）。

A. 通信技术

B. 信息技术

C.网络技术

D.信息学

3.单选题 [NOIP 普及组 2017] 计算机应用的最早领域是( )。

A. 数值计算

B. 人工智能

C. 机器人

D. 过程控制

2. 计算机历史

4.判断题 [GESP202306【1 级】] 诞生于 1958 年的 103 机是中国第一台通用数字电子计算机，比 1946 年在美国诞生的第一台通用电子计算机 ENIAC 晚了十多年。

5.单选题 [NOIP 普及组 2012/NOIP 提高组 2012/GESP202406【1 级】] 1946 年诞生于美国宾夕法尼亚大学的 ENIAC 属于（ ）计算机。

A. 电子管

B. 晶体管

C. 集成电路

D. 超大规模集成电路

6.单选题 [GESP202309【2 级】] 我国第一台大型通用电子计算机使用的逻辑部件是（ ）。

A. 集成电路

B. 大规模集成电路

C. 晶体管

D. 电子管

7.单选题 [NOIP 提高组 2012/NOIP 普及组 2012] 目前计算机芯片（集成电路）制造的主要原料是（ ），它是一种可以在沙子中提炼出的物质。

A. 硅

B. 铜

C. 锗

D. 铝

8.单选题 [NOIP 普及组 2011] 摩尔定律（Moore's law）是由英特尔创始人之一戈登·摩尔（Gordon Moore）提出来的。根据摩尔定律，在过去几十年以及在可预测的未来几年，单块集成电路的集成度大约每（ ）个月翻一番。

A. 1

B. 6

C. 18

D. 36

3. 计算机人物

9.单选题 [NOIP 普及组 2009/NOIP 提高组 2009] 关于图灵机下面的说法哪个是正确的：

A. 图灵机是世界上最早的电子计算机。

B. 由于大量使用磁带操作，图灵机运行速度很慢。

C. 图灵机是英国人图灵发明的，在二战中为破译德军的密码发挥了重要作用。

D. 图灵机只是一个理论上的计算模型。

10.单选题 [NOIP 普及组 2004] 美籍匈牙利数学家冯·诺依曼对计算机科学发展所做出的贡献是（ ）。

A. 提出理想计算机的数学模型，成为计算机科学的理论基础。

B. 是世界上第一个编写计算机程序的人。

C. 提出存储程序工作原理，并设计出第一台具有存储程序功能的计算机 EDVAC。

D. 采用集成电路作为计算机的主要功能部件。

E. 指出计算机性能将以每两年翻一番的速度向前发展。

11.不定项选择题 [NOIP 提高组 2004] 美籍匈牙利数学家冯·诺依曼对计算机科学发展所做出的贡献包括（ ）。

A. 提出理想计算机的数学模型，成为计算机科学的理论基础。

B. 提出存储程序工作原理，对现代电子计算机的发展产生深远影响。

C. 设计出第一台具有存储程序功能的计算机 EDVAC。

D. 采用集成电路作为计算机的主要功能部件。

E. 指出计算机性能将以每两年翻一番的速度向前发展

12.单选题 [NOIP 提高组 2010/NOIP 普及组 2010] 提出 “存储程序 ”的计算机工作原理的是（ ）。

A. 克劳德 ・香农

B. 戈登 ・摩尔

C. 查尔斯 ・巴比奇

D. 冯・诺依曼"

13.单选题 [NOIP 提高组 2013/CSP-S2020] 1948 年，（ ）将热力学中的熵引入信息通信领域，标志着信息论研究的开端。

A. 冯.诺伊曼 (John von Neumann)

B. 图灵（ Alan Turing ）

C. 欧拉（ Leonhard Euler ）

D. 克劳德.香农（ Claude Shannon ）

4. 计算机奖项

14.单选题 [GESP 样题【1 级】] 计算机领域的图灵奖为了纪念（ ）科学家图灵。

A. 英国

B. 德国

C. 瑞典

D. 法国

15.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 在下面各世界顶级的奖项中，为计算机科学与技术领域做出杰出贡献的科学家设立的奖项是（ ）。

A. 沃尔夫奖

B. 诺贝尔奖

C. 菲尔兹奖

D. 图灵奖

16.单选题 [NOIP 普及组 2014/CSP-J2019/CSP-J2021] 计算机界的最高奖是( )。

A. 菲尔兹奖

B. 诺贝尔奖

C. 图灵奖

D. 普利策奖

17.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 以下和计算机领域密切相关的奖项是（ ）。

A. 奥斯卡奖

B. 图灵奖

C. 诺贝尔奖

D. 普利策奖

18.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 在下列关于图灵奖的说法中，不正确的是（ ）。

A. 图灵奖是美国计算机协会于 1966 年设立的，专门奖励那些对计算机事业作出重要贡献的个人

B. 图灵奖有“计算机界诺贝尔奖”之称

C. 迄今为止，还没有华裔计算机科学家获此殊荣

D. 图灵奖的名称取自计算机科学的先驱、英国科学家阿兰·图灵

19.不定项选择题 [NOIP 提高组 2018] 下列关于图灵奖的说法中，正确的有（ ）。

A. 图灵奖是由电气和电子工程师协会（IEEE）设立的。

B. 目前获得该奖项的华人学者只有姚期智教授一人。

C. 其名称取自计算机科学的先驱、英国科学家艾伦·麦席森·图灵。

D. 它是计算机界最负盛名、最崇高的一个奖项，有“计算机界的诺贝尔奖”之称。

20.单选题 [NOIP 普及组 2011/NOIP 提高组 2011] 1956 年( )授予肖克利 (William Shockley) 、巴丁 (John Bardeen) 和 布拉顿 (Walter Brattain) ，以表彰他们对半导体的研究和晶体管效应的发现

A. 诺贝尔物理学奖

B. 约翰·冯·诺伊曼奖

C. 图灵奖

D. 高德纳奖 (Donald E. Knuth Prize)

5. NOI 系列比赛

21.单选题 [NOIP 普及组 2010] 全国青少年信息学奥林匹克系列活动的主办单位是（ ）。

A. 教育部

B. 科技部

C. 共青团中央

D. 中国计算机学会

22.单选题 [NOIP 普及组 2009/NOIP 提高组 2009] 全国信息学奥林匹克的官方网站为参与信息学竞赛的老师同学们提供相关的信息和资源，请问全国信息学奥林匹克官方网站的网址是：

A. http://www.noi.com/

B. http://www.noi.org/

C. http://www.noi.cn/

D. http://www.xinxixue.com/

23.单选题 [NOIP 普及组 2005] 下列活动中不属于信息学奥赛的系列活动的是（ ）。

A. NOIP

B. NOI

C. IOI

D. 冬令营

E. 程序员等级考试

24.不定项选择题 [NOIP 提高组 2005] 下列活动中属于信息学奥赛系列活动的是( )。

A. NOIP

B. NOI

C. IOI

D. 冬令营

E. 国家队选拔赛

25.单选题 [NOIP 普及组 2017] NOI 的中文意思是（ ）。

A. 中国信息学联赛

B. 全国青少年信息学奥林匹克竞赛

C. 中国青少年信息学奥林匹克竞赛

D. 中国计算机协会

26.单选题 [NOIP 提高组 2010] 以下竞赛活动中历史最悠久的是（ ）

A. 全国青少年信息学奥林匹克联赛（ NOIP ）

B. 全国青少年信息学奥林匹克竞赛（ NOI ）

C. 国际信息学奥林匹克竞赛（ IOI）

D. 亚太地区信息学奥林匹克竞赛（ APIO ）

27.单选题 [NOIP 普及组 2018/NOIP 提高组 2018] 中国计算机学会于（ ）年创办全国青少年计算机程序设计竞赛。

A. 1983

B. 1984

C. 1985

D. 1986

28.单选题 [NOIP 普及组 2013] CCF NOIP 复赛全国统一评测时使用的系统软件是（ ）。

A. NOI Windows

B. NOI Linux

C. NOI Mac OS

D. NOI DOS

29.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 从()年开始，NOIP 竞赛将不再支持 Pascal 语言。

A. 2020

B. 2021

C. 2022

D. 2023

30.单选题 [NOIP 普及组 2015/NOIP 提高组 2015] 在 NOI 系列赛事中参赛选手必须使用由承办单位统一提供的设备。 下列物品中不允许选手自带的是 （ ）。

A. 鼠标

B. 笔

C. 身份证

D. 准考证

31.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 参加 NOI 比赛，以下不能带入考场的是（）。

A. 钢笔

B. 适量的衣服

C. U 盘

D. 铅笔

32.不定项选择题 [NOIP 提高组 2018] NOIP 初赛中，选手可以带入考场的有（ ）。

A. 笔

B. 橡皮

C. 手机（关机）

D. 草稿纸

33.单选题 [NOIP 普及组 2009/NOIP 提高组 2009] 在参加 NOI 系列竞赛过程中，下面哪一种行为是不被严格禁止的：

A. 携带书写工具，手表和不具有通讯功能的电子词典进入赛场。

B. 在联机测试中通过手工计算出可能的答案并在程序里直接输出答案来获取分数。

C. 通过互联网搜索取得解题思路。

D. 在提交的程序中启动多个进程以提高程序的执行效率。

34.不定项选择题 [NOIP 提高组 2014] 在 NOI 比赛中，对于程序设计题，选手提交的答案不得包含下列哪些内容（ ）。

A.试图访问网络

B.打开或创建题目规定的输入 /输出文件之外的其他文件

C.运行其他程序

D.改变文件系统的访问权限

E.读写文件系统的管理信息

35.不定项选择题 [NOIP 提高组 2013] CCF NOIP 复赛考试结束后，因（ ）提出的申诉将不会被受理。

A. 源程序文件名大小写错误

B. 源程序保存在指定文件夹以外的位置

C. 输出文件的文件名错误

D. 只提交了可执行文件，未提交源程序

答案

1-5 E B A T A

6-10 D A C D C

11-15 BC D D A D

16-20 C B C BCD A

21-25 D C E ABCDE B

26-30 B B B C A

31-35 C AB B ABCDE ABCD

GW 信奥 CSP 初赛习题集 1-2

计算机基础知识——计算机硬件系统

1. 计算机体系结构概述

1.单选题 [GESP202312【1 级】/GESP202312【2 级】/GESP202312【3 级】/GESP202312【4 级】] 现代计算机是指电子计算机，它所基于的是（ ）体系结构。

A. 艾伦・图灵

B. 冯・诺依曼

C. 阿塔纳索夫

D. 埃克特-莫克利

2.单选题 [NOIP 普及组 2015/NOIP 提高组 2015/CSP-J2021] 在计算机内部用来传送、存贮、加工处理的数据或指令都是以（ ）形式进行的。

A. 二进制码

B. 八进制码

C. 十进制码

D. 智能拼音码

3.单选题 [NOIP 普及组 2011] 从 ENIAC 到当前最先进的计算机，冯・诺依曼体系结构始终占有重要的地位。冯・诺依曼体系结构的核心内容是（ ）。

A. 采用开关电路

B. 采用半导体器件

C. 采用存储程序和程序控制原理

D. 采用键盘输入

4.判断题 [GESP202306【1 级】/GESP 样题【5 级】] 计算机硬件主要包括运算器、控制器、存储器、输入设备和输出设备。

5.单选题 [NOIP 普及组 2007] 一个完整的计算机系统应包括（ ）。

A.系统硬件和系统软件

B. 硬件系统和软件系统

C. 主机和外部设备

D. 主机、键盘、显示器和辅助存储器

6.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 下面哪个部件对于个人桌面电脑的正常运行不是必需的（ ）。

A. CPU

B. 图形卡（显卡）

C. 光驱

D. 主板

E. 内存

7.单选题 [NOIP 普及组 2012] 计算机如果缺少（ ），将无法正常启动。

A. 内存

B. 鼠标

C. U 盘

D. 摄像头

2. 输入输出系统

8.单选题 [GESP202303【1 级】] 以下不属于计算机输入设备的有（ ）。

A．键盘

B．音箱

C．鼠标

D．传感器

9.单选题 [GESP202309【5 级】/GESP202309【6 级】] 近年来，线上授课变得普遍，很多有助于改善教学效果的设备也逐渐流行，其中包括比较常用的手写板，那么它属于哪类设备？（ ）。

A. 输入

B. 输出

C. 控制

D. 记录

10.单选题 [GESP202306【1 级】/GESP 样题【5 级】/GESP 样题【6 级】] 以下不属于计算机输出设备的有（ ）。

A. 麦克风

B. 音箱

C. 打印机

D. 显示器

11.单选题 [NOIP 普及组 2005] 以下哪个不是计算机的输出设备（ ）。

A. 音箱

B. 显示器

C. 打印机

D. 扫描仪

E. 绘图仪

12.不定项选择题 [NOIP 提高组 2005] 以下哪个（些）不是计算机的输出设备（ ）。

A. 鼠标

B. 显示器

C. 键盘

D. 扫描仪

E. 绘图仪

13.单选题 [NOIP 普及组 2014/NOIP 普及组 2018] 以下哪一种设备属于输出设备( )。

A. 扫描仪

B. 键盘

C. 鼠标

D. 打印机

14.不定项选择题 [NOIP 提高组 2004/NOIP 普及组 2004] 彩色显示器所显示的五彩斑斓的色彩，是由哪三色混合而成的（ ）。

A. 红

B. 白

C. 蓝

D. 绿

E. 橙

15.不定项选择题 [NOIP 提高组 2012] 在计算机显示器所使用的 RGB 颜色模型中，（ ）属于三原色之一。

A. 黄色

B. 蓝色

C. 紫色

D. 绿色

3. 处理器系统

16.单选题 [NOIP 普及组 2006] CPU 是( )的简称。

A. 硬盘

B. 中央处理器

C. 高级程序语言

D. 核心寄存器

17.单选题 [NOIP 提高组 2006] 在以下各项中。（ ）不是 CPU 的组成部分。

A. 控制器

B. 运算器

C. 寄存器

D. ALU

E. RAM

18.单选题 [NOIP 提高组 2007/NOIP 普及组 2007] 在以下各项中,( )不是 CPU 的组成部分。

A. 控制器

B. 运算器

C. 寄存器

D. 主板

E. 算术逻辑单元(ALU)

19.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 微型计算机中，控制器的基本功能是（ ）。

A. 控制机器各个部件协调工作

B. 实现算术运算和逻辑运算

C. 获取外部信息

D. 存放程序和数据

20.单选题 [NOIP 普及组 2012/NOIP 提高组 2012] 目前个人电脑的（ ）市场占有率最靠前的厂商包括 Intel、AMD 等公司。

A. 显示器

B. CPU

C. 内存

D. 鼠标

21.单选题 [NOIP 普及组 2016] 以下不是 CPU 生产厂商的是（）。

A. Intel

B. AMD

C. Microsoft

D. IBM

22.单选题 [NOIP 普及组 2015] 在 PC 机中， PENTIUM (奔腾）、酷睿、赛扬等是指（ ）。

A. 生产厂家名称

B. 硬盘的型号

C. CPU 的型号

D. 显示器的型号

23.单选题 [NOIP 普及组 2004] 下列哪个不是 CPU（中央处理单元）（ ）。

A. Intel Itanium

B. DDR SDRAM

C. AMD Athlon64

D. AMD Opteron

E. IBM Power 5

24.不定项选择题 [NOIP 提高组 2009/NOIP 普及组 2009] 关于 CPU 下面哪些说法是正确的：

A. CPU 全称为中央处理器（或中央处理单元）。

B. CPU 能直接运行机器语言。

C. CPU 最早是由 Intel 公司发明的。

D. 同样主频下，32 位的 CPU 比 16 位的 CPU 运行速度快一倍。

25.单选题 [NOIP 普及组 2005] 处理器 A 每秒处理的指令数是处理器 B 的 2 倍。某一特定程序 P 分别编译为处理器 A 和处理器 B 的指令，编译结果处理器 A 的指令数是处理器 B 的 4 倍。已知程序 P 在处理器 A 上执行需要 1 个小时，那么在输入相同的情况下，程序 P 在处理器 B 上执行需要（ ）小时。

A. 4

B. 2

C. 1

D. 1 / 2

E. 1 / 4

26.不定项选择题 [NOIP 提高组 2005] 处理器 A 每秒处理的指令数是处理器 B 的 2 倍。某一特定程序 P 分别编译为处理器 A 和处理器 B 的指令，编译结果处理器 A 的指令数是处理器 B 的 4 倍。已知程序 P 的算法时间复杂度为 O(n^2)，如果处理器 A 执行程序 P 时能在一小时内完成的输入规模为 n，则处理器 B 执行程序 P 时能在一小时内完成的输入规模为（ ）。

A. 4 * n

B. 2 * n

C. n

D. n / 2

E. n / 4

4. 存储器系统

27.不定项选择题 [NOIP 提高组 2004] 下列哪个（些）不是计算机的存储设备（ ）。

A. 文件管理器

B. 内存

C. 显卡

D. 硬盘

E. U 盘

28.单选题 [NOIP 普及组 2004] 下列哪个不是计算机的存储设备（ ）。

A. 文件管理器

B. 内存

C. 高速缓存

D. 硬盘

E. U 盘

29.单选题 [NOIP 普及组 2016] 以下不是存储设备的是（）。

A. 光盘

B. 磁盘

C. 固态硬盘

D. 鼠标

30.单选题 [GESP202309【1 级】] 我们通常说的“内存”属于计算机中的（ ）。

A. 输出设备

B. 输入设备

C. 存储设备

D. 打印设备

31.单选题 [NOIP 提高组 2011/NOIP 普及组 2011] 寄存器是（ ）的重要组成部分。

A. 硬盘

B. 高速缓存

C. 内存

D. 中央处理器 (CPU)

32.单选题 [NOIP 提高组 2010/NOIP 普及组 2010] 主存储器的存取速度比中央处理器 （CPU ）的工作速度慢很多， 从而使得后者的效率受到 影响。而根据局部性原理，

CPU 所访问的存储单元通常都趋于聚集在一个较小的连续区域 中。于是，为了提高系统整体的执行效率，在 CPU 中引入了（ ）

A. 寄存器

B. 高速缓存

C. 闪存

D. 外存

33.单选题 [NOIP 普及组 2005/NOIP 提高组 2005] 以下断电之后仍能保存数据的是（ ）。

A. 硬盘

B. 寄存器

C. 显存

D. 内存

E. 高速缓存

34.单选题 [NOIP 普及组 2006] 以下断电之后仍能保存数据的有（ ）。

A. 寄存器

B. ROM

C. RAM

D. 高速缓存

35.不定项选择题 [NOIP 提高组 2006] 以下断电之后将不能保存数据的有（ ）。

A. 硬盘

B. ROM

C. 显存

D. RAM

36.单选题 [NOIP 普及组 2007] 以下断电之后仍能保存数据的有（ ）。

A. 硬盘

B. 高速缓存

C. 显存

D. RAM

37.不定项选择题 [NOIP 提高组 2007] 以下断电之后仍能保存数据的有（ ）。

A. 硬盘

B. ROM

C. 显存

D. RAM

38.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 计算机在工作过程中，若突然停电，（ ）中的信息不会丢失。

A. ROM 和 RAM

B. CPU

C. ROM

D. RAM

39.单选题 [NOIP 普及组 2014] 断电后会丢失数据的存储器是（ ）。

A. RAM

B. ROM

C. 硬盘

D. 光盘

40.单选题 [NOIP 提高组 2006] BIOS（基本输入输出系统）是一组固化在计算机内（ ）上一个 ROM 芯片上的程序。

A. 控制器

B. CPU

C. 主板

D. 内存条

E. 硬盘

41.单选题 [NOIP 普及组 2009/NOIP 提高组 2009] 关于 BIOS 下面说法哪个是正确的：

A. BIOS 是计算机基本输入输出系统软件的简称。

B. BIOS 里包含了键盘、鼠标、声卡、显卡、打印机等常用输入输出设备的驱动程序。

C. BIOS 一般由操作系统厂商来开发完成。

D. BIOS 能提供各种文件拷贝、复制、删除以及目录维护等文件管理功能。

42.单选题 [GESP202303【2 级】] 以下存储器中的数据不会受到附近强磁场干扰的是（ ）

A．硬盘

B．U 盘

C．内存

D．光盘

43.单选题 [CSP-J2023] 在计算机中，以下哪个选项描述的数据存储容量最小（）

A. 字节 (byte)

B. 比特 (bit)

C. 字 (word)

D. 千字节 (kilobyte）

44.判断题 [GESP 样题【2 级】] 计算机系统中存储的基本单位用 B 来表示，它代表的是字节。

45.单选题 [NOIP 普及组 2017] 计算机存储数据的基本单位是( )。

A. bit

B. Byte

C. GB

D. KB

46.单选题 [GESP202303【1 级】] 计算机系统中存储的基本单位用 B 来表示，它代表的是（ ）。

A.Byte

B.Block

C.Bulk

D.Bit

47.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 一个字节（ byte ）由（ ）个二进制位组成

A. 8

B. 16

C. 32

D. 以上都有可能

48.单选题 [NOIP 普及组 2018/NOIP 普及组 2015] 1MB 等于（ ）。

A. 1000 字节

B. 1024 字节

C. 1000 X 1000 字节

D. 1024 X 1024 字节

49.单选题 [NOIP 普及组 2014/NOIP 提高组 2014] 1TB 代表的字节数是（）。

A. 2 的 10 次方

B. 2 的 20 次方

C. 2 的 30 次方

D. 2 的 40 次方

50.单选题 [NOIP 普及组 2011] 一片容量为 8GB 的 SD 卡能存储大约( )张大小为 2MB 的数码照片。

A. 1600

B. 2000

C. 4000

D. 16000

51.判断题 [GESP202303【2 级】] 明明和笑笑在“小庙会”上分别抽到一个 4GB 和 4096MB 的 U 盘，容量大的盘是笑笑的。

52.单选题 [NOIP 提高组 2007/NOIP 普及组 2007] 在下列各项中，只有（ ）不是计算机存储容量的常用单位。

A. Byte

B. KB

C.MB

D.UB

E.TB

53.单选题 [NOIP 普及组 2009] 关于计算机内存下面的说法哪个是正确的：

A. 随机存储器（RAM）的意思是当程序运行时，每次具体分配给程序的内存位置是随机而不确定的。

B. 1MB 内存通常是指 1024*1024 字节大小的内存。

C. 计算机内存严格说来包括主存（memory）、高速缓存（cache）和寄存器（register）三个部分。

D. 一般内存中的数据即使在断电的情况下也能保留 2 个小时以上。

54.不定项选择题 [NOIP 提高组 2009] 关于计算机内存下面的说法哪些是正确的：

A. 随机存储器（RAM）的意思是当程序运行时，每次具体分配给程序的内存位置是随机而不确定的。

B. 一般的个人计算机在同一时刻只能存/取一个特定的内存单元。

C. 计算机内存严格说来包括主存（memory)、高速缓存（cache).和寄存器（register).三个部分。

D. 1MB 内存通常是指 1024*1024 字节大小的内存。

55.单选题 [NOIP 普及组 2012] 矢量图（Vector Image）图形文件所占的贮存空间比较小， 并且无论如何放大、 缩小或旋转等都不会失真，是因为它（ ）。

A. 记录了大量像素块的色彩值来表示图像

B. 用点、直线或者多边形等基于数学方程的几何图元来表示图像

C. 每个像素点的颜色信息均用矢量表示

D. 把文件保存在互联网，采用在线浏览的方式查看图像

56.单选题 [NOIP 普及组 2014] 下列选项中不属于图像格式的是（）。

A. JPEG 格式

B. TXT 格式

C. GIF 格式

D. PNG 格式

57.单选题 [CSP-S2019] 下列属于图像文件格式的（ ）

A. WMV

B. MPEG

C. JPEG

D. AVI

58.单选题 [NOIP 普及组 2015] 下列选项中不属于视频文件格式的是（ ）。

A. TXT

B. AVI

C. MOV

D. RMVB

59.不定项选择 [NOIP 提高组 2015] 下列属于视频文件格式的有（　　）。

A. AVI

B. MPEG

C. WMV

D. JPEG

60.单选题 [NOIP 普及组 2005/NOIP 提高组 2005] 一位艺术史学家有 20000 幅真彩色图像，每幅图像约占 3M 空间。如果将这些图像以位图形式保存在 CD 光盘上（一张 CD 光盘的容量按 600M 计算），大约需要（ ）张 CD 光盘。

A. 1

B. 10

C. 100

D. 1000

E. 10000

61.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 分辨率为 800x600、16 位色的位图，存储图像信息所需的空间为( )。

A. 937.5KB

B. 4218.75KB

C. 4320KB

D. 2880KB

62.单选题 [CSP-J2020] 现有一张分辨率为 2048×1024 像素的 32 位真彩色图像。请问要存储这张图像，需要多大的存储空间？（ ）。

A. 16MB

B. 4MB

C. 8MB

D. 2MB

63.单选题 [CSP-S2020] 现有一段 8 分钟的视频文件，它的播放速度是每秒 24 帧图像，每帧图像是一幅分辨率为 2048×1024 像素的 32 位真彩色图像。请问要存储这段原始无压缩视频，需要多大的存储空间？（ ）

A. 30G

B. 90G

C. 150G

D. 450G

5. 总线系统

64.单选题 [NOIP 普及组 2014] CPU、存储器、 I/O 设备是通过（）连接起来的。

A. 接口

B. 总线

C. 控制线

D. 系统文件

65.单选题 [NOIP 普及组 2016] 以下是 32 位机器和 64 位机器的区别的是（）。

A. 显示器不同

B. 硬盘大小不同

C. 寻址空间不同

D. 输入法不同

66.单选题 [NOIP 普及组 2012/NOIP 提高组 2012] 地址总线的位数决定了 CPU 可直接寻址的内存空间大小，例如地址总线 为 16 位，其最大的可寻址空间为 64KB。如果地址总线是 32 位，则理论上最大 可寻址的内存空间为（ ）。

A. 128KB

B. 1MB

C. 1GB

D. 4GB

67.单选题 [NOIP 提高组 2016] 某计算机的 CPU 和内存之间的地址总线宽度是 32 位（bit），这台计算机 最 多可以使用 （ ）的内存。

A. 2GB

B. 4GB

C. 8GB

D. 16GB

68.单选题 [NOIP 普及组 2004] 下列说法中错误的是（ ）。

A. CPU 的基本功能就是执行指令。

B. CPU 访问内存的速度快于访问高速缓存的速度。

C. CPU 的主频是指 CPU 在 1 秒内完成的指令周期数。

D. 在一台计算机内部，一个内存地址编码对应唯一的一个内存单元。

E. 数据总线的宽度决定了一次传递数据量的大小，是影响计算机性能的因素之一。

69.不定项选择题 [NOIP 提高组 2004] 下列说法中正确的有（ ）。

A. CPU 的基本功能就是执行指令。

B. CPU 的主频是指 CPU 在 1 秒内完成的指令周期数，主频越快的 CPU 速度一定越快。

C. 内部构造不同的 CPU 运行相同的机器语言程序，一定会产生不同的结果。

D. 在一台计算机内部，一个内存地址编码对应唯一的一个内存单元。

E. 数据总线的宽度决定了一次传递数据量的大小，是影响计算机性能的因素之一。

70.单选题 [NOIP 普及组 2015] 所谓的 “中断 ”是指（ ）。

A. 操作系统随意停止一个程序的运行

B. 当出现需要时， CPU 暂时停止当前程序的执行转而执行处理新情况的过程

C. 因停机而停止一个程序的运行

D. 电脑死机

答案

1-5 B A C T B

6-10 C A B A A

11-15 D ACD D ACD BD

16-20 B E D A B

21-25 C C B AB B

26-30 B ACDE A D C

31-35 D B A B CD

36-40 A AB C A C

41-45 A D B T B

46-50 A A D D C

51-55 F D B BD B

56-60 B D A ABCD C

61-65 A C B B C

66-70 D B B ADE B

GW 信奥 CSP 初赛习题集 1-3

计算机基础知识——计算机软件系统

1. 操作系统

1.判断题 [GESP 样题【1 级】] 操作系统是让用户可以操纵计算机完成各种功能的软件。

2.单选题 [CSP-S2020/NOIP 普及组 2014] 操作系统的功能是（ ）

A. 负责外设与主机之间的信息交换

B. 控制和管理计算机系统的各种硬件和软件资源的使用

C. 负责诊断机器的故障

D. 将源程序编译成目标程序

3.单选题 [NOIP 普及组 2015] 操作系统的作用是（ ）。

A. 把源程序译成目标程序

B. 便于进行数据管理

C. 控制和管理系统资源

D. 实现硬件之间的连接

4.不定项选择题 [NOIP 提高组 2009] 关于操作系统下面说法哪些是正确的：

A. 多任务操作系统专用于多核心或多个 CPU 架构的计算机系统的管理。

B. 在操作系统的管理下，一个完整的程序在运行过程中可以被部分存放在内存中。

C. 分时系统让多个用户可以共享一台主机的运算能力，为保证每个用户都得到及时的响应通常会采用时间片轮转调度的策略。

D. 为了方便上层应用程序的开发，操作系统都是免费开源的。

5.单选题 [NOIP 普及组 2006] Linux 是一种( )。

A. 绘图软件

B. 程序设计语言

C. 操作系统

D. 网络浏览器

6.单选题 [GESP 样题【1 级】/GESP 样题【2 级】] 人们在使用计算机时所提到的 Windows 通常指的是（ ）。

A. 操作系统

B. 多人游戏

C. 上市公司

D. 家居用具

7.判断题 [GESP202312【1 级】/GESP202312【2 级】/GESP202312【3 级】/GESP202312【4 级】] 小杨最近在准备考 GESP，他用的 Dev C++来练习和运行程序，所以 Dev C++也是一个小型操作系统。

8.单选题 [GESP202403【1 级】/GESP202403【2 级】/GESP202403【3 级】/GESP202403【4 级】] 小杨的父母最近刚刚给他买了一块华为手表，他说手表上跑的是鸿蒙，这个鸿蒙是？（ ）

A. 小程序

B. 计时器

C. 操作系统

D. 神话人物

9.不定项选择题 [NOIP 提高组 2014] 下列 ( )软件属于操作系统软件。

A.Microsoft Word

B.Windows XP

C.Android

D.Mac OS X

E.Oracle

10.不定项选择题 [NOIP 提高组 2015] 以下属于操作系统的有（）

A. Windows XP

B. UNIX

C. Linux

D. Mac OS

11.单选题 [NOIP 普及组 2004] 下列哪个软件属于操作系统软件（ ）。

A. Microsoft Word

B. 金山词霸

C. Foxmail

D. WinRAR

E. Red Hat Linux

12.不定项选择题 [NOIP 提高组 2004] 下列哪个（些）软件属于操作系统软件（ ）。

A. Microsoft Word

B. Windows XP

C. Foxmail

D. 金山影霸

E. Red Hat Linux

13.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 在以下各项中，（ ）不是操作系统软件。

A. Solaris

B. Linux

C. Windows Vista

D. Sybase

14.单选题 [NOIP 普及组 2009] 下列软件中不是计算机操作系统的是：

A. Windows

B. Linux

C. OS/2

D. WPS

15.单选题 [NOIP 普及组 2012] （ ）不属于操作系统。

A. Windows

B. DOS

C. Photoshop

D. NOI Linux

16.单选题 [CSP-J2023] 以下哪个不是操作系统?()

A. Linux

B. Windows

C. Android

D. HTML

2. 数据库系统

17.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] 在关系数据库中，存放在数据库中的数据的逻辑结构以（ ）为主。

A. 二叉树

B. 多叉树

C.哈希表

D.二维表

18.不定项选择题 [NOIP 提高组 2007/NOIP 普及组 2007] 冗余数据是指可以由其他数据导出的数据，例如，数据库中已存放了学生的数学、语文和英语的三科成绩，如果还存放三科成绩的总分，则总分就可以看作冗余数据。冗余数据往往会造成数据的不一致，例如，上面 4 个数据如果都是输入的，由于操作错误使总分不等于三科成绩之和，就会产生矛盾。下面关于冗余数据的说法中，正确的是（ ）。

A. 应该在数据库中消除一切冗余数据

B. 与用高级语言编写的数据处理系统相比，用关系数据库编写的系统更容易消除冗余数据

C. 为了提高查询效率，在数据库中可以适当保留一些冗余数据，但更新时要做相容性检验

D. 做相容性检验会降低效率，可以不理睬数据库中的冗余数据

19.单选题 [NOIP 普及组 2004] 下列哪个不是数据库软件的名称（ ）。

A. MySQL

B. SQL Server

C. Oracle

D. 金山影霸

E. Foxpro

20.不定项选择题 [NOIP 提高组 2004] 下列哪个（些）不是数据库软件的名称（ ）。

A. MySQL

B. SQLServer

C. Oracle

D. Outlook

E. Foxpro

21.不定项选择题 [NOIP 提高组 2006] 在下列各数据库系统软件中，以关系型数据库为主体结构的是（ ）。

A. ACCESS

B. SQL Server

C. Oracle

D. Foxpro

答案

1-5 F B C BC C

6-10 A F C BCD ABCD

11-15 E BE D D C

16-20 D D BC D D

21 ABCD

GW 信奥 CSP 初赛习题集 1-4

计算机基础知识——计算机语言与程序

1. 计算机语言

1.不定项选择题 [NOIP 提高组 2011/NOIP 普及组 2011] 汇编语言（ ）。

A. 是一种与具体硬件无关的程序设计语言

B. 在编写复杂程序时，相对于高级语言而言代码量较大，且不易调试

C. 可以直接访问寄存器、内存单元、 I/O 端口

D. 随着高级语言的诞生，如今已完全被淘汰，不再使用

2.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 面向对象程序设计（Object-Oriented Programming）是一种程序设计的方法论，它将对象作为程序的基本单元，将数据和程序封装在对象中，以提高软件的重用性、灵活性和扩展性。下面关于面向对象程序设计的说法中，不正确的是（ ）。

A. 面向对象程序设计通常采用自顶向下设计方法进行设计。

B. 面向对象程序设计方法具有继承性（inheritance）、封装性（encapsulation）、多态性（polymorphism）等几大特点。

C. 支持面向对象特性的语言称为面向对象的编程语言，目前较为流行的有 C++、JAVA、C#等。

D. 面向对象的程序设计的雏形来自于 Simula 语言，后来在 SmallTalk 语言的完善和标准化的过程中得到更多的扩展和对以前思想的重新注解。至今，SmallTalk 语言仍然被视为面向对象语言的基础

3.判断题 [GESP202303【1 级】/GESP 样题【2 级】] 程序员用 C、C++、Python、Scratch 等编写的程序能在 CPU 上直接执行。

4.判断题 [GESP202309【1 级】/GESP202309【2 级】] C++是一种高级程序设计语言。

5.单选题 [GESP202309【6 级】] 以下不属于面向对象程序设计语言的是（ ）。

A. C++

B. Python

C. Java

D. C

6.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 下列哪个程序设计语言不支持面向对象程序设计方法（ ）。

A. C++

B. Object Pascal

C. C

D. Smalltalk

E. Java

7.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 下列哪个（些）程序设计语言支持面向对象程序设计方法（ ）。

A. C++

B. Object Pascal

C. C

D. Smalltalk

E.Java

8.单选题 [NOIP 提高组 2014] 以下哪个是面向对象的高级语言（ ）。

A.汇编语言

B.C++

C.FORTRAN

D.Basic

9.单选题 [NOIP 普及组 2017] 下列不属于面向对象程序设计语言的是( )。

A. C

B. C++

C. Java

D. C#

10.不定项选择题 [NOIP 提高组 2017] 以下是面向对象的高级语言的是( )。

A. 汇编语言

B. C++

C. Fortan

D. Java

11.单选题 [CSP-J2021] 以下不属于面向对象程序设计语言的是（ ）。

A. C++

B. Python

C. Java

D. C

12.单选题 [NOIP 提高组 2018] 下列属于解释执行的程序设计语言是（ ）。

A. C

B. C++

C. Pascal

D. Python

13.单选题 [NOIP 普及组 2010] Pascal 语言、 C 语言和 C++ 语言都属于（ ）

A. 面向对象语言

B. 脚本语言

C. 解释性语言

D. 编译性语言

14.不定项选择题 [NOIP 提高组 2010] Pascal 语言、 C 语言、和 C++ 语言都属于（ ）

A. 高级语言

B. 自然语言

C. 解释型语言

D. 编译性语言

15.单选题 [NOIP 普及组 2005] 下列关于高级语言的说法错误的是（ ）。

A. Fortran 是历史上的第一个面向科学计算的高级语言

B. Pascal 和 C 都是编译执行的高级语言

C. C++是历史上的第一个支持面向对象的语言

D. 编译器将高级语言程序转变为目标代码

E. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

16.不定项选择题 [NOIP 提高组 2005] 下列关于高级语言的说法正确的有（ ）。

A. Ada 是历史上的第一个高级语言

B. Pascal 和 C 都是编译执行的高级语言

C. C++是历史上的第一个支持面向对象的语言

D. 编译器将高级语言程序转变为目标代码

E. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

17.单选题 [NOIP 普及组 2006] 在下列关于计算机语言的说法中，不正确的是（ ）。

A. Pascal 和 C 都是编译执行的高级语言

B. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

C. C++是历史上的第一个支持面向对象的计算机语言

D. 与汇编语言相比，高级语言程序更容易阅读

18.不定项选择题 [NOIP 提高组 2006] 在下列关于计算机语言的说法中，正确的有（ ）。

A. Pascal 和 C 都是编译执行的高级语言

B. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

C. C++是历史上的第一个支持面向对象的计算机语言

D. 高级语言比汇编语言更高级，是因为它的程序的运行效率更高

19.不定项选择题 [NOIP 提高组 2007/NOIP 普及组 2007] 在下列关于计算机语言的说法中，正确的有（ ）。

A. 高级语言比汇编语言更高级，是因为它的程序的运行效率更高

B. 随着 Pascal、C 等高级语言的出现，机器语言和汇编语言已经退出了历史舞台

C. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

D. C 是一种面向过程的高级计算机语言

20.单选题 [NOIP 普及组 2009] 关于程序设计语言，下面哪个说法是正确的：

A. 加了注释的程序一般会比同样的没有加注释的程序运行速度慢。

B. 高级语言开发的程序不能使用在低层次的硬件系统如：自控机床或低端手机上。

C. 高级语言相对于低级语言更容易实现跨平台的移植。

D. 以上说法都不对。

2. 计算机程序

21.单选题 [GESP202306【2 级】/GESP202306【3 级】/GESP202306【4 级】] 高级语言编写的程序需要经过以下（ ）操作，可以生成在计算机上运行的可执行代码。

A. 编辑

B. 保存

C. 调试

D. 编译

22.单选题 [GESP202403【1 级】] 在 Dev C++中对一个写好的 C++源文件要生成一个可执行程序需要执行下面哪个处理步骤？（ ）

A. 创建

B. 编辑

C. 编译

D. 调试

23.单选题 [CSP-J2020] 编译器的主要功能是（ ）。

A. 将源程序翻译成机器指令代码

B. 将源程序重新组合

C. 将低级语言翻译成高级语言

D. 将一种高级语言翻译成另一种高级语言

24.单选题 [CSP-S2019] 编译器的作用是（）。

A. 将源程序重新组合

B. 将一种语言（通常是高级语言）翻译成另一种语言（通常是低级语言）

C. 将低级语言翻译成高级语言

D. 将一种编程语言翻译成自然语言

25.单选题 [NOIP 提高组 2014] 编译器的主要功能是（）.

A.将一种高级语言翻译成另一种高级语言

B.将源程序翻译成指令

C.将低级语言翻译成高级语言

D.将源程序重新组合

26.判断题 [GESP 样题【1 级】] 注释是对于 C++程序员非常有用，但会被 C++编译器忽略。

27.判断题 [GESP202303【1 级】/GESP 样题【2 级】] 在 C++语言中，注释不宜写得过多，否则会使得程序运行速度变慢。

答案

1-5 BC A F T D

6-10 C C B A BD

11-15 D D D AD C

16-20 BDE C AB CD C

21-25 D C A D B

26-27 T F

GW 信奥 CSP 初赛习题集 1-5

计算机基础知识——计算机网络

1. 计算机网络分类

1.单选题 [NOIP 普及组 2007] LAN 的含义是（ ）。

A. 因特网

B. 局域网

C.广域网

D.城域网

2.单选题 [NOIP 普及组 2018] 广域网的英文缩写是（ ）。

A. LAN

B. WAN

C. MAN

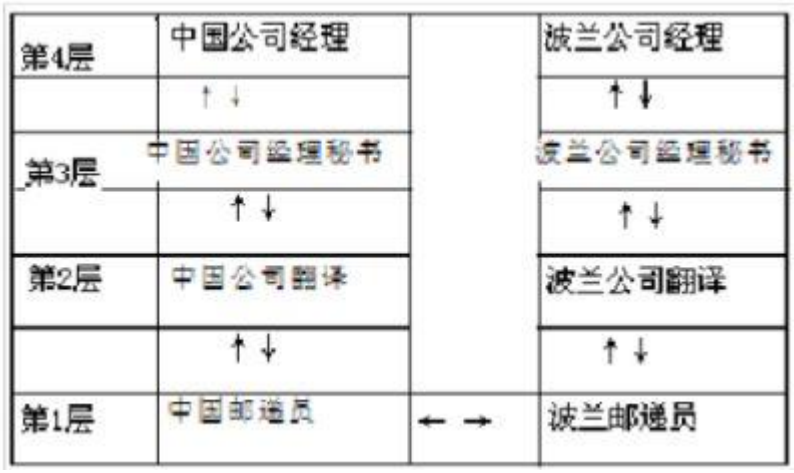
D. LNA

3.判断题 [GESP202309【2 级】] 我们常说的互联网（Internet）是一个覆盖全球的广域网络，它不属于任何一个国家。

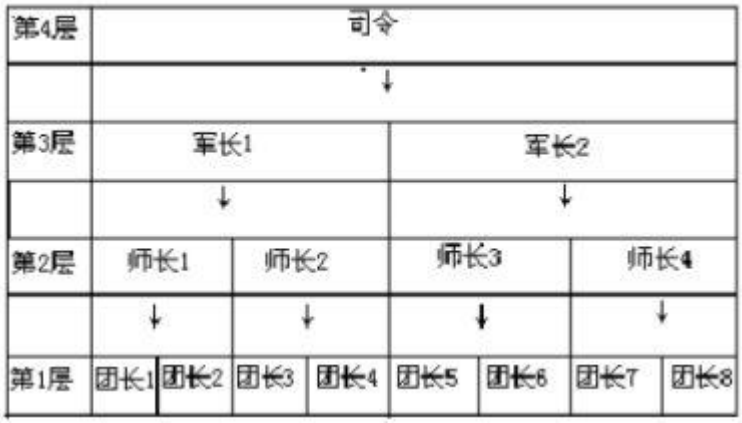
2. 计算机网络体系结构

4.单选题 [NOIP 普及组 2012/NOIP 提高组 2012] 无论是 TCP/IP 模型还是 OSI 模型，都可以视为网络的分层模型，每个网络协议都会被归入某一层中。如果用现实生活中的例子来比喻这些“层”，以下最恰当的是（ ）。

1. 中国公司的经理与缅甸公司的经理交互商业文件



B.军队发布命令



C.国际会议中，每个人都与他国地位对等的人直接进行会谈

D.体育比赛中，每一级比赛的优胜者晋级上一级比赛

5.单选题 [NOIP 提高组 2008] TCP/IP 是一组构成互联网基础的网络协议，字面上包括两组协议：传输控制协议（TCP）和网际协议（IP）。TCP/IP 协议把 Internet 网络系统描述成具有四个层次功能的网络模型,其中提供源节点和目的节点之间的信息传输服务，包括寻址和路由器选择等功能的是（ ）。

A. 链路层

B. 网络层

C. 传输层

D. 应用层

E. 会话层

3. 常见计算机网络协议与服务

6.单选题 [NOIP 普及组 2009] 关于互联网，下面的说法哪一个是正确的：

A. 新一代互联网使用的 IPv6 标准是 IPv5 标准的升级与补充。

B. 互联网的入网主机如果有了域名就不再需要 IP 地址。

C. 互联网的基础协议为 TCP/IP 协议。

D. 互联网上所有可下载的软件及数据资源都是可以合法免费使用的。

7.不定项选择题 [NOIP 提高组 2009] 关于计算机网络，下面的说法哪些是正确的：

A. 网络协议之所以有很多层主要是由于新技术需要兼容过去老的实现方案。

B. 新一代互联网使用的 IPv6 标准是 IPv5 标准的升级与补充。

C. TCP/IP 是互联网的基础协议簇，包含有 TCP 和 IP 等网络与传输层的通讯协议。

D. 互联网上每一台入网主机通常都需要使用一个唯一的 IP 地址，否则就必须注册一个固定的域名来标明其地址。

8.判断题 [GESP202303【2 级】] IPv4 的地址通常用“点分十进制”的表示形式，形如（a.b.c.d），其中 a、b、c、d 都是 1~255 之间的十进制整数（ ）。

9.单选题 [NOIP 普及组 2014/NOIP 提高组 2014] 下列几个 32 位 IP 地址中，书写错误的是（ ）。

A. 162.105.135.27

B. 192.168.0.1

C. 256.256.129.1

D. 10.0.0.1

10.不定项选择题 [NOIP 提高组 2015] 下列选项不是正确的 IP 地址的有（ ）。

A. 202.300.12.4

B. 192.168.0.3

C. 100:128:35:91

D. 11110335-21

11.单选题 [NOIP 普及组 2013/NOIP 提高组 2013] IPv4 协议使用 32 位地址，随着其不断被分配，地址资源日趋枯竭。因此，它正逐渐被使 用（ ）位地址的 IPv6 协议所取代。

A. 40

B. 48

C. 64

D. 128

12.判断题 [GESP202309【5 级】/GESP202309【6 级】] TCP/IP 的传输层的两个不同的协议分别是 UDP 和 TCP。

13.单选题 [NOIP 提高组 2014] TCP 协议属于哪一层协议 （ ）.

A.应用层

B.传输层

C.网络层

D.数据链路层

14.判断题 [GESP202306【2 级】/GESP202306【3 级】] 域名是由一串用点分隔的名字来标识互联网上一个计算机或计算机组的名称，CCF 编程能力等级认证官方网站的域名是 gesp.ccf.org.cn，其中顶级域名是 gesp。

15.判断题 [GESP202306【4 级】] 域名是由一串用点分隔的名字来标识互联网上一个计算机或计算机组的名称，CCF 编程能力等级认证官方网站的域名是 gesp.ccf.org.cn，其中顶级域名是 gesp。

16.单选题 [NOIP 普及组 2013/CSP-J2019] 中国的国家顶级域名是（ ）。

A. .cn

B. .ch

C. .chn

D. .china

17.单选题 [NOIP 普及组 2015] FTP 可以用于（ ） 。

A. 远程传输文件

B. 发送电子邮件

C. 浏览网页

D. 网上聊天

18.单选题 [NOIP 普及组 2005/NOIP 提高组 2005] 常见的邮件传输服务器使用（ ）协议接收邮件。

A. HTTP

B. SMTP

C. TCP

D. FTP

E. POP3

19.单选题 [NOIP 普及组 2012] （ ）是目前互联网上常用的 E-mail 服务协议

A. HTTP

B. FTP

C. POP3

D. Telnet

20.不定项选择题 [NOIP 提高组 2012] （ ）是目前互联网上常用的 E-mail 服务协议。

A. HTTP

B. FTP

C. POP3

D. SMTP

21.单选题 [NOIP 普及组 2014] 以下哪一种是属于电子邮件收发的协议（ ）。

A. SMTP

B. UDP

C. P2P

D. FTP

22.单选题 [NOIP 普及组 2017] 下列协议中与电子邮件无关的是( )。

A. POP3

B. SMTP

C. WTO

D. IMAP

23.不定项选择题 [NOIP 提高组 2011] 为计算机网络中进行数据交换而建立的规则、标准或约定的集合称为网络协议。 下列英 文缩写中，（ ）是网络协议。

A. HTTP

B. TCP/IP

C. FTP

D. WWW

4. 万维网与互联网

24.单选题 [NOIP 普及组 2004] 下列网络上常用的名字缩写对应的中文解释错误的是（ ）。

A. WWW（World Wide Web）：万维网。

B. URL（Uniform Resource Locator）：统一资源定位器。

C. HTTP（Hypertext Transfer Protocol）：超文本传输协议。

D. FTP（File Transfer Protocol）：快速传输协议。

E. TCP（Transfer Control Protocol）：传输控制协议

25.单选题 [NOIP 提高组 2004] 下列哪个网络上常用的名字缩写是错误的（ ）。

A. WWW（WorldWide Web）

B. URL（Uniform Resource Locator）

C. HTTP（Hypertext Transfer Protocol）

D. FTP（Fast Transfer Protocol）

E. TCP（Transfer Control Protocol）。

26.单选题 [NOIP 普及组 2012] （ ）是主要用于显示网页服务器或者文件系统的 HTML 文件的内容，并 让用户与这些文件交互的一种 软件。

A. 资源管理器

B. 浏览器

C. 电子邮件

D. 编译器

27.单选题 [NOIP 提高组 2012] （ ）是主要用于显示网页服务器或者文件系统的 HTML 文件的内容，并让用户与这些文件交互的一种软件。

A. 资源管理器

B. 浏览器

C. 电子邮件

D. 编译器

28.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] Web2.0 是近年来互联网的热门概念之一，其核心思想是互动与分享。下列网站中，（ ）是典型的 Web2.0 应用。

A. Sina

B. Flickr

C. Yahoo

D. Google

29.单选题 [GESP 样题【2 级】] 万维网 WWW 中存储了海量的数据资源，这里用于传输控制的协议是（ ）。

A. URL

B. SMTP

C. HTTP

D. HTML

30.单选题 [NOIP 提高组 2011] 为解决 Web 应用中的不兼容问题，保障信息的顺利流通， （ ）制订了一系列标准， 涉及 HTML、XML、CSS 等，并建议开发者遵循。

A. 微软

B. 美国计算机协会 (ACM)

C. 联合国教科文组织

D. 万维网联盟 (W3C)

31.单选题 [NOIP 普及组 2009] 关于 HTML 下面哪种说法是正确的：

A. HTML 实现了文本、图形、声音乃至视频信息的统一编码。

B. HTML 全称为超文本标记语言。

C. 网上广泛使用的 Flash 动画都是由 HTML 编写的。

D. HTML 也是一种高级程序设计语言。

32.不定项选择题 [NOIP 提高组 2009] 关于 HTML 下面哪些说法是正确的：

A. HTML 全称超文本标记语言，实现了文本、图形、声音乃至视频信息的统一编码。

B. HTML 不单包含有网页内容信息的描述，同时也包含对网页格式信息的定义。

C. 网页上的超链接只能指向外部的网络资源，本网站网页间的联系通过设置标签来实现。

D. 点击网页上的超链接从本质上就是按照该链接所隐含的统一资源定位符（URL）请求网络资源或网络服务。

5. 计算机网络设备

33.单选题 [NOIP 提高组 2004/NOIP 普及组 2004] 一台计算机如果要利用电话线上网，就必须配置能够对数字信号和模拟信号进行相互转换的设备，这种设备是（ ）。

A. 调制解调器

B. 路由器

C. 网卡

D. 网关

E. 网桥

34.不定项选择题 [NOIP 提高组 2016] 可以将单个计算机接入到计算机网络中的网络接入通讯设备有（ ）。

A. 网卡

B. 光驱

C. 鼠标

D. 显卡

35.判断题 [GESP202403【2 级】/GESP202403【3 级】/GESP202403【4 级】] 小杨今年春节回奶奶家了，奶奶家的数字电视要设置 ip 地址并接入到 WIFI 盒子才能收看节目，那这个 WIFI 盒子具有路由器的功能。

36.单选题 [NOIP2015 普及组] 计算机病毒是（ ）。

A. 通过计算机传播的危害人体健康的一种病毒

B. 人为制造的能够侵入计算机系统并给计算机带来故障的程序或指令集合

C. 一种由于计算机元器件老化而产生的对生态环境有害的物质

D. 利用计算机的海量高速运算能力而研制出来的用于疾病预防的新型病毒

37.单选题 [NOIP 普及组 2006] 在计算机中，防火墙的作用是（ ）。

A. 防止火灾蔓延

B. 防止网络攻击

C. 防止计算机死机

D. 防止使用者误删除数据

38.不定项选择题 [NOIP 提高组 2008] 在下列防火墙（firewall）的说法中，正确的有（ ）。

A. 防火墙是一项协助确保信息安全的设备，其会依照特定的规则，允许或是限制数据通过

B. 防火墙可能是一台专属的硬件或是安装在一般硬件上的一套软件

C. 网络层防火墙可以视为一种 IP 数据包过滤器，只允许符合特定规则的数据包通过，其余的一概禁止穿越防火墙

D. 应用层防火墙是在 TCP/IP 的“应用层”上工作，可以拦截进出某应用程序的所有数据包

6. 通信技术

39.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 以下不属于无线通信技术的是( )。

A. 蓝牙

B. WiFi

C. GPRS

D. 以太网

40.不定项选择题 [NOIP 提高组 2005] 下列外设接口中可以通过无线连接的方式连接设备的是（ ）。

A. USB 2.0 高速版

B. 红外

C. 蓝牙

D. 串口

E. IEEE 802.11g 无线网卡

41.单选题 [NOIP 普及组 2012] 蓝牙和 Wi-Fi 都是（ ）设备。

A. 无线广域网

B. 无线城域网

C. 无线局域网

D. 无线路由器

42.单选题 [NOIP 普及组 2015/NOIP 提高组 2015] 下列说法正确的是（ ）。

A. CPU 的主要任务是执行数据运算和程序控制

B. 存储器具有记忆能力，其中信息任何时候都不会丢失

C. 两个显示器屏幕尺寸相同，则它们的分辨率必定相同

D. 个人用户只能使用 Wifi 的方式连接到 I nternet

43.判断题 [GESP202309【6 级】] 5G 网络中，5G 中的 G 表示 Gigabytes/s，其中 1 GB = 1024 MB。

答案

1-5 B B T A B

6-10 C C F C ACD

11-15 D T B F F

16-20 A A E C CD

21-25 A C ABC D D

26-30 B B B C D

31-35 B BD A A T

36-40 B B ABCD D BCE

41-43 C A F

GW 信奥 CSP 初赛习题集 1-6

计算机基础知识——计算机基础操作

1.判断题 [GESP 样题【1 级】] 要执行 Windows 的桌面上的某个程序，通常需至少连续点击程序图标 3 次。

2.判断题 [GESP202303【1 级】] 在 Windows 系统中通过键盘完成对选定文本移动的按键组合是先 Ctrl+X，移动到目标位置后按 Ctrl+V。

3.单选题 [NOIP 普及组 2013] 在 Windows 资源管理器中，用鼠标右键单击一个文件时，会出现一个名为 “复制 ”的操作选项，它的意思是（ ） 。

A. 用剪切板中的文件替换该文件

B. 在该文件所在文件夹中，将该文件克隆一份

C. 将该文件复制到剪切板，并保留原文件

D. 将该文件复制到剪切板，并删除原文件

4.单选题 [NOIP 普及组 2011] 有人认为，在个人电脑送修前，将文件放入回收站中就是已经将其删除了。这种想法是（ ）。

A. 正确的，将文件放入回收站以为着彻底删除、无法恢复

B. 不正确的，只有将回收站清空后，才意味着彻底删除、无法恢复

C. 不正确的，即使回收站清空，文件只是被标记为删除，仍可能通过回复软件找回

D. 不正确的，只要在硬盘上出现过的文件，永远不可能被彻底删除

5.单选题 [NOIP 普及组 2013] 通常在搜索引擎中，对某个关键词加上双引号表示( )。

A. 排除关键词，不显示任何包含该关键词的结果

B. 将关键词分解，在搜索结果中必须包含其中的一部分

C. 精确搜索，只显示包含整个关键词的结果

D. 站内搜索，只显示关键词所指向网站的内容

答案

1-5 F T C C C

GW 信奥 CSP 初赛习题集 2-1

C++初级语法——常量变量

1. 常量

1.单选题 [NOIP 普及组 2010] 2E+03 表示（ ）

A. 2.03

B. 5

C. 8

D. 2000

2.判断题 [GESP202303【1 级】] '3'是一个 int 类型常量。

3.判断题 [GESP 样题【1 级】] 3.0 是一个 int 类型常量。

4.判断题 [GESP202306【1 级】] 10 是一个 int 类型常量。

5.判断题 [GESP 样题【2 级】] 5.0 是一个 int 类型常量。

6.单选题 [GESP 样题【1 级】/GESP202306【1 级】] 常量'3'的数据类型是（ ）。

A. int

B. char

C. bool

D. double

7.单选题 [GESP202303【1 级】] 常量 7.0 的数据类型是（）。

A. double

B. float

C. void

D. int

8.单选题 [GESP 样题【2 级】] 下列关于 C++语言的叙述，不正确的是（ ）。

A. 变量都有类型

B. 常量都有类型

C. 常量 1.0 的类型为 double

D. 常量'1'的类型为 int

2. 变量

1. 变量的命名规则

9.判断题 [GESP 样题【1 级】] C++语言中的标识符只能由大写字母和小写字母组成。

10.判断题 [GESP202303【1 级】] 在 C++语言中，标识符中可以有数字，但不能以数字开头。

11.判断题 [GESP202306【1 级】] 在 C++语言中，标识符的命名不能完全由数字组成，至少有一个字母就可以。

12.判断题 [GESP202303【2 级】] 在 C++语言中，标识符中可以有下划线_，但不能以下划线_开头。

13.判断题 [GESP202306【2 级】] 在 C++语言中，标识符中可以有下划线‘_’。同时，‘_’也是 C++语言的运算符。

14.判断题 [GESP202309【1 级】] 在 C++代码中，不可以将变量命名为 cout ，因为 cout 是 C++的关键字。

15.判断题 [GESP202403【1 级】] 在 C++的程序中，cin 是一个合法的变量名。

16.判断题 [GESP202312【1 级】/GESP202406【2 级】] 在 C++的程序中，不能用 scanf 作为变量名。

17.判断题 [GESP202406【1 级】] 在 C++代码中，不可以将变量命名为 printf ，因为 printf 是 C++语言的关键字。

18.单选题 [GESP202303【1 级】] 以下哪个不是 C++语言的关键字？

A. int

B. for

C. do

D. cout

19.单选题 [GESP202306【1 级】] 以下哪个不是 C++语言的关键字？（ ）

A. double

B. else

C. while

D. endl

20.单选题 [GESP202303【2 级】] 以下哪个不是 C++语言的关键字？

A． return

B． max

C． else

D．case

21.单选题 [GESP202306【2 级】] 以下哪个是 C++语言的关键字？（ ）

A. main

B. max

C. double

D. sqrt

22.单选题 [GESP202309【2 级】] 以下不是 C++关键字的是（ ）。

A. continue

B. cout

C. break

D. goto

23.单选题 [GESP202309【1 级】] 以下 C++不可以作为变量的名称的是（ ）。

A. redStar

B. RedStar

C. red_star

D. red star

24.单选题 [GESP 样题【1 级】] 不可以作为 C++标识符的是（ ）。

A. a_plus_b

B. a_b

C. a+b

D. ab

25.单选题 [GESP202303【1 级】] 以下不可以作为 C++标识符的是（）。

A. x321

B. 0x321

C. x321_ D. _x321

26.单选题 [GESP202306【1 级】] 以下可以作为 C++标识符的是（ ）。

A. number_of_Chinese_people_in_millions

B. 360AntiVirus

C. Man&Woman

D. break

27.单选题 [GESP202312【1 级】] 以下 C++不可以作为变量的名称的是( )。

A. CCF GESP

B. ccfGESP

C. CCFgesp

D. CCF_GESP

28.单选题 [GESP202312【2 级】] 以下不可以做为 C++变量的是( )。

A. FiveStar

B. fiveStar

C. 5Star

D. Star5

29.单选题 [GESP202406【1 级】/GESP202406【2 级】] 在 C++中，下列不可做变量的是( )。

A. five-Star

B. five_star

C. fiveStar

D. _fiveStar

30.单选题 [GESP 样题【2 级】] 不可以作为 C++标识符的是（ ）。

A. 0a0b

B. _a_b

C. _0ab

D. a0b0

31.单选题 [GESP202403【2 级】] 以下选项中不符合 C++变量命名规则的是？（ ）

A. student

B. 2_from

C. _to

D. Text

32.判断题 [GESP202403【2 级】] Xyz ， xYz ， xyZ 是三个不同的变量。

2. 变量的定义

33.单选题 [GESP 样题【1 级】] 按照 C++语言的语法，以下不是正确的变量定义语句是（ ）。

A. int a;

B. int a = 10;

C. int a(10);

D. a = 10;

34.单选题 [CSP-J2023] 在 C++ 中，下面哪个关键字用于声明一个变量，其值不能被修改?

A. unsigned

B. const

C. static

D. mutable

35.单选题 [GESP202303【2 级】] 下列关于 C++语言的叙述，不正确的是（）。

A． 变量定义后，可以使用赋值语句改变它的值

B． 变量定义时，必须指定类型

C． 变量名必须为合法标识符

D． 合法标识符可以以数字开始

3. 变量的存储大小

36.单选题 [GESP 样题【2 级】] 以下语句定义的变量占用 1 字节内存的是（ ）。

A. int a = 1;

B. int b = 'b';

C. bool c = true;

D. double d = 1.0;

37.单选题 [GESP202312【3 级】] 32 位计算机中，C++的整型变量 int 能够表示的数据范围是（ ）。

A. 2^{31}~2^{31}-1

B. 2^{32}

C. -2^{31}~2^{31}-1

D. -2^{31}+1~2^{31}

38.单选题 [GESP202406【3 级】] 一般默认 64 位计算机系统中整型变量（int）还是 32 位，则整数能够表示的数据范围是( )。

A. 0~2^{32}

B. 0~2^{64}

C. -2^{31}~2^{31}-1

D. -2^{63}~2^{63}-1

39.单选题 [GESP202303【2 级】] 下列关于 C++语言的叙述，不正确的是（ ）。

A． double 类型的变量占用内存的大小是浮动的

B． bool 类型的变量占用 1 字节内存

C． int 类型变量的取值范围不是无限的

D． char 类型的变量有 256 种取值

40.单选题 [CSP-J2019/NOIP 普及组 2013/NOIP 提高组 2013] 一个 32 位整型变量占用（ ）个字节。

A. 32

B. 128

C. 4

D. 8

41.判断题 [GESP202403【1 级】] C++语言中 3.0 和 3 的值相等，所以它们占用的存储空间也相同。

答案

1-5 D F F T F

6-10 B A D F T

11-15 F F F F T

16-20 F F D D B

21-25 C B D C B

26-30 A A C A A

30-35 B T D B D

36-40 C C C A C

41 F

GW 信奥 CSP 初赛习题集 2-2

C++初级语法——算数表达式

1. 简单算数表达式

1.判断题 [GESP 样题【1 级】] C++语言中也遵循与“先乘除、后加减”类似的运算符优先级规则。

2.单选题 [GESP 样题【1 级】] 下列符号不是 C++语言的运算符的是（ ）。

A. $

B. %

C. =

D. *

3.判断题 [GESP202312【4 级】] [(1,2)*2]*3 在 C++中是合法的表达式。

4.单选题 [GESP202312【1 级】] C++表达式 10 - 3 * (2 + 1) % 10 的值是( )。

A. 0

B. 1

C. 2

D. 3

5.单选题 [GESP202403【1 级】] C++表达式 (3 - 2) * 3 + 5 的值是( )。

A. -13

B. 8

C. 2

D. 0

6.单选题 [GESP 样题【1 级】] 表达式(3 + 12 / 3 * 2)的计算结果为（ ）。

A. 10

B. 5

C. 11

D. 2

7.单选题 [GESP202306【1 级】] 表达式(4 * (11 + 12) / 4)的计算结果为（ ）。

A. 47

B. 20

C. 23

D. 56

8.单选题 [GESP202306【1 级】] 如果用一个 int 类型的变量 a 表达正方形的边长，则下列哪个表达式不能用来计算正方形的面积？（ ）

A. a * a

B. 1 * a * a

C. a^ 2

D. a * 2 * a / 2

9.单选题 [GESP202303【1 级】] 如果用两个 int 类型的变量 a 和 b 分别表达长方形的长和宽，则下列哪个表达式不能用来计算长方形的周长？

A. a + b * 2

B. 2 * a + 2 * b

C. a + b + a + b

D. b + a * 2 + b

10.判断题 [GESP202403【1 级】] 在 C++代码中变量 n 被赋值为 27，则 cout << n%10 执行后输出的是 7。

2. 变量的类型转换

1. 隐式类型转换

11.判断题 [GESP202406【2 级】] C++表达式 -12 % 10 的值为 2。

12.判断题 [GESP202312【2 级】] 在 C++代码中，虽然变量都有数据类型，但同一个变量也可以先后用不同类型的值赋值。

13.单选题 [GESP202303【1 级】] 下列关于 C++语言的叙述，不正确的是（ ）。

A. 变量定义时可以不初始化

B. 变量被赋值之后的类型不变

C. 变量没有定义也能够使用

D. 变量名必须是合法的标识符

14.单选题 [GESP202306【1 级】] 下列关于 C++语言变量的叙述，正确的是（ ）。

A. 变量可以没有定义

B. 对一个没有定义的变量赋值，相当于定义了一个新变量

C. 执行赋值语句后，变量的类型可能会变化

D. 执行赋值语句后，变量的值可能不会变化

15.判断题 [GESP202312【2 级】] 在 C++代码中，运算符只能处理相同的数据类型，不同类型之间必须转换为相同的数据类型。

16.判断题 [GESP 样题【1 级】] 表达式(6.0/3,0)的计算结果为 2，且结果类型为 int。

17.判断题 [GESP202306【1 级】] 表达式(37 / 4)的计算结果为 9，且结果类型为 int。

18.判断题 [GESP202303【1 级】] 表达式(3.5 * 2)的计算结果为 7.0，且结果类型为 double。

19.判断题 [GESP202303【2 级】] 表达式(10.0 / 2)的计算结果为 5.0，且结果类型为 double。

20.判断题 [GESP202309【2 级】] C++表达式 7.8 / 2 的值为 3.9 ，类型为 float 。

21.判断题 [GESP202312【2 级】] C++表达式 -7/2 的值为整数-3。

22.判断题 [GESP202306【2 级】] 如果 a 是 double 类型的变量，而且值为 3.5，则表达式 a * 10 的计算结果为 35，且结果类型为 int。

23.单选题 [GESP202306【1 级】] 如果 a 和 b 为 int 类型的变量，且值分别为 7 和 2，则下列哪个表达式的计算结果不是 3.5？（ ）

A. 0.0 + a / b

B. (a + 0.0) / b

C. (0.0 + a) / b

D. a / (0.0 + b)

24.单选题 [GESP202303【2 级】] 如果 a、b、c 和 d 都是 int 类型的变量，则下列哪个表达式能够正确计算它们的平均值？

A． (a + b + c + d) / 4

B． (a + b + c + d) % 4

C． (a + b + c + d) / 4.0

D． (a + b + c + d) % 4.0

25.单选题 [GESP 样题【1 级】] 如果用两个 int 类型的变量 a 和 b 分别表达直角三角形两条直角边的长度，则下列哪个表达式可以用来计算三角形的面积？（ ）

A. a * b / 2

B. a / 2 * b

C. 1 / 2 * a * b

D. a * b * 0.5

26.单选题 [GESP 样题【2 级】] 如果用三个 int 类型的变量 a、b 和 h 分别表达梯形的上底、下底和高的长度，则下列哪个表达式可以用来计算梯形的面积？

A. h * (a + b) / 2

B. 0.5 * h * (a + b)

C. 0.5 * (h * a + b)

D. 0.5 * h * a + b

27.单选题 [GESP202406【1 级】] C++表达式 3 - 3 * 3 / 5 的值是( )。

A. -1.2

B. 1

C. 0

D. 2

28.单选题 [GESP202406【1 级】] 表达式 9/4 - 6 % (6 - 2) * 10 的值是( )。

A. -17.75

B. -18

C. -14

D. -12.75

29.单选题 [GESP202306【2 级】] 如果用两个 int 类型的变量 a 和 b 分别表达平行四边形的两条边长，用 int 类型的变量 h 表达 a 边对应的高，则下列哪个表达式不能用来计算 b 边对应的高？（ ）

A. a / b * (0.0 + h)

B. (0.0 + a * h) / b

C. a * h / (b + 0.0)

D. (a + 0.0) * h / b

30.判断题 [GESP202312【8 级】] 已知 int 类型的变量 a 、 b 和 h 中分别存储着一个梯形的顶边长、底边长和高，则这个梯形的面积可以通过表达式 (a + b) * h / 2 求得。

31.单选题 [GESP202303【1 级】] 如果 a 为 int 类型的变量，下列哪个表达式可以正确求出满足“大于等于 a 且是 4 的倍数”的整数中最小的？

A. a * 4

B. a / 4 * 4

C. (a + 3) / 4 * 4

D. a - a % 4 + 4

32.单选题 [GESP202312【3 级】] C++的数据类型转换让人很难琢磨透，下列代码输出的值是（ ）。

```
int a=3;
int b=2;
cout<<a/b*1.0<<endl;
```

A. 1.5

B. 1

C. 2

D. 1.50

33.判断题 [GESP202403【1 级】] C++表达式 ""10""*2 执行时将报错，因为 ""10"" 是字符串类型而 2 是整数类型，它们数据类型不同，不能在一起运算。

34.判断题 [GESP202406【1 级】] 在 C++中有整型变量 N，则表达式 N += 8/4//2 相当于 N += 8/(4/2) 。

2. 强制类型转换

35.单选题 [GESP202403【1 级】] 下面关于整型变量 int x 的赋值语句不正确是( )。

A. x=(3.16);

B. x=3.16;

C. x=int(3.16);

D. x=3.16 int;

36.判断题 [GESP202406【2 级】] C++表达式 int(12.56) 的值为 13。

37.判断题 [GESP202312【1 级】/GESP202309【1 级】] C++表达式 int(3.14) 的值为 3。

38.单选题 [GESP202309【2 级】] C++表达式 int(-123.123 / 10) 的值是（ ）。

A. -124

B. -123

C. -13

D. -12

39.单选题 [NOIP 提高组 2014/CSP-S2019] 若有变量 int a，float x，y，且 a=7，x=2.5， y=4.7， 则表达式 x+a%3*（int）（x+y）%2/4 的值大约是 ( ).

A.2.500000

B.2.750000

C.3.500000

D.0.000000

40.单选题 [NOIP 普及组 2014/CSP-S2019] 设变量 x 为 float 型且已赋值，则以下语句中能将 x 中的数值保留到小数点后两位，并 将第三位四舍五入的是（）。

A. x = (x * 100) + 0.5 / 100.0;

B. x = (x * 100 + 0.5) / 100.0;

C. x = (int)(x * 100 + 0.5)/100.0;

D. x = (x / 100 + 0.5) * 100.0;

41.判断题 [GESP202403【2 级】] 如果有以下 C++代码：

double s;

int t;

s =18.5;

t=int(s)+10;

那么 cout << t 的结果为 28.5 。

42.判断题 [GESP202406【1 级】] 定义 C++的 float 型变量 N ，则语句 cin >> N; cout << int(float(N)) 可以输入正负整数和浮点数，并将其转换为整数后输出。

43.判断题 [GESP202403【2 级】] C++中 cout << float(2022) 与 cout << float('2022') 运行后的输出结果均为 2022。

3. 赋值表达式

44.判断题 [GESP202403【2 级】] bool() 函数用于将给定参数或表达式转换为布尔类型。语句 bool(-1) 返回的是 false 值。

45.判断题 [GESP202306【2 级】] ++和==都是 C++语言的运算符，但+=不是。

46.判断题 [GESP202303【1 级】] 如果 a 为 int 类型的变量，则赋值语句 a = a + 3;是错误的，因为这条语句会导致 a 无意义。

47.判断题 [GESP202312【8 级】] C++语言非常强大，可以用来求解方程的解。例如，如果变量 x 为 double 类型的变量，则执行语句 x * 2- 4 = 0; 后，变量 x 的值会变为 2.0 。

48.单选题 [GESP 样题【1 级】] 如果 a 为 int 类型的变量，且 a 的值为 6，则执行 a = a + 3;之后，a 的值会是（ ）。

A. 0

B. 3

C. 6

D. 9

49.单选题 [GESP202303【2 级】] 如果 a 为 int 类型的变量，且 a 的值为 9，则执行 a -= 3;之后，a 的值会是（ ）

A． 3

B． 6

C． 9

D．12

50.单选题 [GESP202303【1 级】] 如果 a、b 和 c 都是 int 类型的变量，下列哪个语句不符合 C++语法？

A. c = a + b;

B. c += a + b;

C. c = a = b;

D. c = a ++ b;

51.单选题 [GESP202303【1 级】] 如果 a 为 int 类型的变量，且 a 的值为 6，则执行 a *= 3;之后，a 的值会是（ ）。

A. 3

B. 6

C. 9

D. 18

52.单选题 [GESP202306【1 级】] 如果 a 为 int 类型的变量，且 a 的值为 6，则执行 a %= 4;之后，a 的值会是（ ）。

A. 1

B. 2

C. 3

D. 4

53.单选题 [GESP 样题【2 级】] 如果 a 是已定义的 int 类型变量，以下 C++语言的语句不能通过编译的是（ ）。

A. a = 3.0;

B. int a = 3;

C. a = 3;

D. a = '3';

答案

1-5 T A F B B

6-10C C C A T

11-15 F T C D F

16-20 F T T T F

20-25 T F A C D

26-30 B D B A F

30-35 C B T F D

36-40 F T D A C

40-45 F T F F F

46–50 F F D B D

51–53 D B B

GW 信奥 CSP 初赛习题集 2-3

C++初级语法——关系逻辑表达式

1. 关系逻辑运算符

1.判断题 [GESP202309【4 级】] == 和 := 都是 C++语言的运算符。

2.判断题 [GESP202303【2 级】] C++语言中>=是运算符，但=>不是。

3.单选题 [GESP202303【2 级】] 以下哪个不是 C++语言的运算符？

A． \=

B． /=

C． -=

D．!=

4.单选题 [GESP202306【2 级】] 以下哪个不是 C++语言的运算符？（ ）

A. >=

B. /=

C. ||

D. <>

5.单选题 [GESP 样题【2 级】] 下列不是 C++语言的运算符的是（ ）。

A. >=

B. <=

C. ==

D. =>

2. 关系逻辑表达式的条件书写

6.判断题 [GESP202306【1 级】] 如果 a 和 b 为 int 类型的变量，则表达式 a = b 可以判断 a 和 b 是否相等。

7.判断题 [GESP202306【1 级】] 如果 a 为 int 类型的变量，则表达式(a % 4 == 2) 可以判断 a 的值是否为偶数。

8.判断题 [GESP 样题【2 级】] 如果 a 为 int 类型的变量，且表达式(a % 2 == 0) 的计算结果为假，说明 a 的值是奇数。

9.单选题 [GESP 样题【1 级】] 如果 a 为 int 类型的变量，下列表达式不能正确表达“a 是奇数时结果为 0，否则结果非 0”的是（ ）。

A. a %= 2

B. a / 2 * 2 == a

C. a % 2 == 0

D. (a + 1) % 2

10.判断题 [GESP 样题【1 级】] 如果 a 为 int 类型的变量，且表达式(a % 4 == 0) 的计算结果为真，说明 a 的值是 4 的倍数。

11.单选题 [GESP 样题【2 级】] 11. 如果 a 和 b 均为 int 类型的变量，下列表达式能够正确判断“a 是 b 的倍数”的是（）

A. a == b * ?

B. a % b = 0

C. a / b * b == a

D. b % a == 0

12.单选题 [GESP202303【1 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式不能正确判断“a 等于 0 且 b 等于 0”的是（ ）

A. (a == 0) && (b == 0)

B. (a == b == 0)

C. (!a) && (!b)

D. (a == 0) + (b == 0) == 2

13.单选题 [GESP202306【1 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式能正确判断“a 等于 0 且 b 等于 0”的是（ ）。

A. (a == b == 0)

B. !(a || b)

C. (a + b == 0)

D. (a == 0) + (b == 0)

14.单选题 [GESP202303【2 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式能正确判断“a 等于 0 或 b 等于 0” 的是（ ）

A． (!a) || (!b)

B． (a == b == 0)

C． (a == 0) && (b == 0)

D． (a == 0) - (b == 0) == 0

15.单选题 [GESP202306【2 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式能正确判断“a 等于 1 且 b 等于 1”的是（ ）。

A. (a == b) && (b == 1)

B. (a && b)

C. (a == b == 1)

D. (a * b == 1)

16.单选题 [GESP 样题【1 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式能够正确判断“a 不等于 0 或 b 不等于 0”的是（ ）

A. !a == 0 && !b == 0

B. !(a == 0 && b == 0)

C. (a != 0) && (b != 0)

D. a && b

17.单选题 [NOIP 提高组 2006/NOIP 普及组 2006] 在 C++ 中，判断 a 不等于 0 且 b 不等于 0 的正确的条件表达式是（ ）

A. !a==0 || !b==0

B. !((a==0)&&(b==0))

C. !(a==0&&b==0)

D. a!=0 || b!=0

E. a && b

18.单选题 [NOIP 提高组 2007/NOIP 普及组 2007] 在 C++语言中，判断 a 等于 0 或 b 等于 0 或 c 等于 0 的正确的条件表达式是（ ）

A. !((a!=0)||(b!=0)||(c!=0))

B. !((a!=0)&&(b!=0)&&(c!=0))

C. !(a==0&&b==0)||(c!=0)

D. (a=0)&&(b=0)&&(c=0)

E. !((a=0)||(b=0)||(c=0))

3. 关系逻辑表达式的值

19.判断题 [GESP202312【2 级】] C++表达式 3+2 && 5-5 的值为 false。

20.判断题 [GESP202309【2 级】] C++表达式 (2 * 3) || (2 + 5) 的值为 67 。

21.判断题 [GESP202312【3 级】] 执行 C++代码 cout<<(5&&2)<<endl; 后将输出 1 。

22.判断题 [GESP202312【3级】/GESP202312【4级】] 执行C++代码 cout<<(5||2); 后将输出 1 。

23.判断题 [GESP202303【1 级】] 如果 a 为 int 类型的变量，则表达式(a / 4 == 2)和表达式(a >= 8 && a <= 11)的结果总是相同的。

24.判断题 [GESP202309【2 级】] 如果 a 为 int 类型的变量，则表达式 (a >= 5 && a <= 10) 与 (5 <= a <= 10) 的值总是相同的。

25.单选题 [GESP202309【1 级】] C++表达式 2 - 1 && 2 % 10 的值是（ ）。

A. 0

B. 1

C. 2

D. 3

26.单选题 [GESP202309【1 级】] 在 C++语言中， int 类型的变量 x 、 y 、 z 的值分别为 2 、 4 、 6 ，以下表达式的值为真的是（ ）。

A. x > y || x > z

B. x != z - y

C. z > y + x

D. x < y || !x < z

27.单选题 [GESP202306【1 级】] 如果 a、b 和 c 都是 int 类型的变量，下列哪个语句不符合 C++语法？（ ）

A. a = (b == c);

B. b = 5.5;

C. c = a + b + c;

D. a + c = b + c;

28.判断题 [GESP202406【1 级】] C++中定义变量 int N ，则表达式 (!!N) 的值也是 N 的值。

答案

1-5 F T A D D

6-10 F F T A T

11-15 C B B A A

16-20 B E B F F

21-25 T T T F B

26-28 D D F

GW 信奥 CSP 初赛习题集 2-4

C++初级语法——顺序程序结构

1. C++的输入与输出

1.判断题 [GESP202403【1 级】] C++函数 scanf()必须含有参数，且其参数为字符串型字面量，其功能是提示输入。

2.单选题 [GESP202403【1 级】] C++语言中下面可以完成数据输入的语句是（ ）。

A. printf 语句

B. scanf 语句

C. default 语句

D. cout 语句

3.单选题 [GESP202403【1 级】] 执行 C++语句 cin >> a 时如果输入 5+2，下述说法正确的是（ ）。

A. 变量 a 将被赋值为整数 7

B. 变量 a 将被赋值为字符串，字符串内容为 5+2

C. 语句执行将报错，不能输入表达式

D. 依赖于变量 a 的类型。如果没有定义，会有编译错误

4.判断题 [GESP202306【1 级】] 在 C++语言中，计算结果必须存储在变量中才能输出。

5.判断题 [GESP202403【1 级】] C++语句 printf("%d#%d&",2,3) 执行后输出的是 2#3&。

6.判断题 [GESP202406【1 级】] C++的整型 N 被赋值为 5，语句 printf("%d*2",N) 执行后将输出 10 。

7.单选题 [GESP202406【1 级】] C++语句 printf("5%%2={%d}\n",5 % 2) 执行后的输出是( )。

A. 1={1}

B. 5%2={5%2}

C. 5%2={1}

D. 5 ={1}

8.单选题 [GESP202403【1 级】] C++语句 cout << "5%2=" << 5 % 2 执行后的输出是( )。

A. 2 2

B. 1 1

C. 5%2=2

D. 5%2=1

9.判断题 [GESP202406【1 级】] 在 C++代码中变量 X 被赋值为 16.44，则 cout << X / 10 执行后输出的一定是 1 。

10.判断题 [GESP202406【1 级】] C++的整型变量 N 被赋值为 10，则语句 cout << N / 4 << "->" << N % 4 << "->" << N / 4.0 执行后输出是 2->2->2.0 。

11.判断题 [GESP202406【2 级】] 执行 C++代码 cout << '9'+'1'; 的输出为 10。

12.判断题 [GESP202406【2 级】] C++的整型变量 N 被赋值为 10，则语句 cout << N / 3 << "-" << N % 3 执行后输出是 3-1。

13.单选题 [GESP202406【1 级】] 在 C++中，假设 N 为正整数，则表达式 cout << (N % 3 + N % 7) 可能输出的最大值是( )。

A. 6

B. 8

C. 9

D. 10

14.单选题 [GESP202406【1 级】] 对整型变量 i，执行 C++语句 cin >> i, cout << i 时如果输入 5+2 ，下述说法正确的是（ ）。

A. 将输出整数 7

B. 将输出 5

C. 语句执行将报错，输入表达式不能作为输出的参数

D. 语句能执行，但输出内容不确定

15.判断题 [GESP202403【2 级】] cout << (8< 9< 10) 的输出结果为 true 。

16.单选题 [GESP202406【2 级】] 在 C++中， cout << (5 % 2 && 5 % 3) 的输出是( )。

A. 1

B. 2

C. true

D. false

17.单选题 [GESP202406【3 级】] 下面可以正确输出 They're planning a party for their friend's birthday. 的 C++语句是？（ ）

A. cout << 'They\'re planning a party for their friend'\s birthday.' << endl;

B. cout << "They\'re planning a party for their friend's birthday.'" << endl;

C. cout << 'They're planning a party for their friend's birthday.' << endl;

D. cout << "They\'re planning a party for their friend\'s birthday." << endl;

2. 简单顺序结构程序

18.单选题 [GESP202403【1 级】] 下面 C++代码执行后的输出是（ ）。

```
int a=1;
printf("a+1=%d\n",a+1);
```

A. a+1= 2

B. a+1=2

C. 2=2

D. 2= 2

19.单选题 [GESP202403【1 级】] 下面 C++代码执行后的输出是（ ）。

```
int a=1;
cout<<"a+1= "<<a+1<<endl;
```

A. a+1= 2

B. a+1=2

C. 2=2

D. 2= 2

20.单选题 [GESP202309【1 级】] 下面 C++代码段执行后的输出是（ ）。

```
int a=3,b=4;
cout<<"a+b="<<a+b;
```

A. 3+4= 7

B. 3+4=7

C. a+b=7

D. a+b=a+b

21.填空题 [NOIP 普及组 2013]

```
#include <iostream>
using namespace std;
int main(){
```

```
    int a, b;
    cin>>a>>b;
    cout<<a<<"+"<<b<<"="<<a+b<<endl;
}
```

输入： 3 5

输出：

22.填空题 [NOIP 普及组 2012]

```
#include <iostream>
using namespace std;
int a, b, c, d, e, ans;
int main(){
    cin>>a>>b>>c;
    d = a+b;
    e = b+c;
    ans = d+e;
    cout<<ans<<endl;
}
```

输入： 1 2 5

输出： _______

23.填空题 [NOIP 普及组 2014]

```
#include <iostream>
using namespace std;
int main()
{
    int a, b, c, d, ans;
    cin >> a >> b >> c;
    d = a - b;
    a = d + c;
    ans = a * b;
    cout << "Ans = " << ans << endl;
    return 0;
}
```

输入：2 3 4

输出：____

24.单选题 [GESP202406【1 级】] 下面 C++代码执行后的输出是（ ）。

```
float a;
a = 101.101;
a = 101;
printf("a+1={%.0f}",a+1);
```

A. 102={102}

B. a+1={a+1}

C. a+1={102}

D. a 先被赋值为浮点数，后被赋值为整数，执行将报错

25.单选题 [GESP202406【2 级】] 某货币由 5 元，2 元和 1 元组成。输入金额（假设为正整数），计算出最少数量。为实现其功能，横线处应填入代码是（ ）。

```
int N;
cin >>N;
int M5,M2,M1;
M5 = N / 5;
M2 = _____;
M1 = _____;
printf("5*%d+2*%d+1*%d", M5, M2, M1);
```

A. 第 1 横线处应填入：N / 2

第 2 横线处应填入：N - M5 - M2

B. 第 1 横线处应填入：(N - M5 * 5) / 2

第 2 横线处应填入：N - M5 * 5 - M2 * 2

C. 第 1 横线处应填入：N - M5 * 5 / 2

第 2 横线处应填入：N - M5 * 5 - M2 * 2

D. 第 1 横线处应填入：(N - M5 * 5) / 2

第 2 横线处应填入：N - M5 - M2

3. 交换变量

26.单选题 [GESP202303【1 级】] 在下列代码的横线处填写（），可以使得输出是“20 10”。

```
#include <iostream>
using namespace std;
int main(){
    int a=10,b=20;
    a=______;//在此处填入代码
    b=a/100;
    a=a%100;
    cout<<a<<" "<<b<< endl;
    return 0;
}
```

A. a + b

B. (a + b) * 100

C. b * 100 + a

D. a * 100 + b

27.单选题 [GESP202306【1 级】] 在下列代码的横线处填写( ) , 使得输出是`20 10`。

```
#include <iostream>
using namespace std;
int main(){
    int a=10,b=20;
    a=______;//在此处填入代码
    b=a+b;
    a=b-a;
    cout<<a<<" "<<b<< endl;
    return 0;
}
```

A． a+b

B． b

C． a-b

D． b-a

28.单选题 [GESP202303【2 级】] 在下列代码的横线处填写( ) , 使得输出是`50 10`。

```
#include <iostream>
```

```
using namespace std;
int main(){
    int a=10,b=50;
    ______;//在此处填入代码
    b-=a;
    a+=b;
    cout<<a<<" "<<b<< endl;
    return 0;
}
```

A． a -= b

B． a += b

C． a = b - a

D． a = b

29.单选题 [GESP 样题【1 级】/GESP 样题【2 级】] 在下列代码的横线处填写（ ），可以使得输出是“20 10”。

```
#include <iostream>
using namespace std;
int main(){
    int a=10,b=20;
    ________//在此处填入代码
    cout<<a<<" "<<b<< endl;
    return 0;
}
```

A. a = b; b = a;

B. a = max(a, b); b = min(a, b);

C. a = a + b; a = a - b; b = a - b;

D. int tmp = a; a = b; b = tmp;

30.判断题 [GESP202406【8 级】] 已知 double 类型的变量 a 和 b ，则执行语句 a = a + b; b = a - b; a = a - b; 后，变量 a 和 b 的值会互换。

答案

1-5 F B D F T

6-10 F C D F F

11-15 F T B B T

16-20 A D B A C

21-25 3+5=8 10 Ans=9 C B

26-30 D C C D F

GW 信奥 CSP 初赛习题集 2-6

C++初级语法——循环程序结构

1. 循环结构概述

1. 基本概念

1.单选题 [GESP 样题【1 级】] 下列关于 C++语言的叙述，不正确的是（ ）。

A. 变量使用前必须先定义

B. if 语句中的判断条件必须写在 ()中

C. for 语句的循环体必须写在{}中

D. 程序必须先编译才能运行

2.单选题 [GESP202306【2 级】] 下列关于 C++语言的叙述，不正确的是（ ）。

A. if 语句中的判断条件必须用小括号‘(’和‘)’括起来。

B. for 语句中两个‘;’之间的循环条件可以省略，表示循环继续执行的条件一直满足。

C. 循环体包含多条语句时，可以用缩进消除二义性。

D. 除了“先乘除、后加减”，还有很多运算符优先级。

3.判断题 [GESP202403【1 级】/GESP202403【2 级】/GESP202403【3 级】/GESP202403【4 级】] 任何一个 for 循环都可以转化为等价的 while 循环（ ）

4.判断题 [GESP 样题【1 级】/GESP202312【1 级】/GESP202312【2 级】/GESP202312【3 级】/GESP202312【4 级】] 任何一个 while 循环都可以转化为等价的 for 循环

2. 循环次数控制

5.判断题 [GESP202303【2 级】] while 语句的循环体至少会执行一次。

6.判断题 [GESP202303【1 级】] for 语句的循环体至少会执行一次。

7.判断题 [GESP202306【1 级】] do ... while 语句的循环体至少会执行一次。

8.判断题 [GESP202312【2 级】] C++代码中 while(1){...} 的判断条件不是逻辑值，将导致语法错误。

9.判断题 [GESP202306【2 级】/GESP 样题【2 级】] 循环语句的循环体有可能无限制地执行下去。

10.判断题 [GESP202403【1 级】] 在 C++中，while 可能是死循环，而 for 循环不可能是死循环。

11.判断题 [GESP202309【1 级】] 在 C++语言中，do-while 循环不可能导致死循环，但 while 有可能。

12.单选题 [GESP202306【2 级】] 以下哪个循环语句会无限次执行？（ ）

A. for (int a = 0; a; a++) ;

B. for (bool b = false; b <= true; b++) ;

C. for (char c = 'A'; c < 'z'; c++) ;

D. for (double d = 0.0; d < 10.0; d += 0.001) ;

13.判断题 [GESP202406【2 级】] 下面 C++代码执行后将导致死循环。

```
for (int i = 0; i < 10; i++)
continue;
```

14.单选题 [NOIP 提高组 2007/NOIP 普及组 2007] 一个无法靠自身的控制终止的循环称为“死循环”，例如，在 C 语言程序中，语句“while(1) printf(“*”);”就是一个死循环，运行时它将无休止地打印*号。下面关于死循环的说法中，只有（ ）是正确的。

A. 不存在一种算法，对任何一个程序及相应的输入数据，都可以判断是否会出现死循环，因而，任何编译系统都不做死循环检验

B．有些编译系统可以检测出死循环

C. 死循环属于语法错误，既然编译系统能检查各种语法错误，当然也应该能检查出死循环

D. 死循环与多进程中出现的“死锁”差不多，而死锁是可以检测的，因而，死循环也是可以检测的

E. 对于死循环，只能等到发生时做现场处理，没有什么更积极的手段

3. 循环辅助语句

15.判断题 [GESP202406【1 级】] 在 C++， continue 语句通常与 if 语句配合使用。

16.判断题 [GESP202403【1 级】/GESP202406【1 级】] 在 C++，break 语句用于提前终止当前层次循环，适用于 while 循环，但不适用于 for 循环。

17.判断题 [GESP202312【1 级】] 在下面的 C++代码 while(1) continue; 中，由于循环中的 continue 是无条件被执行，因此将导致死循环。

18.判断题 [GESP202309【1 级】] 在下面的 C++代码中 for(int i=1;i<10;i++)continue;，由于循环中的 continue 是无条件被执行，因此将导致死循环。

19.单选题 [GESP202403【2 级】] 下列说法错误的是？（ ）

A. while 循环满足循环条件时不断地运行，直到指定的条件不满足为止

B. if 语句通常用于执行条件判断

C. 在 C++中可以使用 foreach 循环

D. break 和 continue 语句都可以用在 for 循环和 while 循环中

20.判断题 [GESP 样题【2 级】] 如果没有 break 语句，有些功能就无法实现了。

2. 简单循环

1. for 循环

21.判断题 [GESP202403【2 级】] for (i = 0; i < 100; i+=2)；语句中变量 i 的取值范围是 0 到 99。

22.判断题 [GESP202312【1 级】] for(int i = 1; i < 10; i += 3;)；表示 i 从 1 开始到 10 结束间隔为 3，相当于 1、4、7、10。

23.判断题 [GESP202309【1 级】] C++的循环语句 for (int i = 0; i < 10; i += 2) 表示 i 从 0 开始到 10 结束但不包含 10 ，间隔为 2 。

24.单选题 [GESP 样题【1 级】] 在下列代码的横线处填写（ ），可以使得输出是“111111”。

```
#include<iostream>
using namespace std;
int main(){
    for(int i=1;i<=16;_____){//在此处填入代码
        cout<<1;
    }
    return 0;
}
```

A. i++

B. i += 3

C. i *= 2

D. i = i * 3 - 1

25.单选题 [GESP202406【2 级】] 在 C++中，与 for(int i=0; i<10; i++) 效果相同的是( )。

A. for(int i=0; i<10; i+=1)

B. for(int i=1; i<=10; i++)

C. for(int i=10; i>0; i--)

D. for(int i=10; i<1; i++)

26.单选题 [GESP202312【2 级】] 在 C++中，与 for(int i = 10; i < 20; i +=2) cout << i; 输出结果相同的是( )。

A. for(int i = 10; i < 19; i +=2) cout << i;

B. for(int i = 11; i < 19; i +=2) cout << i;

C. for(int i = 10; i < 21; i +=2) cout << i;

D. 以上均不对

27.单选题 [GESP202303【1 级】] 在下列代码的横线处填写（ ），可以使得输出是“1248”。

```
#include<iostream>
using namespace std;
int main(){
    for(int i=1;i<=8;_____){//在此处填入代码
        cout<<i;
    }
    return 0;
}
```

A. i++

B. i *= 2

C. i += 2

D. i * 2

28.单选题 [GESP 样题【2 级】] 在下列代码的横线处填写（ ），可以使得输出是“1357”。

```
#include<iostream>
using namespace std;
int main(){
    for(int i=1;i<=8;_____){//在此处填入代码
        cout<<i;
    }
    return 0;
}
```

A. i++

B. i + 2

C. i *= 2

D. i += 2

29.单选题 [GESP202403【1 级】] 下面 C++代码第 2 行，总共被执行次数是（ ）。

```
int cnt=0;
for(int i=-10;i<10;i++){
    cout<<i<<"" "";
}
```

A. 10

B. 19

C. 20

D. 21

30.单选题 [GESP202309【1 级】] 下面 C++代码段执行后的输出是（ ）。

```
int cnt=0;
for(int i=1;i<=5;i++){
    cnt=cnt+1;
}
```

```
cout<<cnt;
```

A. 1

B. 4

C. 5

D. 10

31.单选题 [GESP202312【1 级】] 下面 C++代码执行后的输出是（ ）。

```
int cnt=0;
for(int i=1;i<10;i++){
    cnt+=1;
    i+=2;
}
cout<<cnt;
```

A. 10

B. 9

C. 3

D. 1

32.单选题 [GESP202309【1 级】] 下面 C++代码段执行后的输出是（ ）。

```
int tnt=0;
for(int i=1;i<5;i+=2){
    tnt=tnt+i;
}
cout<<tnt;
```

A. 1

B. 4

C. 5

D. 10

33.单选题 [GESP202312【1 级】] 执行下面 C++代码后输出是（ ）。

```
int cnt=0;
```

```
for(int i=10;i>3;i-=3){
    cnt=cnt+i;
}
cout<<cnt;
```

A. 3

B. 21

C. 27

D. 49

34.判断题 [GESP202309【2 级】] 下面 C++代码执行后的输出为 10 。

```
int cnt =0;
for(int i=1;i<10;i++){
    cnt += 1;
    i += 1;
}
cout << cnt;
```

35.判断题 [GESP202309【2 级】] 执行以下 C++代码后的输出为 0 。

```
int rst = 0;
for(int i=-100;i<100;i += 2)
    rst += i;
cout << rst;
```

36.判断题 [GESP202309【2 级】] 执行以下 C++代码后的输出为 30 。

```
int rst= 0;
for(int i=0;i<10;i += 2)
    rst += i;
cout << rst;
```

37.判断题 [GESP202312【2 级】] 执行以下 C++代码后的输出为 30 。

```
int sum= 0;
for(int i=-500;i<500;i ++)
    sum += i;
```

```
cout << sum;
```

38.单选题 [GESP202406【1 级】] 执行下面 C++代码后输出的 cnt 的值是（ ）。

```
int cnt=0;
for(int i = 0; i*i < 64; i+=2)
    cnt++;
cout << cnt;
```

A. 8

B. 7

C. 4

D. 1

39.单选题 [GESP202312【1 级】] 下面对 C++代码执行后输出的描述，正确的是（ ）。

```
cin>>N;
cnt=0;
for(int i=1;i<N;i++){
    cnt+=1;
}
cout<<cnt;
```

A. 如果输入的 N 小于 2 整数，第 5 行将输出 0。

B. 如果输入的 N 是大于等于 2 整数，第 5 行将输出 N-1。

C. 如果输入的 N 是大于等于 2 整数，第 5 行将输出 N。

D. 以上说法均不正确。

40.单选题 [GESP202403【2 级】] 下面 C++代码执行后的输出是？（）

```
int n,a,m,i;
n=3,a=5
m=(a-1)*2;
for(i=0; i<n-1; i++)
    m=(m-1)*2;
cout << m;
```

A. 8

B. 14

C. 26

D. 50

41.判断题 [GESP202309【2 级】] 如果 m 和 n 为 int 类型变量，则执行下列代码之后 n 的值为偶数。

```
for (m = 0, n = 1; n < 9; )
    n = ((m = 3 * n, m + 1), m -1);
```

42.单选题 [NOIP 普及组 2014/NOIP 普及组 2016] 若有如下程序段，其中 s、a、b 、c 均已定义为整型变量，且 a、 c 均已赋值， c>0 。

```
s = a;
for(b = 1; b <= c; b++)
    s += 1;
```

则与上述程序段功能等价的赋值语句是( )。

A. s = a + b

B. s = a + c

C. s = s + c

D. s = b + c

43.单选题 [CSP-J2019] 若有如下程序段，其中 s、a、b、c 均已定义为整型变量，且 a、c 均已赋值（c 大于 0）

```
s=a;
for(b=1;b<=c;b++)
    s=s-1;
```

则与上述程序段功能等价的赋值语句是（ ）

A. s = a - c;

B. s = a - b;

C. s = s - c;

D. s = b - c;

44.单选题 [NOIP 普及组 2014] 要求以下程序的功能是计算：

s=1+1/2+1/3+...+1/10

```
#include<iostream>
using namespace std;
int main()
{
    int n;
    float s;
    s = 1.0;
    for(n = 10; n > 1; n--)
        s = s + 1 / n;
    cout << s << endl;
    return 0;
}
```

程序运行后输出结果错误，导致错误结果的程序行是（ ）。

A. s = 1.0;

B. for(n = 10; n > 1; n--)

C. s = s + 1 / n;

D. cout << s <<endl;

45.单选题 [GESP202309【1 级】] 在下列代码的横线处填写（ ），可以使得输出是正整数 1234 各位数字的平方和。

```
int n=1234,s=0;
for(;n;n/=10){
    s+=______;//此处填写代码
}
cout<<s<<endl;
```

A. n / 10

B. (n / 10) * (n / 10)

C. n % 10

D. (n % 10) * (n % 10)

46.单选题 [GESP202406【1 级】] 下面 C++代码执行后输出是（ ）。

```
int Sum = 0, i = 0;
for ( ; i < 10; )
    Sum += i++;
cout << i << "" "" << Sum;
```

A. 9 45

B. 10 55

C. 10 45

D. 11 55

47.单选题 [GESP202309【1 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int n=5,s=1;
for(;n=0;n--){
    s*=n;
}
cout<<s<<endl;
```

A. 1

B. 0

C. 120

D. 无法确定

2. while 与 do-whlie 循环

48.单选题 [GESP202309【1 级】] 下面 C++代码执行后的输出是（ ）。

```
int n=5;
int cnt=1;
while(n>=0){
    cnt+=1;
    n-=2;
}
cout<<cnt;
```

A. 3

B. 4

C. 6

D. 7

49.填空题 [NOIP 普及组 2011]

```
#include <iostream>
using namespace std;
int main(){
    int i, n, m, ans;
    cin>>n>>m;
    i = n;
    ans = 0;
    while (i <= m){
        ans += i;
        i++;
    }
    cout<<ans<<endl;
    return 0;
}
```

输入： 10 20

输出： _____

50.判断题 [GESP202406【2 级】] 下面 C++代码能实现正整数各位数字之和。

```
int N,Sum = 0;
cin >> N;
while (N){
    Sum += N % 10;
    N /= 10;
}
cout << Sum;
```

51.单选题 [GESP202406【2 级】] 假设下面 C++代码执行过程中仅输入正负整数或 0，有关说法错误的是（ ）。

```
int N,Sum = 0;
cin >> N;
while (N){
    Sum += N;
    cin >> N;
```

```
}
cout << Sum;
```

A. 执行上面代码如果输入 0，将终止循环

B. 执行上面代码能实现所有非 0 整数的求和

C. 执行上面代码第一次输入 0，最后将输出 0

D. 执行上面代码将陷入死循环，可将 while (N) 改为 while (N==0)

52.单选题 [NOIP 普及组 2013] 下列程序中，正确计算 1, 2, …, 100 这 100 个自然数之和 sum （初始值为 0 ） 的是 （ ）。

程序段 1

```
i = 1;
do {
    sum += i;
    i++;
} while (i <= 100);
```

程序段 2

```
i = 1;
do {
    sum += i;
    i++;
} while (i > 100);
```

程序段 3

```
i = 1;
while (i < 100) {
    sum += i;
    i++;
}
```

程序段 4

```
i = 1;
while (i >= 100) {
    sum += i;
    i++;
}
```

A. 程序段 1

B. 程序段 2

C. 程序段 3

D. 程序段 4

53.不定项选择题 [NOIP提高组2013] 下列程序中，正确计算 1,2,，，100 这 100 个自然数之和 sum（初始值为 0）的是（ ）。

A.

```
for (i = 1; i <= 100; i++)
    sum += i;
```

B.

```
i = 1;
while (i > 100) {
    sum += i;
    i++;
}
```

C.

```
i = 1;
do {
    sum += i;
    i++;
}
while (i <= 100);
```

D.

```
i = 1;
do {
    sum += i;
    i++;
}
while (i > 100);
```

54.单选题 [NOIP 普及组 2014] 有以下程序

```
#include <iostream>
```

```
using namespace std;
int main()
{
    int s, a, n;
    s = 0;
    a = 1;
    cin >> n;
    do
    {
        s += 1;
        a -= 2;
    } while(a != n);
    cout << s << endl;
    return 0;
}
```

若要使程序的输出值为 2 ，则应该从键盘给 n 输入的值是（）。

A. -1

B. -3

C. -5

D. 0

55.填空题 [NOIP 普及组 2012]

```
include<iostream>
using namespace std;
int n, i, ans;
int main(){
    cin>>n;
    ans = 0;
    for (i = 1; i <= n; i++)
        if (n % i == 0)ans++;
    cout<<ans<<endl;
}
```

输入： 18

输出： _____

3. 循环判断

1. for 循环中判断

56.单选题 [GESP202312【3 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int cnt=0;
for(int i=0;i<=20;i++){
    if(i%3==0||i%5==0)
        cnt++;
}
cout<<cnt<<endl;
```

A. 2

B. 3

C. 5

D. 4

57.单选题 [GESP202303【1 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int sum=0;
for(int i=1;i<=20;i++){
    if(i%3==0||i%5==0)
        sum+=i;
    }
cout<<sum<<endl;
```

A. 210

B. 113

C. 98

D. 15

58.单选题 [GESP202306【1 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int sum;
for(int i=1;i<=20;i++){
    if(i%3==0||i%5==0)
        sum+=i;
    }
```

```
cout<<sum<<endl;
```

A. 63

B. 98

C. 113

D. 无法确定

59.单选题 [GESP 样题【2 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int sum=0;
for(int i=1;i<=20;i++){
    if（20%i==0)
        sum+=i;
    }
cout<<sum<<endl;
```

A. 20

B. 42

C. 22

D. 210

60.单选题 [NOIP 普及组 2013]

```
#include <iostream>
using namespace std;
int main(){
    int a, b, u, i, num;
    cin>>a>>b>>u;
    num = 0;
    for (i = a; i <= b; i++)
        if ((i % u) == 0) num++;
    cout<<num<<endl;
    return 0;
}
```

输入： 1 100 15

输出：

61.填空题 [NOIP 提高组 2013]

```
#include <iostream>
using namespace std;
int main(){
    int a, b, u, v, i, num;
    cin>>a>>b>>u>>v;
    num = 0;
    for (i = a;i <= b;i++)
        if (((i % u) == 0) || ((i % v) == 0))
            num++;
    cout<<num<<endl;
    return 0;
}
```

输入：1 1000 10 15

输出：_____

62.单选题 [GESP202306【2 级】] 在下列代码的横线处填写（ ），可以使得输出是 42。

```
int sum;
for(int i=1;i<=20;i++){
    if(______)//在此处填入代码
        sum+=i;
    }
cout<<sum<<endl;
```

A. i % 3 == 0

B. 20 % i == 0

C. i <= 8

D. i >= 18

63.单选题 [GESP202306【1 级】] 在下列代码的横线处填写（ ），可以使得输出是“147”。

```
#include<iostream>
using namespace std;
int main(){
    for(int i=1;i<=8;i++)
```

```
        if(____)//在此处填入代码
            cout<<i;
    return 0;
}
```

A. i % 2 == 1

B. i % 3 == 1

C. i = i + 3

D. i + 3

64.单选题 [GESP 样题【8 级】] 下列程序的输出是（ ）。

```
int main(){
    int sum=0,x;
    for(x=-10;x<=10;x++)
        if((3*x+5>=-11)&&(3*x+5<=11))
            sum+=x;
    cout<<sum;
    cout<<endl;
}
```

A. -12

B. 0

C. 55

D. -55

65.单选题 [GESP202406【4 级】] 下面程序输出的是（ ）

```
#include<iostream>
using namespace std;
int func();
int main()
{
    int i=2;
    cout<<i<<endl;
    for(int x=0;x<1;x++)
    {
        int i=10;
        cout<<i<<endl;
    }
```

```
13      i=i+1;
14      cout<<i<<endl;
15      {
16          i=i*i;
17          cout<<i<<endl;
18      }
19  }
```

A. 2 2 3 9

B. 2 10 3 9

C. 2 10 11 121

D. 2 10 3 100

66.填空题 [NOIP 普及组 2018/NOIP 提高组 2018]

```
1   #include <cstdio>
2   int main() {
3       int x;
4       scanf(""%d"", &x);
5       int res = 0;
6       for (int i = 0; i < x; ++i) {
7           if (i * i % x == 1) {
8               ++res;
9           }
10      }
11      printf(""%d"", res);
12      return 0;
13  }
```

输入：15

输出：

67.填空题 [NOIP 提高组 2014]

```
1   #include<iostream>
2   using namespace std;
3
4   int main(){
5       int a, b, i, tot, c1, c2;
6       cin >> a >> b;
7       tot = 0;
```

```
    for (i = a; i <= b; i++){
        c1 = i / 10;
        c2 = i % 10;
        if((c1 + c2) % 3 == 0)
            tot++;
    }
    cout << tot << endl;
}
```

输入：7 31

输出：（ ）

68.单选题 [GESP 样题【1 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
int sum=1;
for(int i=1;i<=10;i++){
    if(3<=i<=5)
        sum+=i;
    }
cout<<sum<<endl;
```

A. 56

B. 13

C. 12

D. 60

69.单选题 [GESP202309【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int n=9;
for(int i=2;i<n;i++){
    if(n%i)
        cout<<""1#"";
}
cout<<""0""<<endl;
```

A. 1#0

B. 1#

C. 1#1#1#1#1#1

D. 1#1#1#1#1#1#0

70.单选题 [GESP202406【2 级】] 下面 C++代码执行后，输出是（ ）。

```
int cnt1 = 0, cnt2 = 0;
for (int i = 0; i < 10; i++){
    if (i % 2 == 0)
        continue;
    if (i % 2)
        cnt1 += 1;
    else if (i % 3 == 0)
        cnt2 += 1;
}
cout << cnt1 << "" "" << cnt2;
```

A. 5 2

B. 5 0

C. 0 2

D. 0 0

71.单选题 [GESP202403【1 级】] 下面 C++代码执行后的输出是（ ）。

```
int tnt=0;
for(int i=0;i<10;i++){
    if(i%3&&i%7)
        tnt+=i;
}
cout<<tnt<<endl;
```

A. 0

B. 7

C. 18

D. 20

72.单选题 [GESP202312【1 级】] 下面 C++代码执行后的输出是（ ）。

```
cnt=0;
for(int i=1;i<20;i++){
    if(i%2)
        continue;
    else if(i%3==0&&i%5==0)
```

```
        break;
    cnt+=i;
}
cout<<cnt;
```

A. 90

B. 44

C. 20

D. 10

2. while 与 do-whlie 循环中判断

73.单选题 [GESP202403【1 级】] 下面 C++代码执行后的输出是（ ）。

```
cnt=0;
while(N){
    N-=1;
    if(N%3==0)
        cout<<N<<""#""
}
```

A. 9#6#3#

B. 9#6#3#0#

C. 8#7#5#4#2#1#

D. 10#8#7#5#4#2#1#

74.单选题 [GESP202312【1 级】] 下面 C++代码执行后的输出是（ ）。

```
cnt=0;
N=0;
while(1){
    if(N==0)break;
    cnt+=1;
    N-=2;
}
```

A. 11

B. 10

C. 5

D. 4

75.单选题 [GESP202312【2 级】] 下面 C++代码执行后的输出是（ ）。

```
N=100;
while(N>0){
    if(N%2)
        break;
    else if(N%3==0)
        N-=5;
    else
        N=-20;
    cout<<N;
}
```

A. 100

B. 95

C. 55

D. 0

76.单选题 [GESP202403【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int s,t,ans;
s=2,t=10;
ans=0;
while(s!=t){
    if(t%2==0&&t/2>=s)
        t/=2;
    else
        t-=1;
    ans+=1;
}
cout<<ans;
```

A. 2

B. 3

C. 4

D. 5

77.单选题 [GESP202403【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int n,i,result;
n=81;
i=1,result=1;
while(i*i<=n){
    if(n%(i*i)==0)
        result=i*i;
    i+=1;
}
cout<<result;
```

A. 16

B. 36

C. 49

D. 81

78.单选题 [GESP202406【2 级】] 在下面的 C++代码中，N 必须是小于 10 大于 1 的整数，M 为正整数（大于 0）。如果 M 被 N 整除则 M 为幸运数，如果 M 中含有 N 且能被 N 整除，则为超级幸运数，否则不是幸运数。程序用于判断 M 是否为幸运数或超级幸运数或非幸运数。阅读下面代码，有关说法正确的是（ ）。

```
int N, M;
cout << ""请输入幸运数字："";
cin >> N;
cout << ""请输入正整数："";
cin >> M;
bool Lucky;
if (M % N == 0)
    Lucky = true;
else
    Lucky = false;
while (M){
    if (M % 10 == N && Lucky){
        printf(""%d 是%d 的超级幸运数!"", M, N);
        break;
    }
    M /= 10;
}
if (M == 0)
    if (Lucky)
```

```
20            printf(""%d 是%d 的幸运数!"", M, N);
21        else
22            printf(""%d 非%d 的幸运数!"", M, N);
```

A. 如果 N 输入 3，M 输入 36 则将输出：36 是 3 的超级幸运数!

B. 如果 N 输入 7，M 输入 21 则将输出：21 是 7 的幸运数!

C. 如果 N 输入 8，M 输入 36 则将输出：36 非 8 的超级幸运数!

D. 如果 N 输入 3，M 输入 63 则将输出：63 是 3 的超级幸运数!

79.单选题 [NOIP 普及组 2016]

```
#include <iostream>
using namespace std;
int main() {
    int k = 4, n = 0;
    while (n < k) {
        n++;
        if (n % 3 != 0)
            continue;
        k--;
    }
    cout << k << "","" << n << endl;
    return O;
}
```

程序运行后的输出结果是（）。

A. 2,2

B. 2,3

C. 3,2

D. 3,3

80.单选题 [NOIP 提高组 2016]

```
#include <iostream>
using namespace std;
int main() {
    int k = 4, n = 0;
    while (n < k) {
        n++;
```

```
            if (n % 3 != 0) continue;
            k--;
        }
        cout << k << "","" << n << endl;
        return 0;
}
```

程序运行后的输出结果是（ ）。

A. 2,2

B. 2,3

C. 3,2

D. 3,3

81.填空题 [NOIP 普及组 2016]

```
#include <iostream>
using namespace std;
int main()
{
    int i=100,x=0,y=0;
    while (i>0)
    {
        i--;
        x=i%8;
        if (x==1) y++;
    }
    cout<<y<<endl;
    return 0;
}
```

输出：（ ）

82.单选题 [GESP202309【2 级】] 某班级人数不知，连续输入成绩直到输入负数停止，输入结束后求出平均成绩。在以下 C++代码横线处应填入是( )。

```
double totalScore=0;//总分
int studCount=0;//总人数
while(_____){//此处填写代码
    cin>>score;
    if(score<0)
        break;
```

```
    totalScore+=scorel
    studCount+=1;
}
cout<<""平均分=""<<totalScore/studCount;
```

A. true

B. false

C. True

D. False

83.单选题 [GESP202406【1 级】] 下面的 C++代码用于求 1~N 之间所有奇数之和，其中 N 为正整数，如果 N 为奇数，则求和时包括 N。有关描述错误的是（ ）。

```
int N;
cout << ""请输入正整数："";
cin >> N;
int i = 1, Sum = 0;
while (i <= N){
    if (i % 2 == 1)
        Sum += i;
    i += 1;
}
cout << i << "" "" << Sum;
```

A. 执行代码时如果输入 10，则最后一行输出将是 11 25

B. 执行代码时如果输入 5，则最后一行输出将是 6 9

C. 将 i += 1 移到 if (i % 2 == 1) 前一行，同样能实现题目要求

D. 删除 if (i % 2 == 1) ，并将 i += 1 改为 i += 2 ，同样可以实现题目要求

84.填空题 [NOIP 普及组 2016]

```
#include <iostream>
using namespace std;
int main()
{
    int max, min, sum, count = 0;
    int tmp;
    cin >> tmp;
    if (tmp==0) return 0;
```

```
    max = min = sum = tmp;
    count++;
    while (tmp != 0) {
        cin >> tmp;
        if (tmp != 0)
        {
            sum += tmp;
            count++;
            if (tmp > max) max = tmp;
            if (tmp < min) min = tmp;
        }
    }

    cout << max << "","" << min << "","" << sum / count << endl;
    return 0;
}
```

输入：1 2 3 4 5 6 0 7

输出：（ ）

85. [GESP202309【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int x=1;
while(x<100){
    if(!(x%3))
        cout<<x<<"","";
    else if(x/10)
        break;
    x+=2;
}
cout<<x;
```

A. 1

B. 3,9,11

C. 3,6,9,10

D. 1,5,7,11,13,15

86.单选题 [GESP202312【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int x=1;
while(x<100){
```

```
    if(x%3!=0)
        cout<<x<<"","";
    else if(x/10)
        break;
    else
        x+=5;
    x+=2;
}
cout<<x;
```

A. 1

B. 1,3

C. 15,17

D. 1,10,12

3. 特殊数判断

87.单选题 [GESP202312【2 级】] 下面C++代码用于判断输入的整数是否为对称数，如 1221、12321 是对称数，但 123、972 不是对称数。下面对该题对应代码的说法，正确的是（ ）。

```
cin>>n;
newnum=0;
while(n){
    newnum=newnum*10+n%10;
    n=n/10;
}
if(newnum=n)
    cout<<n<<""为对称数"";
```

A. 代码没有语法错误，如果 N 为对称数，第 8行将能正确输出。

B. 代码没有语法错误，但如果 N 为负数，将导致死循环。

C. 代码存在语法错误，程序不能被执行。

D. 代码没有语法错误，但不能达到预期目标，因为循环结束 N 总为 0。

88.单选题 [GESP202403【2 级】] 以下 C++代码判断一个正整数 N 的各个数位是否都是偶数。如果都是，则输出“是”，否则输出“否”。例如 N=2024 时输出“是”。则横线处应填入（ ）。

```
int N,Flag;
cin >> N;
Flag = true;
while(N != 0){
    if(N %2 != 0){
        Flag= false;
        _________
    }
    e1se
        N /= 10;
}
if(Flag == true)
    cout <<“是”;
e1se
    cout <<""否”;
```

A. break

B. continue

C. N = N / 10

D. N = N % 10

89.单选题 [GESP202403【2 级】] 一个数的所有数字倒序排列后这个数的大小保持不变，这个数就是回文数，比如 101 与 6886 都是回文数，而 100 不是回文数。以下程序代码用于判断一个数是否为回文数，横线处应填写？（ ）

```
int n,a,k;
cin >>n;
a=0;
k=n;
while(n>0){
    a=_________//在此处填写代码
    n /= 10;
}
if(a == k)
    cout<<""是回文数;
else
```

```
12      cout<<""不是回文数"";
```

A. 10 * a + n % 10

B. a + n % 10

C. 10 * a + n / 10

D. a + n / 10

90.单选题 [GESP202406【1 级】] 如果一个整数 N 能够表示为 XX 的形式，那么它就是一个完全平方数，下面 C++代码用于完成判断 N 是否为一个完全平方数，在横线处应填入的代码是（ ）。

```
int N;
cin >> N;
for(int i = 0; i <= N; i++)
    if(____________________)
        cout << N << ""是一个完全平方数\n"";
```

A. i == NN

B. i10 == N

C. i+i == N

D. ii == N

91.单选题 [GESP202312【2 级】] 以下 C++代码用于输出 1-100（含）的整数平方数（完全平方数），如 16 是 4 的平方，横线处应填写（ ）。

```
for(i=1;i<100;i++)
    if(_______)
        cout<<i<<""      "";
```

A. int(sqrt(i)) * int(sqrt(i)) = i

B. int(sqrt(i)) == sqrt(i)

C. int(sqrt(i)) * int(sqrt(i)) == i

D. int(sqrt(i)) = sqrt(i)

4. 循环嵌套

92.判断题 [GESP 样题【2 级】] C++语言中，循环不能写太多层，因为层数是有限制的。

93.单选题 [GESP202309【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int cnt=0;
for(int i=1;i<9;i++){
    for(int j=1;j<i;j+=2){
        cnt+=1;
    }
}
cout<<cnt;
```

A. 16

B. 28

C. 35

D. 36

94.单选题 [GESP202312【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int cnt=0;
for(int i=0;i<5;i++){
    for(int j=0;j<i;j++){
        cnt+=1;
    }
}
cout<<cnt;
```

A. 5

B. 10

C. 20

D. 30

95.单选题 [GESP202309【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int cnt=0;
for(int i=0;i<13;i+=3){
    for(int j=1;j<i;j+=2){
        if(i*j%2==0)
```

```
            break;
        else
            cnt+=1;
    }
}
cout<<cnt;
```

A. 1

B. 3

C. 15

D. 没有输出

96.单选题 [GESP202312【3 级】] 下面 C++代码执行后的输出是（ ）。

```
int temp=0;
for(int i=1;i<7;i++){
    for(int j=1;j<5;j++){
        if(i/j==2)
            temp++;
    }
}
cout<<temp<<endl;
```

A. 10

B. 8

C. 4

D. 3

97.判断题 [GESP202406【2 级】] 下面 C++代码执行后将输出 10。

```
int cnt = 0;
for (int i = 0; i < 10; i++)
    for (int j = 0; j < i; j++){
        cnt += 1;
        break;
    }
cout << cnt;
```

98.判断题 [GESP202406【2 级】] 下面 C++代码执行后，将输出 5。

```
int cnt = 0;
for (int i = 1; i < 5; i++)
    for (int j = i; j < 5; j +=i)
        if (i * j % 2 == 0)
            cnt += 1;
cout << cnt;
```

99.单选题 [GESP202406【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int loopCount = 0;
for (int i=0; i < 10; i++)
    for (int j=1; j < i; j++)
        loopCount += 1;
cout << loopCount;
```

A. 55

B. 45

C. 36

D. 28

100.单选题 [GESP202406【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int loopCount = 0;
for (int i=0; i < 10; i++){
    for (int j=0; j < i; j++)
        if (i * j % 2)
            break;
    loopCount += 1;
}
cout << loopCount;
```

A. 25

B. 16

C. 10

D. 9

101.单选题 [GESP202312【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int cnt=0;
for(i=0;i<10;i++){
    for(j=1;j<i;j+=2){
        if(i*j%2==0){
            cnt++;
            break;
        }
    }
}
if(i>=10)
    cout<<cnt<<""#"";
cout<<cnt;
```

A. 0

B. 8#8

C. 4

D. 4#4

102.单选题 [GESP202312【2 级】] 下面 C++代码执行后的输出是（ ）。

```
n=4
for(int i=0;i<n;i++){
    for(int j=1;j<i;j++){
        if(i*j%2==0){
            cout<<i<<""#"";
        }
    }
    continue;
}
cout<<""0"";
```

A. 2#3#0

B. 1#2#0

C. 1#0#

D. 2#3#

103.单选题 [GESP202403【2 级】] 给定两个整数 与 ，打印出一个栅栏图形，这个栅栏应该分成 段，段与段之间的间隔为 + ， 段内的填充为 k 个 - 。形如 n=5，k=6 时，图形如下：

+-----+-----+-----+-----+-----+

以下程序代码用于绘制该图形，横线处应填写？（ ）

```
int n,k,i,j;
n=5,k=6;
for(int i=0;i<n;i++){
    ______//在此处填写代码
    for(int j=1;j<k;j++){
        cout<<'+';
    }
}
cout<<""+"";
```

A. cout << '+' << endl;

B. cout << '+' << ' ' << endl;

C. cout << '+';

D. cout << '+' << ' ';

104.单选题 [GESP202309【2 级】] 如下图所示，输出 N 行 N 列的矩阵，对角线为 1，横线处应填入（ ）。

```
请输入行列数列：9
1 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 1
int n=0;
cout<<""请输入行列数列："";
cin>>n;
for(int i=1;i<n+1;i++){
    for(int j=1;j<n+1;j++){
        if(_____)//此处填写代码
            cout<<1<<"" "";
        else
            cout<<0<<"" "";
    }
    cout<<endl;
```

```
}
```

A. i = j

B. j != j

C. i >= j

D. i == j

105.单选题 [GESP202309【2 级】] 下面图形每一行从字母 A 开始，以 ABC 方式重复。行数为输入的整数。请在 C++代码段横线处填入合适代码（ ）。

```
请输入行列数量数：7
A
AB
ABC
ABCA
ABCAB
ABCABC
ABCABCA

int n=0;
cout<<""请输入行列数量："";
cin>>n;
for(int i=1;i<n+1;i++){
    for(int j=0;j<i;j++){
        cout<<_____//此处填写代码
    }
    cout<<endl;
}
```

A. 'A' + j / 3

B. (char)('A' + j / 3)

C. 'A' + j % 3

D. (char)('A' + j % 3)

106.单选题 [GESP202312【2 级】] 下面的 C++代码用于实现如下左图所示的效果，应在以下右图 C++代码中填入（ ）。

```
12
0
```

```
01
012
01234
012345
0123456
01234567
012345678
0123456789
01234567890
012345678901
cin>>n;
for(int i=0;i<n;i++){
    nownum=0;
    for(int j=0;j<i+1;j++){
        cout<<nownum<<"""";
        nownum+=1;
        if(nownum==10)
            nownum=0;
    }
}
```

A. 与第 8行下面填入一行： cout << nowNum;

B. 与第 2行下面填入一行： cout << endl;

C. 与第 7行下面填入一行： cout << nowNum;

D. 与第 9行下面填入一行： cout << endl;

107.单选题 [GESP202309【2 级】] 输入行数，约定 1<=lineCount<=9 ，输出以下图形。应在 C++代码横线处填入（ ）。

```
输入行数列：9
请输入行数量：9
                    1
                  1 2 1
                1 2 3 2 1
              1 2 3 4 3 2 1
            1 2 3 4 5 4 3 2 1
          1 2 3 4 5 6 5 4 3 2 1
        1 2 3 4 5 6 7 6 5 4 3 2 1
      1 2 3 4 5 6 7 8 7 6 5 4 3 2 1
1 2 3 4 5 6 7 8 9 8 7 6 5 4 3 2 1
int lineCount=0;
cout<<""请输入行数量："";
```

```
cin>>lineCount;
for(int i=0;i<lineCount;i++){
    for(int j=0;j<_______;j++){//此处填写代码
        cout<<"" "";
    }
    for(int j=1;j<+1;j++){
        cout<<j<<"" "";
    }
    for(int j=i+1;j>0;j--){
        cout<<j<<"" "";
    }
    cout<<endl;
}
```

A. (lineCount - i - 1) * 2

B. (lineCount - i) * 2

C. lineCount - i - 1

D. lineCount - i

108.单选题 [GESP202406【2 级】] 下面 C++代码用于实现如下图所示的效果，其有关说法正确的是（ ）。

```
1
2   4
3   6   9
4   8   12   16
5   10   15   20   25
for (int i = 1; i < 6; i++){ // L1
    for (int j = 1; j < i+1; j++) //L2
        cout << i*j << "" "";
    cout << endl;
}
```

A. 当前代码能实现预期效果，无需调整代码

B. 如果 cout << endl; 移到循环 L2 内部，则可实现预期效果

C. 如果 cout << endl; 移到循环 L1 外部，则可实现预期效果

D. 删除 cout << endl; 行，则可实现预期效果

答案

1-5 C C T T F

6-10F T F T F

11-15 F B F A T

16-20 F T F C F

21-25 F F T B A

26-30 A B D C C

31-35 C B B F F

36-40 F F C B C

41-45 T B A C D

46-50 C A B 165　T

51-55 D A AC　B 6

56-60 A C D B 6

61-65 133　B B A B

66-70 4 8 A D B

71-75 D A B C C

76-80 B D D D D

81-85 13 A C 6,1,3　B

86-90 D D A A D

91-95 C F A B B

96-100 C F T C C

101-105 D A C D D

106-108 D A A

GW 信奥 CSP 初赛习题集 2-7

C++初级语法——处理字符类型

1. ASCII 码概述

1.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] ASCII 码的含义是（ ）。

A. 二—十进制转换码

B. 美国信息交换标准代码

C. 数字的二进制编码

D. 计算机可处理字符的唯一编码

2.单选题 [NOIP 普及组 2009] 关于 ASCII，下面哪个说法是正确的：

A. ASCII 码就是键盘上所有键的唯一编码。

B. 一个 ASCII 码使用一个字节的内存空间就能够存放。

C. 最新扩展的 ASCII 编码方案包含了汉字和其他欧洲语言的编码。

D. ASCII 码是英国人主持制定并推广使用的。

3.判断题 [GESP202303【2 级】] 如果 a 是 int 类型的变量，而且值为 1，则表达式'a'的值为'1'。

4.单选题 [NOIP 普及组 2011] 字符 '0' 的 ASCII 码为 48,则字符 '9' 的 ASCII 码为（ ）。

A. 39

B. 57

C. 120

D. 视具体的计算机而定

5.单选题 [NOIP 提高组 2011] 字符 ”A”的 ASCII 码为十六进制 41 ，则字符 ”Z”的 ASCII 码为十六进制的 （ ）。

A. 66

B. 5A

C. 50

D. 视具体的计算机而定

6.单选题 [NOIP 普及组 2009/NOIP 提高组 2009] 已知大写字母 A 的 ASCII 编码为 65（10 进制），则大写字母 J 的 10 进制 ASCII 编码为：

A. 71

B. 72

C. 73

D. 以上都不是

7.单选题 [GESP202306【3 级】] 已知大写字符'A'的 ASCII 编码的十六进制表示为 0x41，则字符'F'的 ASCII 编码的十六进制表示为（ ）。

A. 46

B. 47

C. 48

D. 49

8.单选题 [GESP202309【3 级】] 已知大写字符 'A' 的 ASCII 编码的十六进制表示为 0x41 ，则字符 'L' 的 ASCII 编码的十六进制表示为（）。

A. 4A

B. 4B

C. 4C

D. 52

9.判断题 [GESP202309【3 级】] 字符常量'3'的值和 int 类型常量 3 的值是相同的只是占用的字节数不同

2. ASCII 码参与关系逻辑运算

10.单选题 [GESP202306【2 级】] 如果 a 为 char 类型的变量，下列哪个表达式可以正确判断“a 是数字”？（ ）

A. '0' <= a && a <= '9'

B. '1' <= a && a <= '0'

C. '0' <= a <= '9'

D. '1' <= a <= '0'

11.单选题 [GESP202303【2 级】] 如果 a 为 char 类型的变量，下列哪个表达式可以正确判断“a 是小写字母”？

A． a <= a <= z

B． a - 'a' <= 'z' - 'a'

C． 'a' <= a <= 'z'

D． a >= 'a' && a <= 'z

12.单选题 [GESP202309【3 级】] 如果 a 为 char 类型的变量，下列哪个表达式可以正确判断“a 是大写字母”？（ ）

A. a - 'A' <= 26

B. 'A' <= a <= 'Z'

C. 'A' <= 'a' <= 'Z'

D. ('A' <= a) && (a <= 'Z')

3. ASCII 码参与算术运算

13.单选题 [GESP202306【2 级】] 下列关于 C++语言的叙述，正确的是（ ）。

A. char 类型变量不能赋值给 int 类型的变量。

B. 两个 int 类型变量相乘，计算结果还是 int 类型。

C. 计算两个 int 类型变量相乘时，如果乘积超出了 int 类型的取值范围，程序会报错崩溃。

D. 计算两个 double 类型变量相除时，如果除数的值为 0.0，程序会报错崩溃。

14.判断题 [GESP202309【1 级】] C++表达式 ('1' + '1') 的值为 '2' 。

15.判断题 [GESP202306【2 级】] 如果 a 为 char 类型的变量，且取值为大写字母'F'，则执行语句 a = a + 1;后，a 的值会变为大写字母'G'。

16.判断题 [GESP202303【2 级】] 如果 a 为 char 类型的变量，且取值为小写字母，则执行语句 a = a - 'a' + 'A';后，a 的值会变为与原值对应的大写字母。

17.单选题 [GESP 样题【2 级】] 已知'A'的 ASCII 码为 65，'1'的 ASCII 码为 49，'r'的 ASCII 码为 114，则表达式'A' + 1 的计算结果为（ ）。

A. 66

B. 'B'

C. 114

D. 'r'

18.单选题 [GESP 样题【2 级】] 10. 如果 a 为 char 类型的变量，且 a 的值为'1'，则执行 a = a + 3;之后，a 的值会是（ ）。

A. 4

B. 13

C. '4'

D. '13'

19.单选题 [GESP202306【2 级】] 如果 a 为 char 类型的变量，且 a 的值为'C'（已知'C'的 ASCII 码为 67），则执行 cout << (a + 2);会输出（ ）。

A. E

B. C+2

C. C2

D. 69

20.单选题 [GESP202303【2 级】] 如果 a 为 char 类型的变量，且 a 的值为'2'，则下列哪条语句执行后，a 的值不会变为'3'？

A． a = a + 1;

B． a + 1;

C． a = 1 + a;

D． ++a;

21.单选题 [GESP 样题【2 级】] 12. 如果 a 为 int 类型的变量，且 a 的值为 1，则下列表达式的值为'3'的是（ ）。

A. a + 2

B. a + '2'

C. 'a' + 2

D. (char)(a + '2')

22.单选题 [GESP202312【1 级】] 定义变量 char c ，下面对 c 赋值的语句，不符合语法的是( )。

A. c = (char)66;

B. c = (char)(66);

C. c = char(66);

D. c = char 66;

23.单选题 [GESP202303【2 级】] 如果 a 和 b 都是 char 类型的变量，下列哪个语句不符合 C++语法？

A． b = a + 1;

B． b = a + '1';

C． b = 'a'++;

D． b = a++;

24.单选题 [GESP202306【2 级】] 如果 a 为 int 类型的变量，b 为 char 类型的变量，则下列哪个语句不符合 C++语法？（ ）

A. a = a + 1.0;

B. a = (int)(b - '0');

C. b = (char)(a + '0');

D. (int)b = a;

25.判断题 [GESP202312【2 级】] C++表达式 2*int('9')*2 的值为 36。

26.单选题 [GESP202303【1 级】] 表达式((3 == 0) + 'A' + 1 + 3.0)的结果类型为（）。

A. double

B. int

C. char

D. bool

27.单选题 [GESP202306【2 级】] 在下列代码的横线处填写（ ），使得输出是 9。

```
#include<iostream>
using namespace std;
int main(){
char a='3',b='6';
cout<<______;//在此处填入代码
return 0;
}
```

A. (a + b)

B. (a + b - '0')

C. (char)(a + b)

D. (char)(a + b - '0')

答案

1-5 B B F B B

6-10 D A C F A

11-15 D D B F T

16-20 T A C D B

21-25 D D C D F

26-27 A D

GW 信奥 CSP 初赛习题集 3-1

C++中级语法——数组

1. 一维数组

1. 数组的定义与初始化

1.判断题 [GESP202306【3 级】] 在 C++语言中，数组被定义时，它的大小就确定了。

2.判断题 [GESP202309【3 级】] 在 C++语言中，定义数组时， [] 中必须指定元素个数。

3.判断题 [GESP 样题【3 级】] C++语言中的数组可以根据需要自动调整大小。

4.单选题 [GESP 样题【3 级】] 若有以下代码，则数组 arr 的长度是( )

```
int arr[] = {1, 2, 3, 4, 5};
```

A. 3

B. 4

C. 5

D. 6

5.判断题 [GESP202406【3 级】] 数组的所有元素在内存中可以不连续存放。

6.判断题 [GESP202406【3 级】] C++中可以对数组和数组的每个基础类型的元素赋值。

7.单选题 [GESP202306【3 级】] 下列关于 C++语言中数组的叙述，不正确的是（ ）。

A. 数组必须先定义后使用。

B. 数组的所有元素在内存中是连续存放的。

C. 除了字符数组，在定义数组时“[]”内必须有常数。

D. 不能对数组赋值，但可以对数组的每个基础类型的元素赋值。

8.单选题 [GESP 样题【3 级】] 在 C++语言中，可以定义一个一维整型数组的是( )

A. int array[5];

B. int array[];

C. int[5] array;

D. int[] array;

9.单选题 [GESP202309【3 级】] 以下数组定义，符合 C++语言语法的是（ ）。

A. double a[];

B. double b[] = {1, 2.0, '3'};

C. double c[3.0];

D. double[] d = new double[3];

10.单选题 [GESP202306【3 级】] 以下数组定义，符合 C++语言语法的是（ ）。

A. int a[];

B. int b['3'];

C. int c[3.0];

D. int[3] d;

11.单选题 [GESP202306【3 级】] 一个数组定义为 double array[3];，则这个数组占用内存的大小为（ ）。

A. 24

B. 12

C. 6

D. 3

12.单选题 [GESP202309【3 级】] 如果数组定义为 long long array[] = {3, 5, 7, 2}; ，则数组 array 占用的字节数为（ ）。

A. 32

B. 16

C. 8

D. 4

2. 数组的简单使用

13.判断题 [GESP202309【3 级】] 在 C++语言中，数组下标的大小决定元素在逻辑上的先后顺序，与元素在内存中位置的先后顺序无关。

14.判断题 [GESP 样题【3 级】] 在 C++语言中，数组的下标从 1 开始计数。

15.判断题 [GESP 样题【3 级】] 在 C++语言中，可以使用浮点数（如 3.0）作为数组下标。

16.判断题 [GESP202306【3 级】] 在 C++语言中，可以使用字符（如'0'）作为数组下标。

17.判断题 [GESP 样题【4 级】] 已知数组 a 定义为 int a[100];，则赋值语句 a['0'] = 3;会导致编译错误。

18.判断题 [GESP202306【3 级】] 在 C++语言中，长度为 n 的数组，合理的下标范围是从 0 到 n，包括 0 和 n。

19.判断题 [GESP202309【3 级】] 在 C++语言中，长度为 n 的数组，访问下标为 n 的元素会引起编译错误。

20.单选题 [GESP202309【3 级】] 下列关于 C++语言中数组的叙述，不正确的是（ ）。

A. 可以定义 0 个元素的数组。

B. 不能定义 -1 个元素的数组。

C. 数组下标越界访问会产生编译错误。

D. 程序运行时发生数组下标的越界访问，程序依然可能正常结束。

21.单选题 [GESP202309【3 级】] 一个数组定义为 double array[3]; ，则可合理访问这个数组的元素的下标最大为（ ）。

A. 2

B. 3

C. 23

D. 24

22.判断题 [GESP202312【3 级】] 下面C++程序将输出 1。

```
int arr[10]={1};
cout<<arr[0]<<endl;
```

23.填空题 [NOIP 普及组 2004]

```
#include<stdio.h>
int main(){
    int u[4],a,b,c,x,y,z;
    scanf(""%d %d %d %d"",&(u[0]),&(u[1]),&(u[2]),&(u[3]));
    a=u[0]+u[1]+u[2]+u[3]-5;
    b=u[0]*(u[1]-u[2]/u[3]+8);
    c=u[0]*u[1]/u[2]*u[3];
    x=(a+b+2)*3-u[(c+3)%4];
    y=(c*100-13)/a/(u[b%3]*5);
    if((x+y)%2==0)z=(a+b+c+x+y)/2;
    z=(a+b+c-x-y)*2;
    printf(""%d\n"",x+y-z);
```

```
    return 0;
}
```

输入：2 5 7 4

输出：

24.填空题 [NOIP 提高组 2004]

```
#include<stdio.h>
int main(){
    int u[4],a,b,c,x,y,z;
    scanf(""%d %d %d %d"",&(u[0]),&(u[1]),&(u[2]),&(u[3]));
    a=u[0]+u[1]+u[2]+u[3]-5;
    b=u[0]*(u[1]-u[2]/u[3]+8);
    c=u[0]*u[1]/u[2]*u[3];
    x=(a+b+2)*3-u[(c+3)%4];
    y=(c*100-13)/a/(u[b%3]*5);
    if((x+y)%2==0)z=(a+b+c+x+y)/2;
    z=(a+b+c-x-y)*2;
    printf(""%d\n"",x+y-z);
    return 0;
}
```

输入：2 5 7 4

输出：

25.填空题 [NOIP 普及组 2006]

```
#include<stdio.h>
int main()
{
    int i,u[4],a,b,x,y=10;
    for(i=0;i<=3;i++)
    scanf(""%d"",&u[i]);
    a=(u[0]+u[1]+u[2]+u[3])/7;
    b=u[0]/((u[1]-u[2])/u[3]);
    x=(u[0]+a+2)-u[(u[3]+3)%4];
    if(x>10)
        y+=(b*100-u[3])/(u[u[0]%3]*5);
    else
        y+=20+(b*100-u[3])/(u[u[0]%3]*5);
    printf(""%d,%d\n"",x,y);
```

```
    return 0;
} /* 注：本例中，给定的输入数据可以避免分母为 0 或下标越界。 */
```

输入： 9 3 9 4

输出： _____

26.填空题 [NOIP 提高组 2006]

```
#include <stdio.h>
int main()
{
    int i,u[4],v[4],x,y=10;
    for(i=0;i<=3;i++)
        scanf(""%d"", &u[i]);
    v[0]=(u[0]+u[1]+u[2]+u[3])/7;
    v[1]=u[0]/((u[1]-u[2])/u[3]);
    v[2]=u[0]*u[1]/u[2]*u[3];
    v[3]=v[0]*v[1];
    x=(v[0]+v[1]+2)-u[(v[3]+3)%4];
    if(x>10) 由 OIFans.c 收集
        y+= (v[2]*100-v[3])/(u[u[0]%3]*5);
    else
        y+=20+(v[2]*100-v[3])/(u[v[0]%3]*5);
    printf(""%d,%d\n"", x,y);
    return 0;
} /* 注：本例中，给定的输入数据可以避免分母为 0 或下标越界。 */
```

输入：9 3 9 4

输出：_____

27.填空题 [NOIP 普及组 2007]

```
#include <stdio.h>
int main()
{
    int i,p[5],a,b,c,x,y=20;
    for(i=0;i<=4;i++) cin>>p[i];
    a=(p[0]+p[1])+(p[2]+p[3]+p[4])/7;
    b=p[0]+p[1]/((p[2]+p[3])/p[4]);
    c=p[0]*p[1]/p[2];
    x=a+b-p[(p[3]+3)%4];
    if(x>10)
```

```
            y+= (b*100-a)/(p[p[4]%3]*5);
        else
            y+=20+(b*100-c)/(p[p[4]%3]*5);
        cout<<x<<","<<y<<endl;
}
// 注：本例中，给定的输入数据可以避免分母为 0 或数组元素下标越界。
```

输入：6 6 5 5 3

输出：_____

28.填空题 [NOIP 提高组 2007]

```
#include <stdio.h>
int main()
{
    int i, p[5], q[5], x, y = 20;
    for ( i = 0; i <= 4; i++ )
        scanf( "%d", &p[i] );
    q[0] = (p[0] + p[1]) + (p[2] + p[3] + p[4]) / 7;
    q[1] = p[0] + p[1] / ( (p[2] + p[3]) / p[4]);
    q[2] = p[0] * p[1] / p[2];
    q[3] = q[0] * q[1];
    q[4] = q[1] + q[2] + q[3];
    x = (q[0] + q[4] + 2) - p[(q[3] + 3) % 4];
    if ( x > 10 )
        y += (q[1] * 100 - q[3]) / (p[p[4] % 3] * 5);
    else
        y += 20 + (q[2] * 100 - q[3]) / (p[p[4] % 3] * 5);
    printf( "%d,%d\n", x, y );
    return(0);
}
/*注：本例中，给定的输入数据可以避免分母为 0 或数组元素下标越界。*/
```

输入：6 6 5 5 3

输出：_____

29.填空题 [NOIP 普及组 2008]

```
#include<iostream>
using namespace std;
int main()
{
```

```
    int i, a, b, c, d, f[4];
    for(i = 0; i < 4; i++) cin >> f[i];
    a = f[0] + f[1] + f[2] + f[3];
    a = a / f[0];
    b = f[0] + f[2] + f[3];
    b = b / a;
    c = (b * f[1] + a) / f[2];
    d = f[(b / c ) % 4];
    if(f[(a + b + c + d) % 4] > f[2])
        cout << a + b<< endl;
    else
        cout << c + d << endl;
    return 0;
}
```

输入： 9 19 29 39

输出： _____

30.填空题 [NOIP 提高组 2008]

```
#include<iostream>
using namespace std;
int main()
{
    int i, a, b, c, d, f[4];
    for(i = 0; i < 4; i++) cin >> f[i];
    a = f[0] + f[1] + f[2] + f[3];
    a = a / f[0];
    b = f[0] + f[2] + f[3];
    b = b / a;
    c = (b * f[1] + a) / f[2];
    d = f[(b / c ) % 4];
    if(f[(a + b + c + d) % 4] > f[2])
        cout << a + b<< endl;
    else
        cout << c + d << endl;
    return 0;
}
```

输入： 9 19 29 39

输出： _____

31.填空题 [NOIP 普及组 2005/NOIP 提高组 2005]

```
#include<stdio.h>
int main(){
    int a,b,c,p,q,r[3];
    scanf(""%d%d%d"",&a,&b,&c);
    p=a/b/c;
    q=b-c+a+p;
    r[0]=a*p/q*q;
    r[1]=r[0]*(r[0]-300);
    if(3*q-p%3<=r[0]&&r[2]==r[2])
        r[1]=r[r[0]/p%2];
    else
        r[1]=q%p;
    printf(""%d\n"",r[0]-r[1]);
    return 0;
}
```

输入：100 7 3

输出：

3. 数组的简单遍历

32.单选题 [GESP202403【3 级】] 执行下面 C++代码后输出的第一个数是（ ）。

```
int main(){
    int a[20],i;
    for(i=0;i<20;i++)
        a[i]= i+1;
    for(;i>0;i--)
        cout<< a[i-1]<<""""";
    cout<< endl;
    return 0;
}
```

A. 20

B. 19

C. 1

D. 不确定

33.单选题 [GESP202403【3 级】] 下面 C++代码执行后数组中大于 0 的数的特征是（ ）。

```
int main(){
    int a[20],i;
    for(i=0;i<20;i++)
        a[i]= i+1;
    for(;i>0;i--)
        if((a[i]%2)&&(a[i]%3))
            a[i]=0;
    for(int i=0;i<20;i++){
        if(a[i])
            cout<<a[i]<<"" "";
    }
    cout<< endl;
    return 0;
}
```

A. 2 的倍数

B. 3 的倍数

C. 能被 2 或 3 整除的数

D. 能被 2 和 3 同时整除的数

34.单选题 [GESP202403【3 级】] 下面 C++程序执行的结果是（ ）。

```
int main(){
    int a[20],i;
    int cnt=0;
    for(i=0;i<20;i++)
        a[i]= i+1;
    for(;i>0;i--)
        if((a[i-1]+a[i-2])%3)
            cnt++;
    cout<<cnt<<endl;
    cout<< endl;
    return 0;
}
```

A. 5

B. 6

C. 10

D. 12

35.判断题 [GESP202312【3 级】] 执行C++代码，将输出 1 3 5 7 9 ， 9 之后还有一个空格。

```
int list[10]={1,2,3,4,5,6,7,8,9,10};
for(int i=0;i<10;i+=2){
    cout<<list[i]<<"" "";
}
```

36.单选题 [GESP202312【1 级】] 对下面的代码，描述正确的是（ ）。

```
#include<stdlib.h>
using namespace std;
int main(){
    int arr[]={2,6,3,5,4,8,1,0,9,10};
    for(int i=0;i<10;i++){
        cout<<arr[i]<<"" "";
    }
    cout<<i<<endl;
    cout<<endl;
}
```

A. 输出 2 6 3 5 4 8 1 0 9 10 10

B. 输出 2 6 3 5 4 8 1 0 9 9

C. 输出 2 6 3 5 4 8 1 0 9 10

D. 提示有编译错误

37.判断题 [GESP202312【3 级】] 执行C++代码将输出 0 5 ， 5 之后还有一个空格。

```
int list[10]={1,2,3,4,5,6,7,8,9,10};
for(int i=0;i<10;i++){
    if(i%5==0){
        cout<<list[i]<<"" "";
    }
}
```

38.单选题 [GESP202309【3 级】] 在下列代码的横线处填写（ ），可以使得输出是“120”。

```
#include <iostream>
using namespace std;
int main(){
    int array[5]={1,2,3,4,5};
    int res =0;
    for(int i=0;i<5;i++)
        _________;//在此处填入代码
    cout << res << endl;
    return 0;
}
```

A. res += array[i];

B. res *= array[i]

C. res = array[i]

D. 以上均不对。

39.单选题 [GESP 样题【4 级】] 在下列代码的横线处填写( ),可以使得输出是“21”。

```
#include <iostream>
using namespace std;
int main(){
    int a[5];
    a[0]= 1;
    for(int i= 1; i<5; i++)
        a[i]= a[i - 1]* 2;
    int sum = 0;
    for(int i = 0; i<5;_______)//在此处填入代码
        sum += a[i];
    cout<< sum<< endl;
    return 0;
}
```

A. i++

B. i += 2

C. i += 3

D. i |= 2

40.单选题 [GESP202406【3 级】] 小杨在做数学题，题目要求找出从 1 到 35 中能被 7 整除的数字，即[7, 14, 21, 28, 35]，则横线处应填写哪个代码？（ ）

```
#include <iostream>
using namespace std;
int main() {
    int arr[35];
    int count = 0;
    for (int i = 1; i <= 35; i++) {
        if (i % 7 == 0)
            _____ // 在此处填入代码
        }
    for (int i = 0; i < count; i++)
        cout << arr[i] << endl;
    return 0;
}
```

A. arr[count++] = i;

B. arr[i] = count++;

C. arr[i] = count;

D. arr[count] = count++;

41.单选题 [GESP202406【3 级】] 某小学男子篮球队招募新成员，要求加入球队的成员身高在 135 厘米以上（不含 135 厘米）。本次报名的人员有 10 人，他们的身高分别是 125、127、136、134、137、138、126、135、140、145。完善以下代码，求出本次球队能够招募到新成员的人数？（ ）

```
#include <iostream>
using namespace std;
int main() {
    int arr[10] = {125, 127, 136, 134, 137, 138, 126, 135, 140, 145};
    int count = 0;
    for(int i=0; i<10; i++)
        _____ // 在此处填入代码
    cout << count << endl;
    return 0;
}
```

A. count = arr[i]>135? 1: 0;

B. count += arr[i]>135? 1: 0;

C. count++;

D. 以上都不对

4. 数组的元素交换

42.填空题 [NOIP 普及组 2016/NOIP 提高组 2016]

```
#include <iostream>
using namespace std;
int main() {
	int a[6] = {1, 2, 3, 4, 5, 6};
	int pi = 0;
	int pj = 5;
	int t , i;
	while (pi < pj) {
		t = a[pi];
		a[pi] = a[pj];
		a[pj] = t;
		pi++;
		pj--;
	}
	for (i = 0; i < 6; i++)
		cout << a[i] << "","";
	cout << endl;
	return 0;

}
```

输出 ： ____

43.填空题 [NOIP 普及组 2013] （序列重排）全局数组变量 a 定义如下：

```
const int SIZE = 100;
int a[SIZE], n;
```

它记录着一个长度为 n 的序列 a [1], a[2], …,a[n] 。

现在需要一个函数，以整数 p (1 ⩽ p ⩽ n) 为参数，实现如下功能：将序列 a 的前 p 个数与后 n－p 个数对调，且不改变这 p 个数（或 n － p 个数）之间的相对位置。

例如，长度为 5 的序列 1, 2, 3, 4, 5 ，当 p = 2 时重排结果为 3, 4, 5, 1, 2 。

有一种朴素的算法可以实现这一需求，其时间复杂度为 O(n)、空间复杂度为 O(n)：

```
void swap1(int p){
    int i, j, b[SIZE];
    for (i = 1; i <= p; i++)
        b[ ① ] = a[i];
    for (i = p + 1; i <= n; i++)
        b[i - p] = ② ;
    for (i = 1; i <= ③ ; i++)
        a[i] = b[i];
}
```

我们也可以用时间换空间，使用时间复杂度为 O(n^2)、空间复杂度为 O(1) 的算法：

```
void swap2(int p){
    int i, j, temp;
    for (i = p + 1; i <= n; i++) {
        temp = a[i];
    for (j = i; j >= ④ ; j--)
        a[j] = a[j - 1];
        ⑤ = temp;
    }
}
```

44.填空题 [NOIP 提高组 2013]（序列重排）全局数组变量 a 定义如下：

```
const int SIZE=100;
int a[SIZE],n;
```

它记录着一个长度为 n 的序列 a[1]，a[2]，，， a[n]。现在需要一个函数，以整数 p(1 ≤p≤ n) 为参数， 实现如下功能： 将序列 a 的前 p 个数与后 n-p 个数对调， 且不改变这 p 个数（或 n-p 个数）之间的相对位置。例如，长度为 5 的序列 1，2，3，4，5，当 p=2 时重排结果为 3，4， 5，1，2。有一种朴素的算法可以实现这一需求，其时间复杂度为 O(n)、空间复杂度为 O(n)：

```
void swap1(int p){
int i, j, b[SIZE];
    for (i = 1;i <= p;i++)
        b[①] = a[i];
    for (i = p + 1;i <= n;i++)
        b[i - p] = a[i];
    for (i = 1;i <= n;i++)
        a[i] = b[i];
```

```
}
```

我们也可以用时间换空间，使用时间复杂度为 O(n^2)、空间复杂度为 O(1) 的算法：

```
void swap2(int p){
    int i, j, temp;
    for (i = p + 1;i <= n;i++) {
        temp = a[i];
        for (j = i;j >=②;j--)
            a[j] = a[j - 1];
        ③ = temp;
        }
}
```

事实上，还有一种更好的算法，时间复杂度为 O(n)、空间复杂度为 O(1)：

```
void swap3(int p){
    int start1, end1, start2, end2, i, j, temp;
    start1 = 1;
    end1 = p;
    start2 = p + 1;
    end2 = n;
    while (true) {
        i = start1;
        j = start2;
        while ((i <= end1) && (j <= end2)) {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
            i++;
            j++;
        }
        if (i <= end1)
            start1 = i;
        else if (④) {
            start1 =⑤;
            end1 =⑥;
            start2 = j;
        }else
            break;
    }
}
```

5. 数组的桶计数

45.单选题 [NOIP 普及组 2011/NOIP 提高组 2011]

```
#include <iostream>
#include <cstring>
using namespace std;
const int SIZE = 100;
int main(){
    int n, i, sum, x, a[SIZE];
    cin>>n;
    memset(a, 0, sizeof(a));
    for (i = 1; i <= n; i++) {
        cin>>x;
        a[x]++;
    }
    i = 0; sum = 0;
    while (sum < (n / 2 + 1)) {
        i++;
        sum += a[i];
    }
    cout<<i<<endl;
    return 0;
}
```

输入：

11

4 5 6 6 4 3 3 2 3 2 1

输出：_____

6. 数组的最值问题

46.判断题 [GESP202403【3 级】] 对定义的数组 int a[7]={2,0,2,4,3,1,6} ，可以用简单循环就找到其中最小的整数。

47.单选题 [CSP-S2022/CSP-J2021] 以比较为基本运算，在 n 个数的数组中找最大的数，在最坏情况下至少要做（ ）次运算。

A. n/2

B. n-1

C. n

D. n+1

48.单选题 [NOIP 提高组 2014/CSP-S2021] 同时查找 2n 个数中的最大值和最小值，最少比较次数为 ( ).

A.3(n-2)/2

B.4n-2

C.4n-2

D.2n-2

49.单选题 [NOIP 提高组 2016] 以比较作为基本运算 ，在 N 个数中找最小数的最少运算次数为 （ ）。

A. N

B. N-1

C. N^2

D. log N

50.单选题 [NOIP 普及组 2018] 给定一个含 N 个不相同数字的数组，在最坏情况下，找出其中最大或最小的数，至少需要 N－1 次比较操作。则最坏情况下，在该数组中同时找最大与最小的数至少需要（ ）次比较操作。（⌈ ⌉表示向上取整，⌊ ⌋表示向下取整）

A.⌈3N / 2⌉－2

B. ⌊3N / 2⌋ －2

C. 2N－2

D. 2N－4

51.单选题 [GESP202406【3 级】] 在下列代码的横线处填写( )，可以使得输出是“7”。

```
#include <iostream>
using namespace std;
int main() {
    int array[5] = {3，7，5，2，4};
    int max = 0;
    for(int i=0; i<5; i++)
        if(_____) // 在此处填写代码
            max = array[i];
    cout << max << endl;
    return 0;
}
```

A. max > array[i]

B. max < array[i]

C. max = array[i]

D. 以上均不对

52.单选题 [GESP202306【3 级】] 在下列代码的横线处填写（ ），可以使得输出是“2”。

```
#include<iostream>
using namespace std;
int main(){
    int array[5]={3,7,5,2,4};
    int min=0;
    for(int i=0;i<5;i++){
        if(_____)//在此处填写代码
            min=array[i];
    }
    cout<<min<<endl;
    return 0;
}
```

A. min > array[i]

B. min < array[i]

C. min = array[i]

D. 以上均不对

53.单选题 [NOIP 提高组 2014] 以下程序实现了找第二小元素的算法。输入时 n 个不等的数构成的数组 S，输出 S 中第 二小的数 SecondMin 。在最坏的情况下，该算法需要做 ( )次比较。

```
if (S[1] < S[2]) {
    FirstMin = S[1];
    SecondMin = S[2];
} else {
    FirstMin = S[2];
    SecondMin = S[1];
}
for (i = 3; i <= n; i++)
    if (S[i] < SecondMin)
        if(S[i] < FirstMin) {
            SecondMin = FirstMin;
            FirstMin = S[i];
        } else {
            Second = S[i];
        }
```

A.2n

B.n-1

C.2n-3

D.2n-2

54.填空题 [NOIP 提高组 2010]

```
#include<iostream>
using namespace std;
int main()
{
    const int SIZE=10;
    int data[SIZE],i,j,cnt,n,m;
    cin>>n>>m;
    for(i=1;i<=n;i++)
        cin>>data[i];
    for(i=1;i<=n;i++)
    {
        cnt=0;
        for(j=1;j<=n;j++)
            if( (data[i]<data[j]) || (data[j]==data[i] && j<i) )
                cnt++;
```

```
            if (cnt==m)
                cout<<data[i]<<endl;
        }
        return 0;
}
```

输入：

5 2

96 -8 0 16 87

输出：_____

7. 数组的多重遍历

55.单选题 [GESP202309【3 级】] 在下列代码的输出是（ ）。

```
#include <iostream>
using namespace std;
int main(){
    int array[10];
    for(int i=0;i< 10;i++)
        array[i]= i;
    for(int p=2;p<10;p++)
        if(array[p]== p)
            for(int n=p;n<10;n += p)
                array[n]= array[n]/p *(p-1);
    int res =0;
    for(int n=1;n<10;n++)
        res += array[n];
    cout << res << endl;
    return 0;
}
```

A. 15

B. 28

C. 45

D. 55

56.填空题 [NOIP 普及组 2006/NOIP 提高组 2006]

```
#include <stdio.h>
int main()
{
    int i,j,m[]={2,3,5,7,13};
    long t;
    for(i=0;i<=4;i++)
    {
        t=1;
        for(j=1;j<m[i];j++)
            t*=2;
        printf(""%ld "",(t*2-1)*t);
    }
    printf(""\n"");
}
```

输出：_____

2. 高维数组

57.单选题 [GESP202306【4 级】] 下列关于 C++语言中数组的叙述，不正确的是（ ）。

A. 一维数组在内存中一定是连续存放的。

B. 二维数组是一维数组的一维数组。

C. 二维数组中的每个一维数组在内存中都是连续存放的。

D. 二维数组在内存中可以不是连续存放的。

58.单选题 [GESP 样题【4 级】] 以下哪个选项能正确定义一个二维数组（ ）

A. int a[][];

B. char b[][4];

C. double c[3][];

D. bool d[3][4];

59.单选题 [GESP202403【8 级】] 以下二维数组的初始化，哪个是符合语法的？（ ）。

A. int a[][] = {{1, 2}, {3, 4}};

B. int a[][2] = {};

C. int a[2][2] = {{1, 2, 3}, {4, 5, 6}};

D. int a[2][] = {{1, 2, 3}, {4, 5, 6}};

60.单选题 [GESP 样题【4 级】] 如果有如下二维数组定义，则 a[0][3]的值为（ ）

int a[2][2] = {{0, 1}, {2, 3}};

A. 编译出错

B. 1

C. 3

D. 0

61.单选题 [GESP202306【4 级】] 一个二维数组定义为 double array[3][10];，则这个二维数组占用内存的大小为（ ）。

A. 30

B. 60

C. 120

D. 240

62.单选题 [GESP202309【4 级】] 一个二维数组定义为 char array[3][10]; ，则这个二维数组占用内存的大小为（ ）。

A. 10

B. 30

C. 32

D. 48

63.单选题 [GESP202306【4 级】] 一个二维数组定义为 int array[5][3];，则 array[1][2]和 array[2][1]在内存中的位置相差多少字节？（ ）

A. 2 字节。

B. 4 字节。

C. 8 字节。

D. 无法确定。

64.单选题 [GESP202309【4 级】] 一个三维数组定义为 long long array[6][6][6]; ，则 array[1][2][3] 和 array[3][2][1] 在内存中的位置相差多少字节？（ ）

A. 70 字节

B. 198 字节

C. 560 字节

D. 无法确定

65.单选题 [GESP 样题【4 级】] 以下哪个选项能正确访问二维数组 array 的元素（ ）

A. array[1, 2]

B. array(1)(2)

C. array[1][2]

D. array{1}{2}

66.判断题 [GESP202403【4 级】] 定义数组 int a[2024][3][16]={2,0,2,4,3,1,6} ，则 cout << a[2023][2][15] 的结果不确定。

67.单选题 [GESP202312【4 级】] 下面C++代码执行后的结果是( )。

```
int arr[3][3]={{1,2,3},{4,5,6},{7,8,9}};
for(int i=0;i<3;i++){
    for(int j=2;j>=0;j--){
        cout<<arr[i][j]<<"" "";
    }
    cout<<endl;
}
```

A.

1 2 3

4 5 6

7 8 9

B.

1 2 3 4 5 6 7 8 9

C.

3 2 1

6 5 4

9 8 7

D.

9 8 7 6 5 4 3 2 1

68.单选题 [GESP202306【4 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
#include<iostream>
using namespace std;
int main()
{
    int array[3][3];
    for(int i=0;i<3;i++){
        for(int j=0;j<3;j++){
            array[i][j]=i*10+j;
        }
    }
    int sum;
    for(int i=0;i<3;i++){
        sum+=array[i][i];
    }
    cout<<sum<<endl;
    return 0;
}
```

A. 3

B. 30

C. 33

D. 无法确定。

69.填空题 [NOIP 提高组 2008] （矩阵中的数字）有一个 n×n(1≤n≤5000)的矩阵 a，对于 1≤i<n,1≤j≤n，a(i,j)<a(i+1,j),a(j,i)<a(j,i+1)。即矩阵中左右相邻的两个元素，右边的元素一定比左边的大。上下相邻的两个元素，下面的元素一定比上面的大。给定

矩阵 a 中的一个数字 k，找出 k 所在的行列（注意：输入数据保证矩阵中的数各不相同）。

```
#include <iostream>
using namespace std;
int n,k,answerx,answery;
int a[5001][5001];
void FindKPosition()
{
    int i = n,j = n;
    while (j > 0)
    {
        if (a[n][j] < k) break;
        j --;
    }
            ①
    while (a[i][j] != k)
    {
        while (          ②          && i > 1) i --;
        while (          ③          && j <= n) j ++;
    }
                ④
                ⑤
}

int main()
{
    int i,j;
    cin >> n;
    for (i = 1;i <= n;i ++)
        for (j = 1;j <= n;j ++)
            cin >> a[i][j];
    cin >> k;
    FindKPosition();
    cout << answerx << "" "" << answery << endl;
    return 0;
}
```

70.填空题 [NOIP 普及组 2012] （坐标统计）输入 n 个整点在平面上的坐标。对于每个点，可以控制所有位于它左下方的点（即 x、y 坐标都比它小），它可以控制的点的数目称为“战斗力”。依次输出每个点的战斗力，最后输出战斗力最高的点的编号（如果若干个点的战斗力并列最高，输出其中最大的编号）。

```
#include<iostream>
using namespace std;
const int SIZE = 100;
int x[SIZE], y[SIZE], f[SIZE];
int n, i, j, max_f, ans;
int main(){
    cin>>n;
    for (i = 1; i <= n; i++)
        cin >> x[i] >> y[i];
    max_f = 0;
    for (i = 1; i <= n; i++){
        f[i] =①;
        for (j = 1; j <= n; j++){
            if (x[j] < x[i] && ②)
                ③;
        }
        if (④){
            max_f = f[i];
            ⑤;
        }
    }
    for (i = 1; i <= n; i++)
        cout<<f[i]<<endl;
    cout<<ans<<endl;
}
```

71.判断题 [GESP202406【4 级】] 在下面这个程序里， a[i][j] 和一个普通的整型变量一样使用。

```
#include<iostream>
using namespace std;
int main()
{
    int a[10][10]={0};
    for(int i=0;i<10;i++)
    {
        for(int j=0;j<10;j++)
        {
            if(i==j)
            {
                a[i][j]=1;
            }
        }
```

```
    }
}
```

答案

1.T

2.F

3.F

4.C

5.F

6.F

7.C

8.A

9.B

10.B

11.A

12.A

13.F

14.F

15.F

16.T

17.F

18.F

19.F

20.C

21.A

22.F

23.263

24.263

25.10,10

26. -13，57

27.15,46

28.129,43

29.23

30.23

31.-7452

32.A

33.C

34.D

35.T

36.D

37.F

38.D

39.B

40.A

41.B

42.6,5,4,3,2,1

43.

1. 正确答案： n - p + i
2. 正确答案： a[i]
3. 正确答案： n
4. 正确答案： i - p + 1
5. 正确答案： a[i - p]"

44.

1. 正确答案： n - p + i
2. 正确答案： i - p + 1
3. 正确答案： a[i - p]

4. 正确答案： j <= end2
5. 正确答案： i
6. 正确答案： j - 1"

45.3

46.T

47.B

48.A

49.B

50.A

51.B

52.D

53.C

54.16

55.B

56.6 28 496 8128 33550336

57.D

58.D

59.B

60.C

61.D

62.B

63.C

64.C

65.C

66.F

67.C

68.D

69.

1.正确答案： j++; / j+=1; / j=j+1;

2.正确答案： a[i][j] > k

3.正确答案： a[i][j] < k

4.正确答案： answerx = i;

5.正确答案： answery = j;"

70.

1. 正确答案： 0
2. 正确答案： y[j] < y[i]
3. 正确答案： f[i]++
4. 正确答案： f[i] >= max_f
5. 正确答案： ans = i"

71.T

GW 信奥 CSP 初赛习题集 3-2

C++中级语法——字符串

1. 字符串的定义与初始化

1.判断题 [GESP 样题【3 级】] 在 C++语言中，字符串是以'\0'结尾的字符数组。

2.判断题 [GESP202406【3 级】/GESP202306【3 级】/GESP202306【4 级】] 字符常量'\0'常用来表示字符串结束，和字符常量'0'相同。

3.单选题 [NOIP 普及组 2016] 以下关于字符串的判定语句中正确的是（）

A. 字符串是一种特殊的线性表

B. 串的长度必须大于零

C. 字符串不可以用数组来表示

D. 空格字符组成的串就是空串

4.判断题 [GESP202403【3 级】] 在 C++语言中，字符数组被定义时，它的大小可以调整。

5.单选题 [GESP202312【3 级】] 下面C++数组的定义中，会丢失数据的是( )。

A. char dict_key[] = {'p','t','o'};

B. int dict_value[] = {33,22,11};

C. char dict_name[]={'chen','wang','zhou'};

D. float dict_value[]={3,2,1};

6.单选题 [NOIP 普及组 2005] 在字符串“ababacbabcbdecced”中出现次数最多的字母出现了（ ）次。

A. 6

B. 5

C. 4

D. 3

E. 2

7.单选题 [GESP202403【3 级】] 定义字符数组 char str[20] = {'G', 'E', 'S', 'P'}; ，则 str 的字符串长度为（ ）。

A. 4

B. 5

C. 19

D. 20

8.单选题 [GESP202406【3 级】] 如果字符串定义为 char str[] = ""GESP""; ，则字符数组 str 的长度为( )。

A. 0

B. 4

C. 5

D. 6

9.单选题 [GESP202306【3 级】] 如果字符串定义为 char str[] = ""Hello"";则字符数组 str 的长度为（ ）。

A. 0

B. 5

C. 6

D. 7

2. 字符串的输入与输出

10.判断题 [GESP202312【3 级】] C++程序执行后，输入 chen a dai 输出应该为：chen 。

```
string str;
cin>>str;
cout<<str;
```

11.单选题 [GESP202312【3 级】] 下面C++代码执行后不能输出""GESP""的是( )。

A. string str(""GESP""); cout<<str<<endl;

B. string str=""GESP""; cout<<str<<endl;

C. string str(""GESP""); cout<<str[1]<<str[2]<<str[3]<<str[4]<<endl;

D. string str{""GESP""}; cout<<str<<endl;

12.单选题 [GESP202312【3 级】] 下面C++代码执行后的输出是（ ）。

```
char ch[10]={'1'};
cout<<ch[2]<<endl;
```

A. 0

B. 1

C. 输出空格

D. 什么也不输出""

13.单选题 [GESP202312【4 级】] 下面C++代码执行后，输出的是( )。

```
int arr[10]={1};
string strArr=""chen a dai"";
cout<<strArr[arr[11]]<<endl;
```

A. chen

B. c

C. chen a dai

D. dai

14.单选题 [GESP202312【3 级】] 执行下面C++代码后输出的是（ ）。

```
string str=(""chen"");
cout<<str[5]<<endl;
```

A. 输出未知的数

B. 输出'n'

C. 输出'\0'

D. 输出空格

15.单选题 [GESP202403【4 级】] 下面 C++代码执行后，输出的是( )

```
int main(){
    int x[]={2,0,2,4};
    char geSP[]=""Grade Examination of SP"";
    cout<<geSP[sizeof(x)]<< endl;
    cout << endl;
    return 0;
}
```

A. G

B. e

C. n

D. P

3. 字符串函数

16.单选题 [GESP 样题【3 级】] 下列哪个是 C++语言中用于获取字符串长度的函数( )

A. length()

B. len()

C. getLength()

D. strlen()

17.判断题 [GESP202312【4 级】] 在 C++中，两个字符串相加的运算符为+相当于字符串的合并运算。下面C++代码执行后，将输出 chenadai 。

```
string a=""chen"";
string b=""
c=""dai"";
string
string name=a+b+c;
cout<<name<<endl;
```

18.判断题 [GESP202312【3 级】] 执行下面C++代码后将输出""China""。

```
string a=""china"";
a.replace(0,1,""C"");
cout<<a<<endl;
```

19.单选题 [GESP202312【3 级】] 执行下面C++代码后输出的是（ ）。

```
string str=(""chen"");
int x=str.length();
cout<<x<<endl;
```

A. 4

B. 3

C. 2

D. 5

20.填空题 [NOIP 提高组 2015]

```
#include <iostream>
using namespace std;
int main()
{
    int len,maxlen;
    string s,ss;
    maxlen=0;
    do
    {
        cin>>ss;
        len=ss.length();
```

```
            if(ss[0]=='#')break;
            if(len>maxlen)
            {
                s=ss;
                maxlen=len;
            }
        }while(true);
        cout<<s<<endl;
        return 0;
}
```

输入：

I

am

a

citizen

of

China

#

输出：（ ）

21.判断题 [GESP202403【3 级】] 执行下面 C++代码后将输出 2 。

```
int main(){
    string str=""gEsP is Interesting"";
    int x= str.find(""s"");
    cout<<x<< endl;
    cout<<endl;
    return 0;
}
```

4. 字符串的简单遍历

22.单选题 [GESP202312【3 级】] 执行下面C++代码后，输出是（ ）。

```
string str=(""chen"");
```

```
int x=str.length();
int temp=0;
for(int i=0;i<=x;i++)
    temp++;
cout<<temp<<endl;
```

A. 4

B. 2

C. 5

D. 3

23.单选题 [GESP202312【4 级】] 下列 C++代码输入 1,2,3,4 ，执行后，将输出的是（ ）。

```
string str="""";
cin>>str;
int strlen=str.length();
for(int i=0;i<strlen;i++)
    if(str[i]<='9'&&str[i]>='0')
        cout<<str[i];
    else
        cout<<""#'
```

A. 1#4#

B. 1#3#

C. 1#2#3#4#

D. 1#2#3#4

24.单选题 [GESP 样题【3 级】] 在下列代码的横线处填写（），可以保证输出是“1357”，不会有多余字符

```
#include <iostream>
#include <string>
using namespace std;
int main(){
    char str[]=""1234567"";
    for(_________)//在此处填入代码
        cout << str[i];
    return 0;
```

```
}
```

A. int i = 0; i < strlen(str); i++

B. int i = 0; str[i] != '\0'; i++

C. int i = 1; i <= 7; i += 2

D. int i = 0; i <= 6; i += 2

25.填空题 [NOIP 普及组 2004]

```
int i, j;
char str1[]=""pig-is-stupid"";
char str2[]=""clever"";
str1[0]='d';
str1[1]='o';
for(i=7,j=0;j<6;i++,j++)
    str1[i]= str2[j];
printf(""%s\n"",str1);
```

输出：

26.填空题 [NOIP 普及组 2005]

```
#include<stdio.h>
int main(){
    char str[20]=""Today-is-terrible!"";
    int i;
    for(i=6;i<= 10;i++)
        if(str[i]=='-')str[i-1]='x';
    for(i=12;i>=0;i--)
        if(str[i]=='t')str[i+1]='e';
    printf(""s\n"",str);
    return 0;
}
```

输出：

27.单选题 [GESP202403【3 级】] 已知字符 '0' 的 ASCII 编码的十进制表示为 48，则执行下面 C++代码后，输出是（ ）。

```
string s=""316"";
```

```
int n=s.length();
int x=0;
for(int i=0;i<n; i++)
    x += s[i];
cout<< x<<endl;
cout<< endl;
```

A. 10

B. 58

C. 154

D. 316

28.填空题 [NOIP 普及组 2015]

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string str;
    int i;
    int count;
    count = 0;
    getline(cin,str);
    for (i = 0; i < str.length();i++)
    {
        if (str[i]>='a' && str[i] <= 'z')
            count++;
    }
    cout << ""It has "" << count << "" lowercases"" << endl;
    return 0;
}
```

输入：NOI2016 will be held in Mian Yang.

输出：

29.填空题 [NOIP 提高组 2013]

```
#include <iostream>
#include <string>
using namespace std;
```

```
int main() {
    string str;
    cin>>str;
    int n = str.size();
    bool isPlalindrome = true;
    for (int i = 0;i < n/2;i++) {
        if (str[i] != str[n-i-1]) isPlalindrome = false;
    }
    if (isPlalindrome)
        cout<<""Yes""<<endl;
    else
        cout<<""No""<<endl;
}
```

输入：

abceecba

输出：_____

30.填空题 [NOIP 普及组 2010]

```
#include<iostream>
#include<iostream>
using namespace std;
int main() {
    string s;
    char m1, m2;
    int i;
    getline(cin, s);
    m1 = ' ';
    m2 = ' ';
    for (i = 0; i < s.length(); i++)
        if (s[i] > m1) {
            m2 = m1;
            m1 = s[i];
        } else if (s[i] > m2)
            m2 = s[i];
    cout<<int(m1)<<' '<<int(m2)<<endl;
    return 0;
}
```

输入： Expo 2010 Shanghai China

输出： _____

31.填空题 [NOIP 普及组 2014] （数字删除）下面程序的功能是将字符串中的数字字符删除后输出。请填空。

```
#include <iostream>
using namespace std;
int delnum(char *s)
{
    int i, j;
    j = 0;
    for(i = 0; s[i] != '\0'; i++)
        if(s[i] < '0' ① s[i] > '9')
        {
            s[j] = s[i];
            ②;
        }
    return ③;
}
const int SIZE = 30;
int main()
{
    char s[SIZE];
    int len, i;
    cin.getline(s, sizeof(s));
    len = delnum(s);
    for(i = 0; i < len; i++)
        cout << ④;
    cout << endl;
    return 0;
}
```

5. 字符串的计数问题

32.单选题 [GESP202312【3 级】] 下面C++代码用于统计每种字符出现的次数，当输出为 3 时，横线上不能填入的代码是（ ）。

```
string str=""GEsp is a good programming test!"";
int x=0;
for(int i=0;i<str.length();i++)
    if(_____)
        x++;
```

```
cout<<x<<endl;
```

A. str[i]=='o'

B. str[i]=='a'+14

C. str[i]==115

D. str[i]==111

33.单选题 [GESP202312【4 级】] 在如下的 C++代码中实现了对字符串中出现的 26 个字母的个数统计，横线处应填入是（ ）。

```
string str=""HELLO CHEN A DAI"";
int strlen=str.length();
char alpha[26]={65};
int cnt[26]={0};
for(int i=1;i<26;i++)
    ___________;
for(int i=e;i<26;i++)
    cout<<alpha[i]<<"" "";
cout<<endl;
for(int i=0;i<26;i++)
    for(int j=0;j<strlen;j++)
        if(alpha[i]==str[j])
            cnt[i]++;
for(int i=0;i<26;i++)
    cout<<cnt[i]<<"" "";
```

A. alpha[i]=alpha[i-1]+1;

B. alpha[i]=alpha[i]+1;

C. alpha[i+1]=alpha[i]+1;

D. alpha[i-1]=alpha[i]+1;

34.填空题 [NOIP 普及组 2017]

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    int t[256];
```

```
    char s[10];
    int i;
    scanf(""%s"", s);
    for(i = 0; i < 256; i++) t[i] = 0;
    for(i = 0; i < strlen(s); i++) t[s[i]]++;
    for(i = 0; i < strlen(s); i++)
        if(t[s[i]] == 1) {
            cout << s[i] << endl;
            return 0;
        }
    cout << ""no"" << endl;
    return 0;
}
```

输入：

xyzxyw

输出：（ ）

35.单选题 [GESP202406【3 级】] 已知字符 '0' 的 ASCII 编码的十进制表示为 48，则执行下面 C++代码后，输出是( )。

```
#include <iostream>
using namespace std;
int main() {
string s = ""0629"";
int n = s.length();
int x = 0;
for(int i = 0; i < n; i++) x += s[i];
cout << x << endl;
return 0;
}
```

A. 17

B. 158

C. 209

D. 316

6. 字符串的 ASCII 码偏移

36.填空题 [NOIP 普及组 2018]

```
#include <cstdio>
char st[100];
int main() {
    scanf(""%s"", st);
    for (int i = 0; st[i]; ++i) {
        if ('A' <= st[i] && st[i] <= 'Z')
            st[i] += 1;
    }
    printf(""%s"", st);
    return 0;
}
```

输入：QuanGuoLianSai

输出：

37.单选题 [GESP202403【3 级】] 在下列代码的横线处填写（ ），可以使得输出是 GESP IS INTERESTING 。

```
string str=""gEsP is Interesting"";
int x=str.length();
for(int i=0;i<x;i++)
    if((str[i]>='a')&&(str[i]<='z'))
        ____________;
cout<<str<<endl;
cout<<endl;
```

A. str[i]+='a'-'A'

B. str[i]+=20

C. str[i]+='A'-'a'

D. 无法实现

38.填空题 [NOIP 普及组 2014]

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string st;
```

```
    int i, len;
    getline(cin, st);
    len = st.size();
    for(i = 0; i < len; i++)
        if(st[i] >= 'a' && st[i] <= 'z')
            st[i] = st[i] - 'a' + 'A';
    cout << st << endl;
    return 0;
}
```

输入：Hello, my name is Lostmonkey.

输出：____

39.填空题 [NOIP 普及组 2016]

```
#include <iostream>
using namespace std;
int main()
{
    int i,length1, length2;
    string s1, s2;
    s1 = ""I have a dream."";
    s2 = ""I Have A Dream."";
    length1 = s1.size();
    length2 = s2.size();
    for (i = 0; i < length1; i++)
        if (s1[i] >= 'a' && s1[i] <= 'z')
            s1[i] -= 'a'-'A';
    for (i = 0; i < length2; i++)
        if (s2[i] >= 'a' && s2[i] <= 'z')
            s2[i] -= 'a'-'A';
    if (s1 == s2) cout << ""="" << endl;
    else if (s1 > s2) cout << "">"" << endl;
    else cout << ""<"" << endl;

    return 0;
}
```

输出：（ ）

40.填空题 [NOIP 普及组 2011]

```
#include <iostream>
```

```
#include <string>
using namespace std;
int main(){
    string map = ""22233344455566677778889999"";
    string tel;
    int i;
    cin>>tel;
    for (i = 0; i < tel.length(); i++)
        if ((tel[i] >= '0') && (tel[i] <= '9'))
            cout<<tel[i];
        else if ((tel[i] >= 'A') && (tel[i] <= 'Z'))
            cout<<map[tel[i] - 'A'];
    cout<<endl;
    return 0;
}
```

输入： CCF-NOIP-2011

输出： _____

41.填空题 [NOIP 提高组 2008]

```
#include <iostream>
#include <cstring>
using namespace std;
int i,j,len;
char s[50];
int main()
{
cin >>s;
len = strlen(s);
for (i = 0;i < len; ++i)
{
if (s[i] >= 'A' && s[i] <= 'Z') s[i] -= 'A' - 'a';
}
for (i = 0;i < len; ++i)
{
if (s[i] < 'x') s[i] += 3; else s[i] += -23;
}
cout << s << '/';
for (j = 1;j < 4;j ++)
{
for (i = 0;i < len-j; i = i + j)
{
s[i] = s[i + j] ;
```

```
}
}
cout << s << endl;
return 0;
}
```

输入：ABCDEFGuvwxyz

输出：_____

42.填空题 [NOIP 普及组 2017]

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string ch;
    int a[200];
    int b[200];
    int n, i, t, res;
    cin >> ch;
    n = ch.length();
    for(i = 0; i < 200; i++) b[i] = 0;
    for(i = 1; i <= n ; i++) {
        a[i] = ch[i-1] - '0';
        b[i] = b[i-1] + a[i];
    }
    res = b[n];
    t = 0;
    for(i = n; i > 0; i--) {
        if(a[i] == 0) t++;
        if(b[i-1] + t < res) res = b[i-1] + t;
    }
    cout << res << endl;
    return 0;
}
```

输入：

10011010110011011010111110001

输出：（ ）

43.单选题 [CSP-J2020]

```cpp
#include <cstdlib>
#include <iostream>
using namespace std;

char encoder[26] = {'C','S','P',0};
char decoder[26];

string st;

int main()  {
   int k = 0;
   for (int i = 0; i < 26; ++i)
      if (encoder[i] != 0) ++k;
   for (char x ='A'; x <= 'Z'; ++x) {
      bool flag = true;
      for (int i = 0; i < 26; ++i)
         if (encoder[i] ==x) {
            flag = false;
            break;
         }
         if (flag) {
            encoder[k]= x;
            ++k;
         }
   }
   for (int i = 0; i < 26; ++i)
       decoder[encoder[i]- 'A'] = i + 'A';
   cin >> st;
   for (int i = 0; i < st.length( ); ++i)
      st[i] = decoder[st[i] -'A'];
   cout << st;
   return 0;
}
```

• 判断题

输入的字符串应当只由大写字母组成，否则在访问数组时可能越界。（ ）

若输入的字符串不是空串，则输入的字符串与输出的字符串一定不一样。( )

将第 12 行的 i < 26 改为 i < 16，程序运行结果不会改变。（ ）

将第 26 行的 i < 26 改为 i < 16，程序运行结果不会改变。（ ）

• 单选题

5） 若输出的字符串为 ABCABCABCA，则下列说法正确的是（ ）。

6）若输出的字符串为 CSPCSPCSPCSP，则下列说法正确的是（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 输入的字符串中既有 S 又有 P

B. 输入的字符串中既有 S 又有 B

C. 输入的字符串中既有 A 又有 P

D. 输入的字符串中既有 A 又有 B

6.

A. 输入的字符串中既有 P 又有 K

B. 输入的字符串中既有 J 又有 R

C. 输入的字符串中既有 J 又有 K

D. 输入的字符串中既有 P 又有 R

44.单选题 [CSP-J2019]

```
#include <cstdio>
#include <cstring>
using namespace std;
char st[100];
int main() {
```

```
6       scanf(""%s"", st);
7       int n = strlen(st);
8       for (int i = 1; i <= n; ++i) {
9           if (n % i == 0) {
10              char c = st[i - 1];
11              if (c >= 'a')
12                  st[i - 1] = c - 'a' + 'A';
13          }
14      }
15      printf(""%s"", st);
16      return 0;
17  }
```

判断题

输入的字符串只能由小写字母或大写字母组成。（ ）

若将第 8 行的 i = 1 改为 i = 0，程序运行时会发生错误。（ ）

若将第 8 行的 i <= n 改为 i * i <= n，程序运行结果不会改变。（ ）

若输入的字符串全部由大写字母组成，那么输出的字符串就跟输入的字符串一样。（ ）

选择题

若输入的字符串长度为 18，那么输入的字符串跟输出的字符串相比，至多有（ ）个字符不同。

若输入的字符串长度为（ ），那么输入的字符串跟输出的字符串相比，至多有 36 个字符不同。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 18

B. 6

C. 10

D. 1

6.

A. 36

B. 100000

C. 1

D. 128

答案

1.T

2.F

3.A

4.F

5.C

6.B

7.A

8.C

9.C

10.T

11.C

12.D

13.B

14.A

15.C

16.D

17.T

18.T

19.A

20.citizen

21.T

22.C

23.D

24.D

25.dog-is-clever

26.Today-ix-terrible!

27.C

28.It has 18 lowercases

29.Yes

30.120 112

31.

1. 正确答案： ||
2. 正确答案： j++
3. 正确答案： j
4. 正确答案： s[i]"

32.C

33.A

34.z

35.C

36.RuanHuoMianTai

37.C

38.HELLO,MY NAME IS LOSTMONKEY

39. =

40.22366472011

41. defghijxyzabc/hfizxjaybcccc

42.11

43.ABABAD

44.BABABB

GW 信奥 CSP 初赛习题集 3-3

C++中级语法——结构体与联合体

1.判断题 [GESP202309【4 级】] 在 C++语言中，可以通过定义结构体，定义一个新的数据类型。

2.判断题 [GESP202309【4 级】] 在 C++语言中，可以定义结构体类型的数组变量，定义结构体时也可以包含数组成员。

3.填空题 [NOIP 普及组 2015/NOIP 提高组 2015]

```
#include <iostream>
using namespace std;
struct point{
    int x;
    int y;
};
int main()
{
    struct EX{
        int a;
        int b;
        point c;
    }e;
    e.a = 1;
    e.b = 2;
    e.c.x = e.a + e.b;
    e.c.y = e.a * e.b;
    cout << e.c.x << ',' << e.c.y << endl;
    return 0;
}
```

输出：（ ）

4.单选题 [CSP-J2023] 阅读下述代码，请问修改 data 的 value 成员以存储 3.14，正确的方式是

```
union Data{
        int num;
        float value;
        char symbol;
};
union Data data;
```

A. data.value = 3.14;

B. value.data = 3.14;

C. data -> value = 3.14;

D. value->data = 3.14;

答案

1.T

2.T

3.3,2

4.A

GW 信奥 CSP 初赛习题集 3-4

C++中级语法——函数

1. 函数的基本概念

1.判断题 [GESP202303【2 级】] 在 C++语言中，一个程序不能有多个 main 函数。

2.判断题 [GESP202306【4 级】] 在 C++语言中，一个函数没有被调用时，它的参数不占用内存。

3.单选题 [GESP202306【4 级】] 下列关于 C++语言中函数的叙述，正确的是（ ）。

A. 函数必须有名字。

B. 函数必须有参数。

C. 函数必须有返回值。

D. 函数定义必须写在函数调用前。

4.单选题 [GESP202309【4 级】/GESP 样题【7 级】] 下列关于 C++语言中函数的叙述，正确的是（ ）。

A. 函数调用前必须定义。

B. 函数调用时必须提供足够的实际参数。

C. 函数定义前必须声明。

D. 函数声明只能写在函数调用前。

5.单选题 [GESP 样题【4 级】] 以下关于 C++函数的形参和实参的叙述，正确的是（ ）

A. 形参是实参的别名

B. 实参是形参的别名

C. 形参和实参是完全相同的

D. 形参用于函数声明，实参用于函数调用

6.单选题 [GESP202309【4 级】] 下列关于 C++语言中函数的叙述，不正确的是（ ）。

A. 两个函数的声明可以相同。

B. 两个函数的局部变量可以重名。

C. 两个函数的参数可以重名。

D. 两个函数可以重名。

7.判断题 [GESP202403【4 级】] 两个函数之间可以使用全局变量来传递数据。

8.单选题 [GESP202312【6 级】] 下面有关布尔类型的函数的说法，正确的是（ ）。

A. bool 类型函数只能返回 0 或者 1 两种值

B. bool 类型函数可以返回任何整数值

C. bool 类型函数必须有参数传递

D. bool 类型函数没有返回值

2. 自定义函数的编写

9. [GESP202406【4 级】] 下面函数不能正常执行的是（ ）

A.

```
#include<iostream>
using namespace std;
int func()
{
    //...
}

int main()
{
    //...
}
```

B.

```
#include<iostream>
using namespace std;
int main()
{
    fac();
}
int func()
{
    //...
}
```

C.

```
#include<iostream>
using namespace std;
int func()
{
    //...
}
int main()
{
    fac();
}
```

D.

```
#include<iostream>
using namespace std;
int  fac();
int main()
{
    fac();
}
int func()
{
    //...
}
```

10.单选题 [GESP 样题【5 级】] 有关下面 C++代码正确的是（ ）。

```
#include <iostream>
using namespace std;
void f1(){
    cout<<""f1()""<< endl;
}
```

```
void f1(int x){
    cout<<""f1(""<< x<<"")""<< endl;
}
int main(){
    f1();
    f1(0);
    f1('@');
    return 0;
}
```

A. 该程序不能正常运行，因为 f1 函数被重复定义。

B. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1()、f1()

C. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1(0)、f1(0)

D. 该程序可以正常运行，输出结果共 3 行，依次为 f1()、f1(0)、f1(48)

11.单选题 [GESP202406【4 级】] 下列代码中，输出结果是（ ）

```
#include<iostream>
using namespace std;

int func(int x,int y)
{
    int a=x,b=y;
    int t;
    t=a;
    a=b;
    b=t;
    cout<<a<<"" ""<<b<<"" "";
}

int main()
{
    int c,d;
    c=12;
    d=24;
    func(12,24);
    cout<<c<<"" ""<<d<<endl;
}
```

A. 12 24 24 12

B. 24 12 12 24

C. 12 12 24 24

D. 24 24 12 12

12.填空题 [NOIP 普及组 2010]

```
#include <iostream>
using namespace std;
int rSum(int j)
{
    int sum = 0;
    while (j != 0) {
        sum = sum * 10 + (j % 10);
        j = j / 10;
    }
    return sum;
}
int main()
{
    int n, m, i;

    cin>>n>>m;
    for (i = n; i < m; i++)
        if (i == rSum(i))
            cout<<i<<' ';
    return 0;
}
```

输入：90 120

输出：

13.填空题 [NOIP 提高组 2014]

```
#include <iostream>
using namespace std;
const int SIZE = 100;
int alive[SIZE];
int n;
int next(int num){
    do {
        num++;
        if (num > n)
            num = 1;
    } while(alive[num] == 0);
    return num;
```

```
}
int main() {
    int m, i, j, num;
    cin >> n >> m;
    for (i = 1; i <= n; i++)
        alive[i] = 1;
    num = 1;
    for (i = 1; i <= n; i++){
        for(j = 1; j < m; j++)
            num = next(num);
        cout << num << "" "";
        alive[num] = 0;
        if (i < n)
            num = next(num);
    }
    cout << endl;
    return 0;
}
```

输入：

11 3

输出：（ ）

14.填空题 [NOIP 提高组 2011]

```
#include<iostream>
#include<cstring>
#include<string>
using namespace std;
const int SIZE=10000;
const int LENGTH=10;
int n,m,a[SIZE][LENGTH];
int h(int u,int v)
{
    int ans,i;
    ans=0;
    for(i=1;i<=n;i++)
        if( a[u][i]!=a[v][i])
            ans++;
    return ans;
}
int main()
{
```

```
    int sum,i,j;
    cin>>n;
    memset(a,0,sizeof(a));
    m=1;
    while(1)
    {
        i=1;
        while( (i<=n) && (a[m][i]==1) )
            i++;
        if(i>n)
            break;
        m++;
        a[m][i]=1;
        for(j=i+1;j<=n;j++)
            a[m][j]=a[m-1][j];
    }
    sum=0;
    for(i=1;i<=m;i++)
        for(j=1;j<=m;j++)
            sum+=h(i,j);
    cout<<sum<<endl;
    return 0;
}
```

输入：7

输出：_____

15.填空题 [NOIP 提高组 2018]

```
#include <iostream>
using namespace std;
string s;
long long magic(int l, int r) {
    long long ans = 0;
        for (int i = l; i <= r; ++i) {
        ans = ans * 4 + s[i] - ‘a’ + 1;
    }
return ans;
}
int main() {
    cin >> s;
    int len = s.length();
    int ans = 0;
    for (int l1 = 0; l1 < len; ++l1) {
```

```
            for (int r1 = l1; r1 < len; ++r1) {
                bool bo = true;
                for (int l2 = 0; l2 < len; ++l2) {
                    for (int r2 = l2; r2 < len; ++r2) {
                        if (magic(l1, r1) == magic(l2, r2)
                          && (l1 != l2 || r1 != r2))
                          bo = false;
                    }
                }
                if (bo) {
                    ans += 1;
                }
            }
        }
        cout << ans << endl;
        return 0;
}
```

输入：abacaba

输出：

16.填空题 [NOIP 提高组 2004]

```
#include <stdio.h>
char c[3][200];
int s[10],m,n;
void numara(){
    int i, j, cod, nr;
    for(j=0;j<n; j++){
        nr =0;cod =1;
        for(i=0;i<m; i++){
            if(c[i][j]== '1'){
                if(!cod){
                    cod=1;s[nr]++;nr =0;
                }
            }
            else{
                if(cod){
                    nr=1;cod =0;
                }
                else nr++;
            }
        }
        if(!cod)
```

```
                s[nr]++;
        }
}
int main(){
    int i, j;
    scanf(""%d %d\n"",&m, &n);
    for(i=0;i<m; i++)gets(c[i]);
    numara();
    for(i=1;i<=m; i++)
        if(s[i]!=0)
            printf(""%d %d "",i,s[i]);
    return 0;
}
```

输入：

3 10

1110000111

1100001111

1000000011

输出：

17.填空题 [NOIP 普及组 2008] （字符串替换）给定一个字符串 S（S 仅包含大小写字母），下面的程序将 S 中的每个字母用规定的字母替换，并输出 S 经过替换后的结果。程序的输入是两个字符串，第一个字符串是给定的字符串 S，第二个字符串 S′ 由 26 个字母组成，它是 a~z 的任一排列，大小写不定，S′ 规定了每个字母对应的替换字母：S′ 中的第一个字母是字母 A 和 a 的替换字母，即 S 中的 A 用该字母的大写替换，S 中的 a 用该字母的小写替换；S′ 中的第二个字母是字母 B 和 b 的替换字母，即 S 中的 B 用该字母的大写替换，S 中的 b 用该字母的小写替换；……以此类推。

```
#include <iostream>
#include <string.h>
char change[26], str[5000];
using namespace std;

void CheckChangeRule()
{
    int i;
    for (i = 0;i < 26;i ++)
```

```
    {
        if ( [                    ①                    ] )
            change[i] -= 'A' - 'a';
    }
}

void ChangeString()
{
    int i;
    for (i = 0;i <strlen(str);i ++)
    {
        if (  [                    ②                    ]  )
            str[i] = change[str[i] - 'A'] -'a' + 'A';
        else
            [                ③                ]
    }
}

int main()
{
  int i;
cin >> str ;
    cin >> change;
    CheckChangeRule();
    [                ④                ]
    cout << str << endl;
    return 0;
}
```

18.填空题 [NOIP 普及组 2004]

```
#include <stdio.h>
char c[3][200];
int s[10],m,n;
void numara(){
    int i, j, cod, nr;
    for(j=0;j<n; j++){
        nr =0;cod =1;
        for(i=0;i<m; i++){
            if(c[i][j]== '1'){
                if(!cod){
                    cod=1;s[nr]++;nr =0;
                }
            }
```

```
                else{
                    if(cod){
                        nr=1;cod =0;
                    }
                    else nr++;
                }
            }
            if(!cod)
                s[nr]++;
        }
}
int main(){
        int i, j;
        scanf(""%d %d\n"",&m, &n);
        for(i=0;i<m; i++)gets(c[i]);
        numara();
        for(i=1;i<=m; i++)
            if(s[i]!=0)
                printf(""%d %d "",i,s[i]);
        return 0;
}
```

输入：

3 10

1110000111

1100001111

1000000011

输出：

3. 系统自带的数学函数

19.判断题 [GESP202406【7 级】] 使用 math.h 或 cmath 头文件中的对数函数，表达式 log(128) 的结果类型为 double 、值约为 7.0 。

20.判断题 [GESP202306【2 级】] 在使用 C++语言编写程序时，不能使用 sqrt、abs 等数学函数，包含<cmath>或<math.h>头文件后就能够使用了。

21.判断题 [GESP 样题【2 级】] 表达式 sqrt(9)的计算结果为 3，且结果类型为 int 类型。

22.判断题 [GESP202306【2 级】] 表达式 sqrt(9.0)的计算结果为 3，且结果类型为 int。

23.判断题 [GESP 样题【7 级】] pow(1,2)返回的结果是浮点数。

24.判断题 [GESP 样题【7 级】] 定义变量 double x=exp(-1)，则 x<0 为真。

25.判断题 [GESP202403【2 级】] 已知 A 的 ASCII 码值为 65，表达式 int('C')+abs(-5.8) 的值为 72.8。

26.判断题 [GESP202406【8 级】] 已知 int 类型的变量 a 和 b 中分别存储着一个直角三角形的两条直角边的长度，则斜边的长度可以通过表达式 sqrt(a * a + b * b) 求得。

27.判断题 [GESP202403【2 级】] 如果变量 a 的值使得 C++表达式 sqrt(a)==abs(a) ，则 a 的值为 0。

28.单选题 [GESP202403【2 级】] 下列 4 个表达式中，答案不是整数 8 的是？（ ）

A. abs(-8)

B. min(max(8, 9), 10)

C. int(8.88)

D. sqrt(64)

29.单选题 [CSP-S2021]

```
#include <iostream>
```

```cpp
#include <cmath>
using namespace std;

const double r = acos(0.5);

int a1, b1, c1, d1;
int a2, b2, c2, d2;

inline int sq(const int x) { return x * x; }
inline int cu(const int x) { return x * x * x; }

int main()
{
    cout.flags(ios::fixed);
    cout.precision(4);

    cin >> a1 >> b1 >> c1 >> d1;
    cin >> a2 >> b2 >> c2 >> d2;

    int t = sq(a1 - a2) + sq(b1 - b2) + sq(c1 - c2);

    if (t <= sq(d2 - d1)) cout << cu(min(d1, d2)) * r * 4;
    else if (t == sq(d2 + d1)) cout << 0;
    else {
        double x = d1 - (sq(d1) - sq(d2) + t) / sqrt(t) / 2;
        double y = d2 - (sq(d2) - sq(d1) + t) / sqrt(t) / 2;
        cout << (x * x * (3 * d1 - x) + y * y * (3 * d2 - y)) * r;
    }

    cout << endl;
    return 0;
}
```

假设输入的所有数的绝对值都不超过 1000，完成下面的判断题和单选题：

判断题

将第 21 行中 t 的类型声明从 int 改为 double， 不会 影响程序运行的结果。( )

将第 26、27 行中的 / sqrt(t) / 2 替换为/ 2 / sqrt(t)，不会影响程序运行的结果。( )

将第 28 行中的 x * x 改成 sq(x)、y * y 改成 sq(y)，不会影响程序运行的结果。( )

当输入为 0 0 0 1 1 0 0 1 时，输出为 1.3090 ( )

单选题

当输入为 1 1 1 1 1 1 1 2 时，输出为（ ）。

A. 3.1416

B. 6.2832

C. 4.7124

D. 4.1888

（2.5 分）这段代码的含义为（ ）。

A. 求圆的面积并

B. 求球的体积并

C. 求球的体积交

D. 求椭球的体积并

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 3.1416

B. 6.2832

C. 4.7124

D. 4.1888

6.

A. 求圆的面积并

B. 求球的体积并

C. 求球的体积交

D. 求椭球的体积并

30.填空题 [NOIP 普及组 2004] 题目描述:

给出三角形三边的边长，求此三角形内切圆（如下图所示，三角形的内切圆是和三角形三边都相切的圆）的面积。

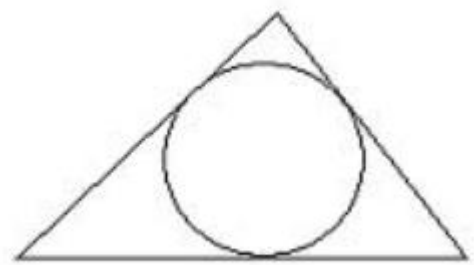
输入:三个正实数 a、b、c（满足 a+b>c，b+c>a，c+a>b），表示三角形三边的边长。

输出:三角形内切圆的面积，结果四舍五入到小数点后面 2 位。

输入样例:3 4 5

输出样例:3.14

```
#include <math.h>
int main(){
float a, b, c, r, s, t;
scanf(""%f %f %f"", &a, &b, &c);
s = (  ①  ) / 2;
t =    ②   (s * (s - a) * (s - b) * (s - c));
r = t / s;
printf(""  ③  \n"", 3.1415927 * r *    ④   );
return 0;
}
```



31.单选题 [CSP-J2023]

```
#include<iostream>
#include<cmath>
using namespace std;

double f(double a,double b,double c){
      double s=(a+b+c)/2;
      return sqrt(s*(s-a)*(s-b)*(s-c));
}
int main(){
    cout.flags(ios::fixed);
    cout.precision(4);

    int a,b,c;
    cin>>a>>b>>c;
    cout<<f(a,b,c)<<endl;
    return 0;
}
```

假设输入的所有数都为不超过的正整数，完成下面的判断题和单选题：

判断题

（2 分）当输入为 2 2 2 时，输出为 1.7321（ ）

（2 分）将第 7 行中的 (s-b)*(s-c) 改为 (s-c)*(s-b) 不会影响程序运行的结果（ ）

（2 分）程序总是输出四位小数（ ）

单选题

当输入为 3 4 5 时，输出为（ ）

当输入为 5 12 13 时，输出为（ ）

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 6.0000

B. 12.0000

C. 24.0000

D. 30.0000

5.

A. 24.0000

B. 30.0000

C. 60.0000

D. 120.0000

答案

1.T

2.T

3.A

4.B

5.D

6.A

7.T

8.A

9.B

10.D

11.B

12.99 101 111

13.3 6 9 1 5 10 4 11 8 2 7

14.57344

15.16

16.1 4 2 1 3 3

17.

正确答案： change[i] >= 'A' && change[i] <= 'Z' / change[i] <= 'Z'

正确答案： str[i] >= 'A' && str[i] <= 'Z' / str[i] <= 'Z'

正确答案： str[i] = change[str[i] - 'a'];

正确答案： ChangeString();"

18.1 4 2 1 3 3

19.F

20.T

21.F

22.F

23.T

24.F

25.T

26.T

27.F

28.B

29.ABBADC

30.

①、a+b+c

②、sqrt

③、%.2f

④、r

31.AAAAB

GW 信奥 CSP 初赛习题集 3-5

C++中级语法——递归与递推

1. 递归

1. 递归的概念

1.判断题 [GESP202406【4 级】] 函数不可以调用自己。

2.单选题 [NOIP 普及组 2013/NOIP 提高组 2013/NOIP 普及组 2018] 下面的故事与（ ）算法有着异曲同工之妙。 从前有座山，山里有座庙，庙里有个老和尚在给小和尚讲故事：“从前有座山，山 里有座庙，庙里有个老和尚在给小和尚讲故事：‘从前有座山，山里有座庙，庙里有个 老和尚给小和尚讲故事 .... ’

A. 枚举

B. 递归

C. 贪心

D. 分治

3.单选题 [CSP-J2022] 以下对递归方法的描述中，正确的是：（ ）。

A. 递归是允许使用多组参数调用函数的编程技术

B. 递归是通过调用自身来求解问题的编程技术

C. 递归是面向对象和数据而不是功能和逻辑的编程语言模型

D. 递归是将用某种高级语言转换为机器代码的编程技术

4.单选题 [GESP202403【5 级】] 递归函数在调用自身时，必须满足（ ），以避免无限递归？

A. 有终止条件

B. 函数参数递减（或递增）

C. 函数返回值固定

D. 以上都对

5.单选题 [NOIP 普及组 2008] 递归过程或函数调用时，处理参数和返回地址，通常使用一种称为（ ）的数据结构。

A. 队列

B. 多维数组

C. 线性表

D. 栈

6.判断题 [GESP202403【5 级】] 在 C 语言中，递归的实现方式通常会占用更多的栈空间，可能导致栈溢出。

7.单选题 [NOIP 普及组 2012/NOIP 提高组 2012/CSP-S2021] 在程序运行过程中，如果递归调用的层数过多，会因为（ ）引发错误。

A. 系统分配的栈空间溢出

B. 系统分配的堆空间溢出

C. 系统分配的队列空间溢出

D. 系统分配的链表空间溢出

8.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] 近 20 年来，许多计算机专家都大力推崇递归算法，认为它是解决较复杂问题的强有力的工具。在下列关于递归算法的说法中，正确的是（ ）。

A. 在 1977 年前后形成标准的计算机高级语言“FORTRAN77”禁止在程序使用递归，原因之一是该方法可能会占用更多的内存空间

B. 和非递归算法相比，解决同一个问题，递归算法一般运行得更快一些

C. 对于较复杂的问题，用递归方式编程一般比非递归方式更难一些

D. 对于已经定义好的标准数学函数 sin(x)，应用程序中的语句“y=sin(sin(x));”就是一种递归调用

2. 递归函数的编写

9.单选题 [GESP202406【5 级】] 给定如下函数：

```
int fun(int n) {
    if (n == 1) return 1;
    if (n == 2) return 2;
    return fun(n - 2) - fun(n - 1);
}
```

则当 时，函数返回值为（ ）。

A. 0

B. 1

C. 21

D. -11

10.单选题 [GESP202406【5 级】] 给定如下函数（函数功能同上题，增加输出打印）：

```
int fun(int n) {
    cout << n << "" "";
    if (n == 1) return 1;
    if (n == 2) return 2;
    return fun(n - 2) - fun(n - 1);
}
```

则当 时，屏幕上输出序列为（ ）。

A. 4 3 2 1

B. 1 2 3 4

C. 4 2 3 1 2

D. 4 2 3 2 1

11.单选题 [CSP-S2022] ack 函数在输入参数“(2,2)”时的返回值为（）。

```
unsigned ack(unsigned m, unsigned n) {
    if (m == 0) return n + 1;
```

```
    if (n == 0) return ack(m - 1, 1);
    return ack(m - 1, ack(m, n - 1));
}
```

A. 5

B. 7

C. 9

D. 13

12.填空题 [NOIP 普及组 2014]

```
#include <iostream>
using namespace std;
int fun(int n)
{
    if(n == 1)
        return 1;
    if(n == 2)
        return 2;
    return fun(n - 2) - fun(n - 1);
}
int main()
{
    int n;
    cin >> n;
    cout << fun(n) << endl;
    return 0;
}
```

输入：7

输出：____

13.单选题 [GESP202403【5 级】] 下面的 C++ 代码片段用于计算阶乘。请在横线处填入（ ），实现正确的阶乘计算。

```
int factorial(int n){
    if(n==0||n==1){
        return 1;
    }
    else{
        ________//在此处填入代码
```

```
    }
}
```

A. return n * factorial(n - 1);

B. return factorial(n - 1) / n;

C. return n * factorial(n);

D. return factorial(n / 2) * factorial(n / 2);

14.单选题 [GESP 样题【5 级】] 有关下面 C++代码说法错误的是（ ）。

```
#include <iostream>
using namespace std;
int factA(int n){
    if(n<= 1)
        return 1;
    int ret = 1;
    for(int i=2;i <= n; ++i)
        ret *= i;
    return ret;
}
int factB(int n){
    return n==1?1:n*factB(n-1);
}
int main(){
    int n;
    cin >> n;
    cout<<factA(n)<<' '<<factB(n)<< endl;
    return 0;
}
```

A. factA()采用循环方式求 n 的阶乘，factB()采用递归方式求 n 的阶乘

B. 程序执行时如果输入 5，能实现求其阶乘，程序输出结果为 120 120

C. 任何递归程序都可以使用循环改写

D. 程序执行时如果输入 100，不能正确求出 100 的阶乘

15.单选题 [GESP202309【5 级】] 有关下面 C++代码说法错误的是（ ）。

```
//sumA()和 sumB()用于求从 1 到 N 之和
#include <iostream>
using namespace std;
```

```
int sumA(int n){
    int sum = 0;
    for(int i=1;i<n+1;i++)
        sum += i;
    return sum;
}
int sumB(int n){
    if(n == 1)
        return 1;
    else
        return n+sumB(n-1);
}
int main(){
    int n= 0;
    cin >> n;
    cout << sumA(n)<<sumB(n)<< endl;
    return 0;
}
```

A. sumA() 用循环方式求从 1 到 N 之和， sumB() 用递归方式求从 1 到 N 之和。

B. 默认情况下，如果输入正整数 1000 ，能实现求从 1 到 1000 之和。

C. 默认情况下，如果输入正整数 100000 ，能实现求从 1 到 100000 之和。

D. 一般说来， sumA() 的效率高于 sumB() 。

16.单选题 [CSP-J2021] 考虑如下递归算法，则调用 solve(7) 得到的返回结果为（ ）。

```
solve(n)
    if n<=l return 1
    else if n>=5 return n*solve(n-2)
    else return n*solve(n-1)
```

A. 105

B. 840

C. 210

D. 420

17.单选题 [CSP-S2021] 有如下递归代码

```
solve(t, n):
    if t=1 return 1
    else return 5*solve(t-1,n) mod n
```

则 solve(23,23) 的结果为（ ）。

A. 1

B. 7

C. 12

D. 22

18.单选题 [GESP202403【7 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int down(int n){
    if(n<= 1)
        return n;
    return down(n-1)+down(n-2)+ down(n-3);
}
int main(){
    cout<<down(6)<< endl;
    return 0;
}
```

A. 6

B. 13

C. 20

D. 无法正常结束。

19.单选题 [GESP202309【5 级】] 下面代码执行后的输出是（ ）。

```
#include <iostream>
using namespace std;
int jumpFloor(int N){
    cout << N <<""#"";
    if(N==1||N== 2)
        return N;
    else
        return jumpFloor(N-1)+ jumpFloor(N-2);
```

```
}
int main(){
    cout<< jumpFloor(4)<<endl;
    return 0;
}
```

A. 4#3#2#2#4

B. 4#3#2#2#1#5

C. 4#3#2#1#2#4

D. 4#3#2#1#2#5

20.填空题 [NOIP 普及组 2018]

```
#include <iostream>
using namespace std;
int n, m;
int findans(int n, int m) {
    if (n == 0) return m;
    if (m == 0) return n % 3;
    return findans(n - 1, m) - findans(n, m - 1) + findans(n - 1, m - 1);
}
int main(){
    cin >> n >> m;
    cout << findans(n, m) << endl;
    return 0;
}
```

输入：5 6

输出：

21.单选题 [CSP-S2021] 有如下递归代码

```
solve(t, n)
    if t = 1 return 1
    else return 5 * solve(t - 1, n) mod n
```

则 solve(23, 23)的结果为（ ）。

A. 1

B. 7

C. 12

D. 22

22.填空题 [NOIP 普及组 2010]

```
#include <iostream>
using namespace std;

const int NUM = 5;

int r(int n)
{
    int i;
    if (n <= NUM)
        return n;
    for (i = 1; i <= NUM; i++)
        if (r(n - i) < 0)
            return i;
    return -1;
}

int main()
{
    int n;

    cin>>n;
    cout<<r(n)<<endl;
    return 0;
}
```

（1）

输入：7

输出：_____（4 分）

（2）

输入：16

输出：_____（4 分）

23.填空题 [NOIP 提高组 2010]

```
#include <iostream>
```

```
using namespace std;

const int NUM = 5;

int r(int n)
{
    int i;
    if (n <= NUM)
        return n;
    for (i = 1; i <= NUM; i++)
        if (r(n - i) < 0)
            return i;
    return -1;
}

int main()
{
    int n;

    cin>>n;
    cout<<r(n)<<endl;
    return 0;
}
```

输入：16

输出：_____

24.填空题 [NOIP 提高组 2005]

```
#include <stdio.h>
long g(long k){
    if(k<= 1)return k;
    return(2002*g(k-1)+2003*g(k-2))%2005;
}
int main(){
    long n;
    scanf(""%ld"",&n);
    printf(""%ld\n"",g(n));
    return 0;
}
```

输入：2005

输出：

25.填空题 [NOIP 普及组 2008]

```
#include<iostream>
using namespace std;
void foo(int a, int b, int c)
{
    if(a > b)
        foo(c, a, b);
    else
        cout<<a<<','<<b<<','<<c<<endl;
}
int main()
{
    int a, b, c;
    cin >> a >> b >> c;
    foo(a, b, c);
    return 0;
}
```

输入: 3 1 2

输出:

26.填空题 [NOIP 提高组 2008]

```
#include<iostream>
using namespace std;
void foo(int a, int b, int c)
{
    if(a > b)
        foo(c, a, b);
    else
        cout<<a<<','<<b<<','<<c<<endl;
}
int main()
{
    int a, b, c;
    cin >> a >> b >> c;
    foo(a, b, c);
    return 0;
}
```

输入: 2 1 3

输出：

27.填空题 [NOIP 提高组 2008]

```
#include<iostream>
using namespace std;
void f(int a, int b, int c)
{
    cout << a << b << c << ‘/’;
    if(a == 3 && b == 2 && c == 1)
        return;
    if(b < c)
        f(a, c, b);
    else if(a < b)
    {
        if(a < c)
        f(c, a, b);
    else
        f(b, c, a);
    }
}
int main()
{
    int a, b, c;
    cin >> a >> b >> c;
    f(a, b, c);
    cout << endl;
    return 0;
}
```

输入: 1 3 2

输出： _______

28.填空题 [NOIP 提高组 2011]

```
#include<iostream>
using namespace std;
int n;
void f2(int x,int y);
void f1(int x,int y)
{
     if(x<n)
          f2(y,x+y);
```

```
}
void f2(int x,int y)
{
      cout<<x<<' ';
      f1(y,x+y);
}
int main()
{
      cin>>n;
      f1(0,1);
      return 0;
}
```

输入：30

输出：_____

29.填空题 [NOIP 普及组 2006/NOIP 提高组 2006]

```
#include<iostream.h>
#include<iomanip.h>
void digit(long n,long m){
    if(m>0)
        cout <<setw(2)<<n%10;
    if(m>1)
        digit(n/10,m/10);
    cout <<setw(2)<<n%10;
}
void main(){
    long x,x2;
    cout <<""Input a number:""<<endl;
    cin >>x;
    x2=1;
    while(x2<x)x2*=10;
    x2/=10;
    digit(x,x2);
    cout <<endl;
}
```

输入：9734526

输出：

30. [NOIP 普及组 2011]

```
#include<iostream>
using namespace std;
int solve(int n,int m)
{
    int i,sum;
    if(m==1) return 1;
    sum=0;
    for(i=1;i<n;i++)
        sum+= solve(i,m-1);
    return sum;
}
int main()
{
    int n,m;
    cin>>n>>m;
    cout<<solve(n,m)<<endl;
    return 0;
}
```

输入：7 4

输出：

31.填空题 [NOIP 普及组 2017]

```
#include <stdio.h>
int g(int m, int n, int x) {
    int ans=0;
    int i;
    if(n == 1)
        return 1;
    for(i = x; i <= m / n; i++)
        ans += g(m - i, n - 1, i);
    return ans;
}
int main()
{
    int t, m, n;
    scanf(""%d%d"", &m, &n);
    printf(""%d"", g(m, n, 0));
    return  0;
}
```

输入：7 3

输出：（ ）

32.填空题 [NOIP 提高组 2017]

```
#include <iostream>
using namespace std;
int g(int m, int n, int x)
{
    int ans = 0;
    int i;
    if (n == 1)
        return 1;
    for (i = x; i <= m / n; i++)
        ans += g(m - i, n - 1, i);
    return ans;
}
int main()
{
    int t, m, n;
    cin >> m >> n;
    cout << g(m, n, 0) << endl;
    return 0;
}
```

输入：8 4

输出：_____

33.填空题 [NOIP 提高组 2014]

```
#include <iostream>
using namespace std;

int fun(int n, int minNum, int maxNum) {
    int tot, i;
    if(n == 0)
        return 1;
    tot = 0;
    for (i = minNum; i <= maxNum; i++){
        tot += fun(n - 1, i + 1, maxNum);
    }
    return tot;
}

```

```
int main(){
    int n,m;
    cin >> n >> m;
    cout << fun(m, 1, n) << endl;
    return 0;
}
```

输入：6 3

输出：（ ）

34.单选题 [GESP202312【5 级】] 阅读下面的 C++代码，执行后其输出是( )。

```
int stepCount=0;
int fracA(int N){
    stepCount +=1;
    cout<< stepCount << ""->"";
    int rtn=1;
    for(int i=1;i <= N; i++)
        rtn*= i;
    return rtn;
}
int fracB(int N){
    stepCount += 1;
    cout << stepCount <<""->"";
    if(N == 1)
        return 1;
    return N*fracB(N-1);
}
int main(){
    cout<<fracA(5);
    cout<<""<===>"";
    cout<< fracB(5);
    return 0;
}
```

A. 1->120<===>2->120

B. 1->120<===>1->120

C. 1->120<===>1->2->3->4->5->120

D. 1->120<===>2->3->4->5->6->120

35.填空题 [NOIP 提高组 2004]

```
#include <stdio.h>
int number,ndata,data[100],sum;
void solve(int s,int sign, int n){
    int i;
    for(i=s;i<ndata;i++){
        sum += sign*(number /n/data[i]);
        solve(i+1,-sign,n*data[i]);
    }
}
int main(){
    int i;
    scanf(""%d %d"",&number, &ndata);
    sum =0;
    for(i=0;i<ndata;i++)scanf(""%d"",&(data[i]));
    solve(0,1,1);
    printf(""%d\n"",sum);
    return 0;
}
```

输入：1000 3 5 13 11

输出：

36.单选题 [GESP202309【5 级】] 印度古老的汉诺塔传说：创世时有三根金刚柱，其中一柱从下往上按照大小顺序摞着 64 片黄金圆盘，当圆盘逐一从一柱借助另外一柱全部移动到另外一柱时，宇宙毁灭。移动规则：在小圆盘上不能放大圆盘，在三根柱子之间一次只能移动一个圆盘。下面的 C++代码以递归方式实现汉诺塔，横线处应填入代码是（ ）。

```
#include <iostream>
using namespace std;
//递归实现汉诺塔，将 N 个圆盘从 A 通过 B 移动 C
//圆盘从底到顶，半径必须从大到小
void Hanoi(string A, string B, string C,int N){
    if(N == 1)
        cout << A<<""->""<<C<< endl;
    else{
        Hanoi(A,C,B,N-1);
        cout << A << "" ->""<<C<< endl;
        _________;//此处填写代码
    }
}
```

```
int main(){
    Hanoi(""甲"",""乙"",""丙"",3);
    return 0;
}
```

A. Hanoi(B, C, A, N - 2)

B. Hanoi(B, A, C, N - 1)

C. Hanoi(A, B, C, N - 2)

D. Hanoi(C, B, A, N - 1)

37.填空题 [NOIP 提高组 2015]

```
#include <iostream>
using namespace std;
int fun(int n,int fromPos,int toPos)
{
    int t,tot;
    if(n==0)return 0;
    for(t=1;t<=3;t++)
        if(t!=fromPos && t != toPos)break;
    tot=0;
    tot+=fun(n-1,fromPos,t);
    tot++;
    tot+=fun(n-1,t,toPos);
    return tot;
}
int main()
{
    int n;
    cin>>n;
    cout<<fun(n,1,3)<<endl;
    return 0;
}
```

输入：5

输出：（ ）

38.单选题 [GESP 样题【5 级】] 下面 C++代码意在实现字符串反序的功能。关于这段代码，以下说法正确的是（ ）

```
#include <iostream>
```

```
#include <cstring>
using namespace std;
void printsReverse(char* sln,int len){
    if(len<= 1){
        cout<< sln[0];
    }else{
        cout<< sln[0];
        printsReverse(sln+1,len-1);
    }
}
int main(){
    char sln[100]=""Hello"";
    printsReverse(sln,strlen(sln));
    return 0;
}
```

A. 这段代码可以正确实现字符串反序的功能，其输出为`olleH`

B. 这段代码不能正确实现字符串反序的功能，其输出为`Hello`

C. 这段代码不能正确实现字符串反序的功能，其输出为`HHHHH`

D. 这段代码不能正确实现字符串反序的功能，其输出为`ooooo`

39.单选题 [GESP202309【5 级】] 下面 C++代码以递归方式实现字符串反序，横线处应填上代码是（ ）。

```
//字符串反序
#include <iostream>
#include <string>
using namespace std;
string sReverse(string sln){
    if(sln.length()<= 1)
        return sln;
    else
        return _____//此处填写代码
}
int main(){
    string sln;
    cin >>sln;
    cout<<sReverse(sln)<< endl;
    return 0;
}
```

A. sln[sln.length() - 1] + sReverse(sln.substr(0, sln.length() - 1));

B. sln[0] + sReverse(sln.substr(1, sln.length() - 1));

C. sReverse(sln.substr(0, sln.length() - 1)) + sln[sln.length() - 1];

D. sReverse(sln.substr(1, sln.length() - 1)) + sln[sln.length() - 1];

40.填空题 [NOIP 提高组 2016]

```
#include<iostream>
using namespace std;

int lps(string seq, int i, int j) {
    int len1, len2;
    if (i == j)
        return 1;
    if (i > j)
        return 0;
    if (seq[i] == seq[j])
        return lps(seq, i + 1, j - 1) + 2;
    len1 = lps(seq, i, j - 1);
    len2 = lps(seq, i + 1, j);
    if (len1 > len2)
        return len1;
    return len2;
}

int main() {
    string seq = ""acmerandacm"";
    int n = seq.size();
    cout << lps(seq, 0, n - 1) << endl;
    return 0;
}
```

输出 ： ____

2. 递推

41.判断题 [GESP 样题【4 级】] 递推算法通常有初始值。

42.单选题 [GESP202406【4 级】] 下面关于递推的说法不正确的是（ ）。

A. 递推表现为自己调用自己

B. 递推是从简单问题出发，一步步的向前发展，最终求得问题。是正向的

C. 递推中，问题的 n 要求是在计算中确定，不要求计算前就知道 n

D. 斐波那契数列可以用递推实现求解

3. 斐波那契问题

43.判断题 [GESP202306【4 级】] 数列 1, 1, 2, 3, 5, 8 ... 是以意大利数学家列昂纳多・斐波那契命名的数列，从第三个数开始，每个数是前面两项之和。如果计算该数列的第 n 项（其中 n>3）fib(n)，我们采用如下方法：① 令 fib(1)=fib(2)=1 ②用循环 for i=3 to n 分别计算 f(i) ③输出 fib(n)。这体现了递推的编程思想。

44.单选题 [GESP 样题【4 级】] 以下哪个递推关系式表示斐波那契数列（ ）

A. F(n) = F(n-1) + F(n-2) + F(n-3)

B. F(n) = F(n-1) + F(n-2)

C. F(n) = F(n-1) * F(n-2)

D. F(n) = F(n-1) / F(n-2)

45.单选题 [GESP202403【4 级】] 下面 C++函数中采用的算法是（ ）

```
int fib(int n){
    int i, f[n]={0,1};
    for(int i=2; i<=n; i++)
        f[i]=f[i-1]+f[i-2];
    return f[n];
}
```

A. 递推

B. 递归

C. 迭代

D. 循环

46.单选题 [GESP202309【4 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
#include <iostream>
```

```
using namespace std;
int main(){
    int fib[10];
    fib[0]= 0;
    fib[1]= 1;
    for(int i=2;i<10;i++)
        fib[i]=fib[i-1]+fib[i- 2];
    cout<< fib[10]<< endl;
    return 0;
}
```

A. 0

B. 5

C. 55

D. 无法确定。

47.单选题 [GESP 样题【7 级】] 下面定义的函数用来求斐波那契数列的 F(n)，其中 ，描述正确的是（ ）。

```
int fab(int n)
{
    int f[n+1];
    f[0]=0,f[1]=1;
    for (int i=2;i<=n; i++)
        f[i]= f[i-1]+ f[i-2];
    return f[n];
}
```

A. f[0]和 f[1]是递归终止条件

B. 数组 f 保存算法执行过程的状态

C. 使用倍增法来求解

D. 算法不能正常结束

48.判断题 [GESP202312【5 级】] 以下 C++代码能以递归方式实现斐波那契数列，该数列第 1、2 项为 1，以后各项均是前两项之和。

```
int Fibo(int N){
    if(N==1|N== 2)
        return 1;
```

```
    else{
        int m=fiboA(N-1);
        int n=fiboB(N-2);
        return m+n;
    }
}
```

49.单选题 [GESP202406【5 级】] 下面 C++代码用于求斐波那契数列，该数列第 1、2 项为 1，以后各项均是前两项之和。函数 fibo()属于( )。

```
int fibo(int n) {
    if (n <= 0)
        return 0;
    if (n == 1 || n == 2)
        return 1;
    int a = 1 , b = 1, next;
    for (int i = 3; i <= n; i++) {
        next = a + b;
        a = b;
        b = next;
    }
    return next;
}
```

A. 枚举算法

B. 贪心算法

C. 迭代算法

D. 递归算法

50.单选题 [GESP202406【7 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int fib(int n) {
    if (n <= 1)
        return n;
    return fib(n - 1) + fib(n - 2);
}
int main() {
    cout << fib(6) << endl;
    return 0;
```

```
}
```

A. 5

B. 8

C. 13

D. 无法正常结束。

51.单选题 [GESP 样题【5 级】] 下面代码执行后的输出是（ ）。

```
#include <iostream>
using namespace std;
int fibonacci(int N){
    cout<<N<<","";
    if(N==1||N==2){
        return 1;
    } else{
        return fibonacci(N-1)+fibonacci(N-2);
    }
}
int main(){
    cout<<fibonacci(5)<< endl;
    return 0;
}
```

A. 5,4,3,2,1,2,1,5

B. 5,4,3,2,1,2,3,2,1,5

C. 5,4,4,3,2,1,3,2,1,5

D. 5,4,3,2,1,3,2,1,5

52.单选题 [GESP202312【5 级】] 下面 C++代码用于求斐波那契数列，该数列第 1、2 项为 1，以后各项均是前两项之和。下面有关说法错误的是( )。

```
int fiboA(int N){
    if(N==1||N==2)
        return 1;
    return fiboA(N-1)+fiboA(N-2);
}
int fiboB(int N){
    if(N==1||N == 2)
```

```
        return 1;
    int last2=1,last1=1;
    int nowVal=0;
    for(int i=2;i<N;i++){
        nowVal = last1 + last2;
        last2 = last1;
        last1 = nowVal;
    }
    return nowVal;
}
```

A. fiboA( ) 用递归方式， fiboB() 循环方式

B. fiboA( ) 更加符合斐波那契数列的数学定义，直观易于理解，而 fiboB() 需要将数学定义转换为计算机程序实现

C. fiboA( ) 不仅仅更加符合数学定义，直观易于理解，且因代码量较少执行效率更高

D. fiboB( ) 虽然代码量有所增加，但其执行效率更高

53.单选题 [GESP202312【6 级】] 下面的 fiboA() 和 fiboB() 两个函数分别实现斐波那契数列，该数列第 1、第 2 项值为 1，其余各项分别为前两项之和。下面有关说法错误的是（ ）。

```
int fiboA(int n)
{
    if(n==0)
        return 1;
    if(n==1)
        return 1;
    else
        return fiboA(n-1)+fiboA(n-2);
}
int fiboB(int n){
    if((n==8)||(n==1)){
        fiboB[n]=n;
        return n;
    }
    else{
        if(fiboB[n]== 8){
            fiboB[n]=FiboB(n-1)+FiboB(n-2);
        }
        return fiboB[n];
    }
```

```
}
```

A. fiboA() 采用递归方式实现斐波那契数列

B. fiboB() 采用动态规划算法实现斐波那契数列

C. 当 N 值较大时， fiboA() 存在大量重复计算

D. 由于 fiboA() 代码较短，其执行效率较高

54.单选题 [NOIP 提高组 2013] 斐波那契数列的定义如下： F1=1，F2=1，Fn=Fn-1+Fn-2（n≥3）。如果用下面的函数计算斐波那契数列的第 n 项，则其时间复杂度为（ ）。

```
int F(int n){
    if (n <= 2)
        return 1;
    else
        return F(n - 1) + F(n - 2);
}
```

A. O(1)

B. O(n)

C. $O(n^2)$

D. O(Fn)

55.单选题 [CSP-S2021] 斐波那契数列的定义为：F1=1,F2=1,Fn=Fn-1+Fn-2(n>=3)。现在用如下程序来计算斐波那契数列的第 n 项，其时间复杂度为（ ）。

```
F(n):
    if n <= 2 return 1
    else return F(n - 1) + F(n - 2)
```

A. O(n)

B. $O(n^2)$

C. $O(2^n)$

D. O(nlogn)

56.单选题 [GESP202403【5 级】] 下面的代码片段用于计算斐波那契数列。该代码的时间复杂度是（ ）？

```
int fibonacci(int n){
  if(n<= 1){
      return n;
  else {
      return fibonacci(n-1)+fibonacci(n-2);
}
```

A.O(1)

B.O(n)

C.O(2^n)

D.O(logn)

57.单选题 [GESP202403【8 级】] 下面程序的时间复杂度为（ ）。

```
int fibonacci(int n){
  if(n<= 1){
    return n;
  else {
   return fibonacci(n-1)+fibonacci(n-2);
}
```

A.O(2^n)

B.O(Φ^n)，其中$\Phi=(\sqrt{5}+1)/2$

C.O(n)

D.O(1)

答案

1.F

2.B

3.B

4.A

5.D

6.T

7.A

8.A

9.D

10.C

11.B

12.-11

13.A

14.C

15.C

16.C

17.A

18.A

19.D

20.8

21.A

22."1. 1

2. 4"

23.4

24.31

25.2,3,1

26.1 3 2

27.132/213/231/312/321/

28.1 2 5 13 34

29.6 2 5 4 3 7 9 9 7 3 4 5 2 6

30.20

31.8

32.15

33.20

34.D

35.328

36.B

37.31

38.A

39.A

40.5

41.T

42.A

43.T

44.B

45.A

46.D

47.B

48.F

49.C

50.B

51.B

52.C

53.D

54.D

55.C

56.C

57.B

GW 信奥 CSP 初赛习题集 4

C++高级语法——指针

1. 指针的基本概念

1.单选题 [CSP-J2020] 在内存储器中每个存储单元都被赋予一个唯一的序号，称为（）。

A. 地址

B. 序号

C. 下标

D. 编号

2.判断题 [GESP 样题【4 级】] C++语言中的指针变量可以指向任何类型的数据。

3.判断题 [GESP202309【4 级】] 在 C++语言中，指针变量在逻辑上指向另一个变量在内存中的位置，指针变量本身不占用内存。

4.判断题 [GESP 样题【5 级】] 在 C++语言中，可以使用 delete 来释放指针指向的内存空间。

5.单选题 [GESP202306【4 级】] 下列关于 C++语言中指针的叙述，不正确的是（ ）。

A. 指针变量中存储的是内存地址。

B. 定义指针变量时必须指定其指向的类型。

C. 指针变量只能指向基本类型变量，不能指向指针变量。

D. 指针变量指向的内存地址不一定能够合法访问。

6.单选题 [GESP202309【4 级】] 下列关于 C++语言中指针的叙述，不正确的是（ ）。

A. 可以定义指向 int 类型的指针。

B. 可以定义指向自定义结构体类型的指针。

C. 自定义结构体类型可以包含指针类型的元素。

D. 不能定义指向 void 类型的指针，那没有意义。

2. 指针的简单使用

7.单选题 [GESP202306【4 级】] 一个变量定义为 int *p = nullptr;，则下列说法正确的是（ ）。

A. 该指针变量的类型为 int。

B. 该指针变量指向的类型为 int。

C. 该指针变量指向的内存地址是随机的。

D. 访问该指针变量指向的内存会出现编译错误。

8.单选题 [GESP 样题【4 级】] 在 C++中，指针变量的大小（单位：字节）是（ ）

A. 2

B. 4

C. 8

D. 与编译器有关

9.单选题 [GESP 样题【4 级】] 以下哪个选项是 C++中正确的指针变量声明（ ）

A. int *p;

B. int p*;

C. *int p;

D. int* p*;

10.单选题 [GESP202309【4 级】] 如果 n 为 int 类型的变量，一个指针变量定义为 int *p = &n; ，则下列说法正确的是（ ）。

A. 指针变量 p 的值与变量 n 是相同的。

B. 指针变量 p 的值与变量 n 的地址是相同的。

C. 指针变量 p 指向的值为 'n' 。

D. 指针变量 p 指向的值与变量 n 的地址是相同的。

11.判断题 [GESP202406【4 级】] 下面代码输出的值等于 0。

```
#include<iostream>
using namespace std;
int main()
{
    int *p=NULL;
    cout<<p<<endl;
}
```

12.单选题 [CSP-J2022] 运行以下代码片段的行为是（ ）。

```
int x=101;
int y=201;
int *p=&x;
int *q=&y;
p=q;
```

A. 将 x 的值赋为 201

B. 将 y 的值赋为 101

C. 将 q 指向 x 的地址

D. 将 p 指向 y 的地址

13.单选题 [GESP202312【4 级】] 如果变量 x 的地址是 0x6ffe14,下面C++代码执行以后输出的是（ ）。

```
int *p=NULL;
int x=2;
p=&x;
p++;
cout<<p<<endl;
```

A. 0x6ffe11

B. 0x6ffe14

C. 0x6ffe18

D. 0x6ffe15

14.单选题 [GESP202406【4 级】] 假设变量 a 的地址是 0x6ffe14，下面程序的输出是（ ）。

```
#include<iostream>
using namespace std;
int main()
{
    int *p;
    int a=10;
    p=&a;
    p++;
    cout<<p<<endl;
}
```

A. 10

B. 0x6ffe14

C. 0x6ffe15

D. 0x6ffe18

15.单选题 [GESP202406【4 级】] 如果下列程序输出的地址是 0x6ffe00 ，则 cout<<a+1<<endl; 输出的是（ ）

```
#include<iostream>
using namespace space std;
int main()
{
    int *p;
    int a=10;
    p=&a;
    p++;
    cout<<p<<endl;
}
```

A. 0x6ffe04

B. 0x6ffe0C

C. 0x6ffe08

D. 0x6ffe00

3. 数字数组与指针

16.判断题 [GESP202312【3 级】] 下面C++代码将输出 1

```
int list[10]={1};
cout<<list<<endl;
```

17.单选题 [GESP202403【4 级】] 对二维数组 int arr[3][16]; ，则 arr[1] 占用内存的大小为（ ）字节。

A. 4

B. 16

C. 48

D. 64

18.单选题 [GESP202403【4 级】] 对二维数组 int arr[3][16]; ，若 arr 的地址是 0x28cbc0 ，则 arr[1] 的值是（ ）。

A. 0x28cbc4

B. 0x28cbd0

C. 0x28cc00

D. 不确定

19.单选题 [GESP202306【4 级】/GESP202309【4 级】] 一个数组定义为 int a[5] = {1, 2, 3, 4, 5};，一个指针定义为 int * p = &a[2];，则执行 a[1] = *p;后，数组 a 中的值会变为（ ）。

A. {1, 3, 3, 4, 5}

B. {2, 2, 3, 4, 5}

C. {1, 2, 2, 4, 5}

D. {1, 2, 3, 4, 5}

20.单选题 [GESP202312【4 级】] 下面C++代码执行后输出是（ ）。

```
int arr[3]={1,2,3};
int *p=NULL;
p=arr;
p++;
cout<<*p<<endl;
```

A. 1,2,3

B. 1

C. 2

D. 3

21.判断题 [GESP202406【4 级】] 下面程序两个输出结果是一样的。

```
#include<iostream>
using namespace std;
int main()
{
    int a[2][3]={0};
    cout<<a<<endl;
    cout<<&a[0][0]<<endl;
}
```

22.单选题 [GESP202406【4 级】] 下面程序中，如果语句 cout<<p<<endl; 输出的是 0x6ffe00 ，则 cout<<++p<<endl; 输出的是（ ）

```
int x[10][10][10]={{0}};
int *p;
p=&x[0][0][0];
cout<<p<<endl;
cout<<++p<<endl;
```

A. 0x6ffe0c

B. 0x6ffe09

C. 0x6ffe06

D. 0x6ffe04

4. 字符串与指针

23.单选题 [GESP 样题【5 级】] 使用如下代码片段定义四个字符串（假设头文件已正确定义），以下说法错误的是（ ）。

```
string str1=""abc"";
string str2= str1;
char str3[]=""abc"";
char *str4= str3;
```

A. 对于四个字符串，都可以使用 std::cout 输出其中的内容（例如，cout << str3;

B. str3 只占用 4 字节内存，但 str1 却要占用更多内存

C. 由于 str2 由 str1 直接赋值得到，因此二者指向同一块内存，即修改 str1 的内容后 str2 的内容也会随之改变

D. 由于 str4 由 str3 直接赋值得到，因此二者指向同一块内存，即修改 str3 的内容后 str4 的内容也会随之改变

24.单选题 [GESP202403【4 级】] 下面 C++代码执行后输出是（ ）

```
int main(){
    char *p=""I love GESP!"";
    cout<<p+5 << endl;
    cout << endl;
    return 0;
}
```

A. e

B. I lov

C. e GESP!

D. GESP!

25.单选题 [GESP202403【4 级】] 执行下列 C++代码时输出中的第 2 行是（ ）。

```
int main(){
    char *s[]={(char*)""2024"",(char*)""3.16"",(char*)""GESP""};
```

```
    for(int i=0;i< 2; i++){
        cout<<*s+i << endl;
    }
    cout << endl;
    return 0;
}
```

A. 2024

B. 3.16

C. 024

D. 3

5. 结构体与指针

26.单选题 [GESP202403【4 级】] 在如下的 C++代码执行后，设第 11 和 12 行的输出地址值分别为 X 和 Y ，则下面正确的是（ ）。

```
struct pass{
    int no;
    char name[20];
    int level;
};
int main(){
    struct pass XiaoYang;
    cout << ""&XiaoYang=""<< &xiaoYang << endl;//第 11 行
    cout <<""&(xiaoYang.no)=""<<&(xiaoYang.no)<< endl; //第 12 行
    cout << end;
    return 0;
}
```

A. X>Y

B. X==Y

C. X<Y

D. 不确定

6. 函数与指针

1. 函数传递参数的方法

27.单选题 [GESP 样题【4 级】] 在 C++中，以下哪种方式不能用于向函数传递参数（ ）

A. 值传递

B. 引用传递

C. 指针传递

D. 模板传递

28.判断题 [GESP 样题【4 级】/GESP202306【4 级】] 在 C++语言中，函数的参数默认以地址传递方式进行传递。

29.判断题 [GESP202403【4 级】] 在 C++语言中，函数的参数为指针时，可以在函数内部修改该参数的值。

30.判断题 [GESP202309【4 级】] 在 C++语言中，在函数调用时，通过引用传递的参数不会复制实际参数，因此不会额外占用内存。

31.单选题 [GESP202312【4 级】] 下面有关函数参数的说法，正确的是( )。

A. 函数参数传递时，主函数当中采用值传递方式将参数传递给子函数时，若子函数将参数值改变，主函数当中的参数值不变。

B. 函数参数传递时，主函数当中采用值传递方式将参数传递给子函数时，若子函数将参数值改变，主函数当中的参数值将随子函数一样改变而改变。

C. 函数参数传递时，主函数如果将参数的地址传递给子函数，若子函数将参数值改变，主函数当中的参数值将不改变。

D. 函数参数传递可以不满足子函数的参数个数要求。

32.判断题 [GESP 样题【4 级】] 如果希望设计一个函数 xchg，实现交换两个 int 变量的值，则它的声明可以写为 void xchg(int a, int b);。

33.判断题 [GESP202406【4 级】] 函数参数传递过程中，如果传常量值、常量引用和常量指针都是不能被修改的，它们可以防止函数对实参的值或地址进行修改。

2. 函数传递指针

34.单选题 [GESP202312【4 级】] 在 C++中,执行下面代码后，输出的是（ ）。

```
int point(int *p){
    return *p**p;
}
int main(){
    int a=20;
    int *p=&a;
    *p=point(p);
    cout<<*p<<endl;
}
```

A. 400

B. 200

C. 20

D. 100

35.单选题 [GESP202306【4 级】] 在下列代码的横线处填写（ ），可以使得输出是“20 10”。

```
#include <iostream>
using namespace std;
void xchg(_____){// 在此处填入代码
    int t= *x;
    *x = *y;
    *y=t;
}
int main(){
    int a=10,b= 20;
    xchg(&a, &b);
    cout << a<<""""<< b<< endl;
    return 0;
}
```

A. int x, int y

B. int * x, int * y

C. int a, int b

D. int & a, int & b

36.填空题 [NOIP 普及组 2007]

```
#include<stdio.h>
void fun( int *a, int *b )
{
    int *k;
    k = a; a = b; b = k;
}
int main()
{
    int a = 3, b = 6, *x = &a, *y = &b;
    fun( x, y );
    printf( ""%d,%d "", a, b );
}
```

输出:

37.填空题 [NOIP 提高组 2007]

```
#include <stdio.h>
void fun( int *a, int *b )
{
    int *k;
    k = a; a = b; b = k;
}
main()
{
    int a = 3, b = 6, *x = &a, *y = &b;
    fun( x, y );
    printf( ""No.1: %d,%d "", a, b );
    fun( &a, &b );
    printf( ""No.2: %d,%d\n"", a, b );
}
```

输出:

38.填空题 [NOIP 普及组 2015/NOIP 提高组 2015]

```
#include<iostream>
using namespace std;
int fun(char *a, char *b)
{
    a = b;
    (*a)++;
}
int main()
{
    char c1,c2,*p1,*p2;
    c1='A';
    c2='a';
    p1 = &c1;
    p2 = &c2;
    fun(p1,p2);
    cout << c1 << c2 << endl;
    return 0;
}
```

输出：（ ）

3. 函数传递数组

39.单选题 [GESP202312【4 级】] 下面C++代码最后执行后输出是( )。

```
int fun1(int *n)
{
    return *n**n;
}
int main(){
    int arr[10]={2};
    arr[1]=fun1(arr);
    cout<<arr[1]<<endl;
}
```

A. 1

B. 2

C. 3

D. 4

1.单选题 [GESP202403【4 级】] 下面 C++代码执行后输出是( )。

```
int foo(float *f){
    return int(*f*2);
}
int main(){
    float fnum[10]={1.1};
    fnum[1]=foo(fnum);
    cout << fnum[0]+fnum[1]<< endl;
    cout << end1;
    return 0;
}
```

A. 1

B. 1.1

C. 3

D. 3.1

2.单选题 [GESP202406【8 级】] 以下函数声明，哪个是符合 C++语法的？（ ）。

A. void BubbleSort(char a[][], int n);

B. void BubbleSort(char a[][20], int n);

C. void BubbleSort(char a[10][], int n);

D. void BubbleSort(char[,] a, int n);

3.单选题 [GESP 样题【4 级】/GESP202306【4 级】] 以下哪个函数声明在调用时可以传递二维数组的名字作为参数？

A. void BubbleSort(int a[3][4]);

B. void BubbleSort(int a[][]);

C. void BubbleSort(int * a[]);

D. void BubbleSort(int ** a);

4.单选题 [GESP202312【8 级】] 以下哪个函数声明是符合语法的，且在调用时可以将二维数组的名字作为实际参数传递给形式参数 a ？（ ）。

A. void QuickSort(int a[][10], int n);

B. void QuickSort(int a[5][], int m);

C. void QuickSort(int a[][], int n, int m);

D. void QuickSort(int ** a, int n, int m);

5.填空题 [NOIP 普及组 2006]

```
#include ""stdio.h""
#define N 7
int fun(char s[],char a,int n)
{
    int j;
    j=n;
    while(a<s[j]&&j>0) j--;
    return j;
}
int main()
{
    char s[N+1];
    int k,p;
    for(k=1;k<=N;k++)
       s[k]='A'+2*k+1;
    printf(""%d\n"",fun(s,'M',N));
}
```

输出： _____

6.填空题 [NOIP 提高组 2006]

```
#include ""stdio.h""
#define N 7
int fun1(char s[],char a,int n)
{
      int j;
      j=n;
      while(a<s[j] && j>0) j--;
      return j;
}
int fun2(char s[],char a,int n)
{
      int j;
      j=1;
```

```
    while(a>s[j] && j<=n) j++;
    return j;
}
void main()
{
    char s[N+1];
    int k,p;
    for(k=1;k<=N;k++)
        s[k]='A'+2*k+1;
    p=fun1(s,'M',N);
    printf( ""%d\n”,p+fun2(s,'M',N));
}
```

输出：_____

7.填空题 [NOIP 普及组 2007]

```
#include ""ctype.h""
#include ""stdio.h""
void expand( char s1[], char s2[] )
{
    int i, j, a, b, c;
    j = 0;
    for ( i = 0; (c = s1[i]) != '\0'; i++ )
        if ( c == '-' ){
            a = s1[i - 1]; b = s1[i + 1];
            if ( isalpha( a ) && isalpha( b ) || isdigit( a ) && isdigit( b ) ) /*函数 isalpha(a) 用于判断字符 a 是否为字母，isdigit(b) 用于判断字符 b 是否为数字,如果是,返回 1，否则返回 0 */
            {
                j--;
                do
                    s2[j++] = a++;
                while ( tolower( a ) < tolower( s1[i + 1] ) );
            }/*函数 tolower(a) 的功能是当字符 a 是大写字母,改为小写,其余情况不变*/
            else
                s2[j++] = c;
        }
        else
            s2[j++] = c;
    s2[j] = '\0';
}
int main()
{
```

```
    char s1[100], s2[300];
    printf( ""input s1:"" );
    gets( s1 );
    expand( s1, s2 );
    printf( ""%s\n"", s2 );
}
```

输入：wer2345d-h454-82qqq

输出：

8.填空题 [NOIP 普及组 2007/NOIP 普及组 2007] （求字符串的逆序）下面的程序的功能是输入若干行字符串，每输入一行，就按逆序输出该行，最后键入-1 终止程序。请将程序补充完整

```
#include <iostream.h>
#include <string.h>
int maxline=200,kz;
int reverse(char s[])
{
     int i,j,t;
    for(i=0,j=strlen(s)-1; i<j; ①  ,   ② )
    {
         t=s[i]; s[i]=s[j]; s[j]=t;
     }
    return 0;
}
void main()
{
    char line[100];
    cout<<""continue? -1 for end.""<<endl;
    cin>>kz;
    while( ③ )
    {
        cin>>line;
    ④ ;
    cout<<line<<endl;
    cout<<""continue? -1 for end.""<<endl;
    cin>>kz;
    }
}
```

答案

1.A

2.T

3.F

4.T

5.C

6.D

7.B

8.D

9.A

10.B

11.T

12.D

13.C

14.D

15.B

16.F

17.D

18.C

19.A

20.C

21.T

22.D

23.C

24.C

25.C

26.B

27.D

28.F

29.F

30.F

31.A

32.F

33.T

34.A

35.B

36.3,6

37.No.1:3,6 No.2:3,6

38.Ab

39.D

1.D

2.B

3.A

4.A

5.5

6.11

7.wer2345defgh45456782qqq

8.

正确答案： i++ / i=i+1 / i+=1 / ++i

正确答案： j-- / j=j-1 / j-=1 / --j

正确答案： kz!=-1

正确答案： reverse(line)

GW 信奥 CSP 初赛习题集 5-1

计算机数学——离散数学

1. 数理逻辑

1. 逻辑值计算

1.单选题 [NOIP 普及组 2005] 设 A = true，B = false，C = false，D = true，以下逻辑运算表达式值为真的是（ ）。

A. (A∧B)∨(C∧D)

B. ((A∧B)∨C)∧D

C. A∧((B∨C)∧D)

D. (A∧(B∨C))∨D

E. (A∨B)∧(C∧D)

2.不定项选择题 [NOIP 提高组 2005] 设 A = true，B = false，C = false，D = true，以下逻辑运算表达式值为真的有（ ）。

A. (A∧B)∨(C∧D)

B. ((A∧B)∨ C) ∧ D

C. A∧((B∨C) ∨ D)

D. (A∧(B∨ C)) ∨D

E. (A ∨B)∧(C ∨D)

3.单选题 [NOIP 普及组 2006] 设 A=B=D=true，C=false，以下逻辑运算表达式值为真的有（ ）。

A. (¬A∧B)∨(C∧D)

B. ¬((A∨B∨D)∧C)

C. ¬A∧(B∨C∨D)

D. (A∧B∧C)∨¬D

4.不定项选择题 [NOIP 提高组 2006] 设 A=B=D=true，C=E=false，以下逻辑运算表达式值为真的有（ ）。

A. (¬A∧B)∨(C∧D)∨¬E

B. ¬(((A∧B)∨C)∧D∧E)

C. A∧(B∨C∨D∨E)

D. (A∧(B∨C)) ∧D∧E

5.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] 设 A=B=true，C=D=false，以下逻辑运算表达式值为假的有（ ）。

A. (¬ A∧B)∨(C∧D∨A)

B. ¬ (((A∧B)∨C)∧D)

C. A∧(B∨C∨D)∨D

D. (A∧(D∨C)) ∧B

6.单选题 [NOIP 普及组 2008] 设 A=true，B=false，C=true，D=false，以下逻辑运算表达式值为真的是（ ）。

A. (A∧B)∨(C∧D∨¬A)

B. ((¬A∧B)∨C)∧¬D

C. (B∨C∨D)∧D∧A

D. A∧(D∨¬C)∧B

7.不定项选择题 [NOIP 提高组 2008] 设 A=true，B=false，C=true，D=false，以下逻辑运算表达式值为真的有（ ）。

A. (A∧B)∨(C∧D∨¬A)

B. ((¬A∧B)∨C)∧¬D

C. (B∨C∨D)∨D∧A

D. A∧(D∨¬C)∧B

8.单选题 [CSP-J2020] 设 x=true,y=true,z=false，以下逻辑运算表达式值为真的是（ ）。

A. (y∨z)∧x∧z

B. x∧(z∨y) ∧z

C. (x∧y) ∧z

D. (x∧y)∨(z∨x)

9.不定项选择题 [NOIP 提高组 2014] 若逻辑变量 A、C 为真， B、D 为假，以下逻辑运算表达式真的有 （ ）.

A.(B ∨C∨D)∨D∧A

B.((¬ A ∧ B)∨C)∧B

C.(A ∧B)∨(C∧D∨¬A)

D.A ∧(D∨ ¬C)∧B

10.不定项选择题 [NOIP 提高组 2007] 命题“P→Q”可读做 P 蕴涵 Q，其中 P、Q 是两个独立的命题。只有当命题 P 成立而命题 Q 不成立时，命题“P→Q”的值为 false，其他情况均为 true。与命题“P→Q”等价的逻辑关系式是（ ）。

A. ¬ P∨Q

B. P∧Q

C. ¬ (P∨Q)

D. ¬ (¬ Q∧P)

11.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 以下逻辑表达式的值恒为真的是（ ）

A. P∨(┐P ∧Q) ∨(┐P∧┐Q)

B. Q∨ (┐P∧Q) ∨(P∧┐Q)

C. P∨Q∨(P∧┐Q) ∨(┐P ∧Q)

D. P∨¬Q∨(P∧¬Q)∨(¬P∧¬Q)

12.不定项选择题 [NOIP 提高组 2011] 在布尔逻辑中，逻辑“或”的性质有（ ）

A. 交换律： P∨Q = Q∨P

B. 结合律： P∨(Q∨R) = (P∨Q)∨R

C. 幂等律： P∨P = P

D. 有界律： P∨1 = 1 (1 表示逻辑真)

13.单选题 [NOIP 普及组 2013] 逻辑表达式（ ）的值与变量 A 的真假无关。

A. (A ∨ B) ∧¬ A

B. (A ∨ B) ∧¬B

C. (A ∧ B) ∧(¬A ∧ B)

D. (A ∨ B) ∧ ¬A ∧ B

14.不定项选择题 [NOIP 提高组 2012] 逻辑异或⊕是一种二元运算，其真值表如下所示。

a b a ⊕b

F F F

F T T

T F T

T T F

以下关于逻辑异或的性质，正确的有（ ）。

A. 交换律：a⊕b=b⊕a

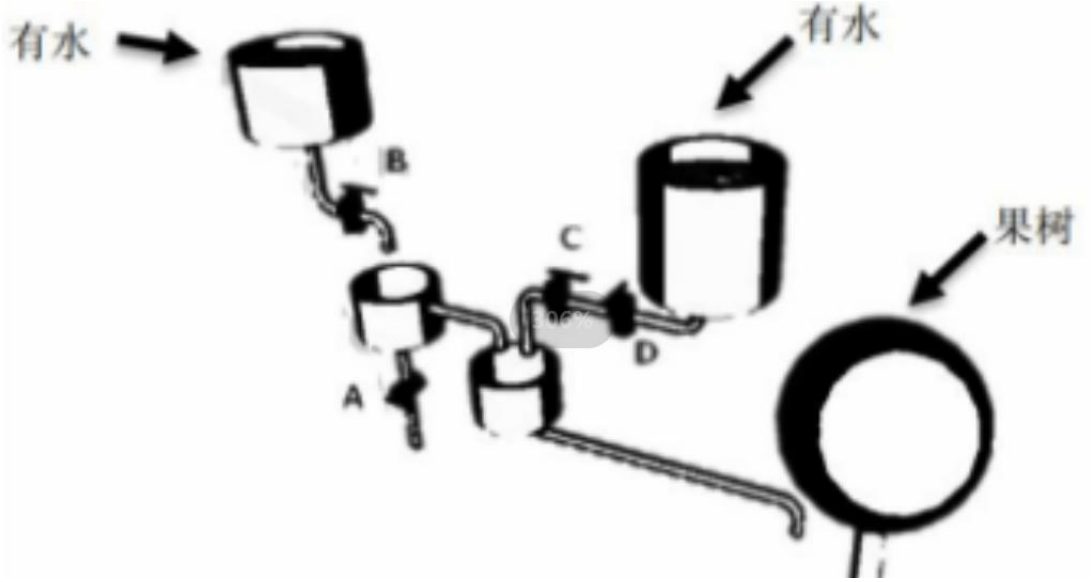
B. 结合律:(a⊕b)⊕c=a⊕(b⊕c)

C. 关于逻辑与的分配律:a⊕(b∧c)=(a⊕b)∧(a⊕c)

D. 关于逻辑或的分配律:a⊕(b∨c)=(a⊕b)∨(a⊕c)

2. 逻辑推理

15.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 下图表示一个果园灌溉系统，有 A、B、C、D 四个阀门，每个阀门可以打开或关上，所有管道粗细相同，以下设置阀门的方法中，可以让果树浇上水的有（ ）。



A. B 打开，其他都关上

B. AB 都打开， CD 都关上

C. A 打开，其他都关上

D. D 打开，其他都关上

16.填空题 [NOIP 普及组 2018/NOIP 提高组 2018] 甲乙丙丁四人在考虑周末要不要外出郊游。已知①如果周末下雨，并且乙不去，则甲一定不去；②如果乙去，则丁一定去；③如果丙去，则丁一定不去；④如果丁不去，而且甲不去，则丙一定不去。如果周末丙去了，则甲____（去了/没去），乙____（去了/没去），丁 ___ （去了/没去），周末____（下雨/没下雨）。

17.单选题 [NOIP 普及组 2013/NOIP 提高组 2013] 某系统自称使用了一种防窃听的方式验证用户密码。密码是 n 个数 s1, s2, , , sn ，均为 0 或 1 。 该系统每次随机生成 n 个数 a1, a2, …… , an ， 均为 0 或 1 ， 请用户回答 (s1a1+ s2a2+ …… + snan) 除以 2 的余数。如果多次的回答总是正确，即认为掌握密码。该系 统认为，

即使问答的过程被泄露，也无助于破解密码——因为用户并没有直接发送密码。 然而，事与愿违。例如，当 n = 4 时，有人窃听了以下 5 次问答：就破解出了密码 s1 = _____ ， s2 = _____ ，s3 = _____ ， s4 = _____ 。(答案用逗号隔开)

问答编号	系统生成的 n个数				掌握密码的用户的回答
	a1	a2	a3	a4	
1	1	1	0	0	1
2	0	0	1	1	0
3	0	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0

二、集合论

18.单选题 [NOIP 提高组 2004] 设全集 I = {a, b, c, d, e, f, g}，集合 A = {a, b, c}，B = {b, d, e}，C = {e, f, g}，那么集合(A -B) U (~C∩B) 为（ ）。

A. {a, b, c, d}

B. {a, b, d, e}

C. {b, d, e}

D. {b, c, d, e}

E. {d, f, g}

19.单选题 [NOIP 普及组 2005/NOIP 提高组 2005] 设全集 I = {a, b, c, d, e, f, g, h}，集合 A = {a, b, c, d, e, f}，B = {c, d, e}，C = {a, d}，那么集合 A∩B∩~C 为（ ）。

A. {c, e}

B. {d, e}

C. {e}

D. {c, d, e}

E. {d, f}

答案

1.D

2.CDE

3.B

4.ABC

5.D

6.B

7.BC

8.D

9.AB

10.AD

11.A

12.ABCD

13.C

14.AB

15.A

16.去了 没去 没去 没下雨

17.0 1 1 1

18.A

19.A

GW 信奥 CSP 初赛习题集 5-2

计算机数学——初等数论

1. 模运算的应用

1.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 2017 年 10 月 1 日是星期日，1999 年 10 月 1 日是( )。

A. 星期三

B. 星期日

C. 星期五

D. 星期二

2.单选题 [GESP202312【1 级】] 假设现在是上午十点，求出 N 小时（正整数）后是第几天几时，如输入 20 小时则为第 2 天 6 点，如 N 输入 4 则为今天 14 点。为实现相应功能，应在横线处填写代码是( )。

```
int N, dayX,hourX;
    cin >>N;
    dayX =_____,hourX =_______;
    if (dayX ==0)
        cout<<""今天""<<hourX <<""点"";
    else
        cout<<""第""<<(dayX+1)<<""天""<<hourX <<""点"";
```

A. (10 + N) % 24 , (10 + N) / 24

B. (10 + N) / 24 , (10 + N) % 24

C. N % 24 , N / 24

D. 10 / 24 , 10 % 24

3.单选题 [CSP-J2020] 干支纪年法是中国传统的纪年方法，由 10 个天干和 12 个地支组合成 60 个天干地支。由公历年份可以根据以下公式和表格换算出对应的天干地支。

天干 =（公历年份）除以 10 所得余数

地支 =（公历年份）除以 12 所得余数

天干	甲	乙	丙	丁	戊	己	庚	辛	壬	癸		
	4	5	6	7	8	9	0	1	2	3		
地支	子	丑	寅	卯	辰	巳	午	未	申	酉	戌	亥
	4	5	6	7	8	9	10	11	0	1	2	3

例如，今年是 2020 年，2020 除以 10 余数为 0，查表为""庚”；2020 除以 12，余数为 4，查表为“子” 所以今年是庚子年。

请问 1949 年的天干地支是（ ）

A. 己酉

B. 己亥

C. 己丑

D. 己卯

4.单选题 [NOIP 普及组 2016] 如果开始时计算机处于小写输入状态，现在有一只小老鼠反复按照 CapsLock、 字母键 A、字母键 S 和字母键 D 的顺序循环按键，即 CapsLock、A、 S、D、CapsLock、 A、S、D、……，屏幕上输出的第 81 个 字符是字母（）。

A. A

B. S

C. D

D. a

5.单选题 [NOIP 提高组 2016] 如果开始时计算机处于小写输入状态 ，现在有一只小老鼠反复按照 CapsLock、 字母键 A、字母键 S 和字母键 D 的顺序来回按键 ，即 CapsLock、A、S、D、 S、A、CapsLock、A、S、D、S、A、CapsLock、A、S、D、S、A、……， 屏幕上输出的第 81 个字符是字母 （ ）。

A. A

B. S

C. D

D. a

6.单选题 [NOIP 普及组 2018] 如果开始时计算机处于小写输入状态，现在有一只小老鼠反复按照 CapsLock、字母键 A、字母键 S、字母键 D、字母键 F 的顺序循环按键，即 CapsLock、A、S、D、F、CapsLock、A、S、D、F、……，屏幕上输出的第 81 个字符是字母（ ）。

A. A

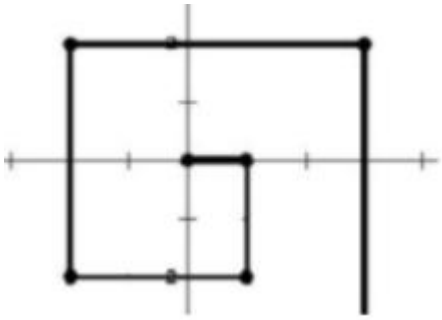
B. S

C. D

D. a

7.填空题 [NOIP 提高组 2007] N 个人在操场里围成一圈，将这 N 个人按顺时针方向从 1 到 N 编号，然后，从第一个人起，每隔一个人让下一个人离开操场，显然，第一轮过后，具有偶数编号的人都离开了操场。依次做下去，直到操场只剩下一个人，记这个人的编号为 J(N)，例如，J(5)=3，J(10)=5，等等。则 J(400)=_____。（提示：对 N=2^m+r 进行分析，其中 0≤r<2^m ）。

8.填空题 [NOIP 普及组 2017] 一个人站在坐标（0，0）处，面朝 x 轴正方向。第一轮，他向前走 1 单位距离，然后右转；第二轮，他向前走 2 单位距离，然后右转；第三轮，他向前走 3 单位距离，然后右转……他一直这么走下去。请问第 2017 轮后，他的坐标是：（___，___）。



9.单选题 [GESP202403【2 级】] 下面 C++代码执行后的输出是（ ）。

```
int n,masks,days,cur;
n=17,masks=10,day=0;
cur=2;
while(masks!=n){
    if(cur==0||cur==1)
        masks+=7;
    masks-=1;
    days+=1;
    cur=(cur+1)%7;
}
```

A. 5

B. 6

C. 7

D. 8

10.填空题 [NOIP 普及组 2009]

```
#include<stdio.h>
const int c=2009;
int main()
{
    int n,p,s,i,j,t;
    scanf(""%d%d"",&n,&p);
    s=0;t=1;
    for(i=1;i<=n;i++)
    {
        t=t*p%c;
        for(j=1;j<=i;j++)
        s=(s+t)%c;
    }
```

```
    printf(""%d\n"",s);
    return 0;
}
```

11.填空题 [NOIP 提高组 2004]

```
#include <stdio.h>
const int u[3]={1,-3,2};
const int v[2]={-2,3};
int g(int n){
    int i,sum =0;
    for(i=1;i<=n;i++)
        sum += u[i % 3]* i;
    return sum;
}
int main(){
    int n,i,sum =0;
    scanf(""%d"",&n);
    for(i=1;i<=n;i++)
        sum += v[i% 2]* g(i);
    printf(""%d\n"",sum);
    return 0;
}
```

输入：103

输出：

12.填空题 [NOIP 普及组 2015] （打印月历）输入月份 m(1≤m≤12)，按一定格式打印 2015 第 m 月的月历。

例如，2015 年一月的月历打印效果如下（第一列为周日）：

```
#include<iostream>
using namespace std;
const int dayNum[]={-1,31,28,31,30,31,30,31,31,30,31,30,31};
int m, offset, i;
int main()
{
    cin >> m;
    cout <<""S    M    T    W    T    F    S""<<endl;//  '为 tab 制表符
    ①;
    for (i = 1; i < m; i++)
```

```
            offset = ②;
        for (i = 0; i < offset; i++)
            cout <<' ';
        for (i = 1; i <= ③;i++)
        {
            cout << ④;
            if(i==dayNum[m]||⑤==0)
                cout << endl;
            else
                cout << '    ';
        }
        return 0;
}
```

13.填空题 [NOIP 普及组 2004/NOIP 提高组 2004] Joseph

题目描述:

原始的 Joseph 问题的描述如下：有 n 个人围坐在一个圆桌周围，把这 n 个人依次编号为 1，…，n。从编号是 1 的人开始报数，数到第 m 个人出列，然后从出列的下一个人重新开始报数，数到第 m 个人又出列，…，如此反复直到所有的人全部出列为止。比如当 n=6，m=5 的时候，出列的顺序依次是 5，4，6，2，3，1。

现在的问题是：假设有 k 个好人和 k 个坏人。好人的编号的 1 到 k，坏人的编号是 k+1 到 2k。我们希望求出 m 的最小值，使得最先出列的 k 个人都是坏人。

输入:仅有的一个数字是 k（0 < k <14）。

输出:使得最先出列的 k 个人都是坏人的 m 的最小值。

输入样例:4

输出样例:30

程序：

```
#include <stdio.h>
long k, m, begin;
int check(long remain){
    long result = (   ①   ) % remain;
    if (   ②   ){
        begin = result; return 1;
    }
    else return 0;
}
```

```
int main(){
    long i, find = 0;
    scanf(""%ld"", &k);
    m = k;
    while(  ③  ) {
        find = 1; begin = 0;
        for (i = 0; i < k; i++)
            if (!check(  ④  )){
                find = 0; break;
            }
        m++;
    }
    printf(""%ld\n"",    ⑤  );
    return 0;
}
```

14.单选题 [CSP-J2021]（Josephus 问题）有 n 个人围成一个圈，依次标号 0 至 n−1。从 0 号开始，依次 0,1,0,1,…交替报数，报到 1 的人会离开，直至圈中只剩下一个人。求最后剩下人的编号。

```
#include <iostream>
using namespace std;

const int MAXN = 1000000;
int F[MAXN];

int main() {
    int n;
    cin >> n;
    int i = 0, p = 0, c = 0;
    while (①) {
        if (F[i] == 0) {
            if (②) {
                F[i] = 1;
                ③;
            }
            ④;
        }
        ⑤;
    }

    int ans = -1;
    for (i = 0; i < n; i++) {
```

```
            if (F[i] == 0)
                ans = i;
        }
        cout << ans << endl;
        return 0;
}
```

①处应填（ ）

A.i < n

B.c < n

C.i < n- 1

D.c < n-1

②处应填（ ）

A.i % 2 == 0

B.i % 2 == 1

C.p

D.!p

③处应填（ ）

A.i++

B.i = (i + 1) % n

C.c++

D.p ^= 1

④处应填（ ）

A.i++

B.i = (i + 1) % n

C.c++

D.p ^= 1

⑤处应填（ ）

A.i++

B.i = (i + 1) % n

C.c++

D.p ^= 1

1.

A. i < n

B. c < n

C. i < n - 1

D. c < n - 1

2.

A. i % 2 == 0

B. i % 2 == 1

C. p

D. !p

3.

A. i++

B. i = (i + 1) % n

C. c++

D. p ^=1

4.

A. i++

B. i = (i + 1) % n

C. c++

D. p ^=1

5.

A. i++

B. i = (i + 1) % n

C. c++

D. p ^=1

15.填空题 [NOIP 普及组 2009]

```
#include <stdio.h>
```

```
int main()
{
    int a[3],b[3];
    int i,j,tmp;
    for (i=0;i<3;i++)
    scanf(""%d"",&b[i]);
    for (i=0;i<3;i++)
    {
        a[i]=0;
        for (j=0;j<=i;j++)
        {
            a[i]+=b[j];
            b[a[i]%3]+=a[j];
        }
    }
    tmp=1;
    for (i=0;i<3;i++)
    {
        a[i]%=10;
        b[i]%=10;
        tmp*=a[i]+b[i];
    }
    printf(""%d\n"",tmp);
    return 0;
}
```

输入： 2 3 5

输出： _______

16.填空题 [NOIP 提高组 2009]

```
#include <iostream>
using namespace std;
int main()
{
    int a[4],b[4];
    int i,j,tmp;
    for (i=0;i<4;i++)
    cin >> b[i];
    for (i=0;i<4;i++)
    {
        a[i]=0;
        for (j=0;j<=i;j++)
        {
```

```
                a[i]+=b[j];
                b[a[i]%4]+=a[j];
            }
        }
        tmp=1;
        for (i=0;i<4;i++)
        {
            a[i]%=10;
            b[i]%=10;
            tmp*=a[i]+b[i];
        }
        cout << tmp << endl;
        return 0;
}
```

输入： 2 3 5 7

输出： _____

2. 因倍数问题

1. 求因数

17.判断题 [GESP202312【5 级】/GESP202312【6 级】] 小杨想写一个程序来算出正整数 N 有多少个因数，经过思考他写出了一个重复没有超过 N/2 次的循环就能够算出来了。

18.单选题 [GESP202309【1 级】/GESP202312【2 级】] 下面 C++代码用于求正整数的所有因数，即输出所有能整除一个正整数的数。如，输入 10，则输出为 1、2、5、10；输入 12，则输出为 1、2、3、4、6、12；输入 17，则输出为 1、17。在横线处应填入代码是（ ）。

```
int n=0:
cout<<""请输入一个正整数:"";
cin >>n;
for(_______)//此处填写代码
    if(n %i == 0)
        cout << i<< endl;
```

A. int i = 1; i < n; i + 1

B. int i = 1; i < n + 1; i + 1

C. int i = 1; i < n; i++

D. int i = 1; i < n + 1; i++

19.单选题 [GESP202309【2 级】] 以下 C++代码实现从大到小的顺序输出 N 的所有因子。例如，输入 N = 18 时输出 18 9 6 3 2 1 ，横线处应填入（ ）。

```
int N =0;
cin >>N;
for(_______)//此处填写代码
    if(!(N % i))
        cout << i <<' ';
```

A. ; ;

B. int i = 1; i < N; i++

C. int i = N; i > 0; i--

D. int i = N; i > 1; i--

20.单选题 [GESP202312【4 级】] 以下 C++代码用于实现每个整数对应的因数，如输入 12 ，则输出 1 2 3 4 6 12 ；如输入 18 ，则输出 1 2 3 6 9 18 。横线处应填入代码是

```
int n:
cin >>n;
for(int i=1;i<=n;i++){
    _______//此处填写代码
        cout << i<< endl;
}
```

A. if(n%i==0)

B. if(n/i==0)

C. if(n%i!=0)

D. if(n/i!=0)

21.单选题 [GESP202312【2 级】/GESP202312【4 级】] 输入一个正整数 N，想找出它所有相邻的因数对，比如，输入12，因数对有(1,2)、(2,3)、(3,4)。下面哪段代码找不到所有的因数对？（ ）

A. for(i=1;i<N;i++) if(!(N%i) && !(N%(i+1))) printf(""(%d,%d)\n"", i, i+1);

B. for(i=2;i<N;i++) if(!(N%i) && !(N%(i+1))) printf(""(%d,%d)\n"", i, i+1);

C. for(i=2;i<N/2;i++) if(!(N%(i-1)) && !(N%i)) printf(""(%d,%d)\n"", i-1, i);

D. for(i=1;i<N/2;i++) if(!(N%i) && !(N%(i+1))) printf(""(%d,%d)\n"", i, i+1);

22.单选题 [CSP-J2023]

```
#include <iostream>
#include <cmath>
using namespace std;

int solve1(int n){
      return n*n;
}

int solve2(int n){
    int sum=0;
    for(int i=1;i<=sqrt(n);i++){
          if(n%i==0){
              if(n/i==i){
                  sum+=i*i;
              }else{
                  sum+=i*i+(n/i)*(n/i);
              }
          }
     }
     return sum;
}
int main(){
      int n;
      cin>>n;
      cout<<solve2(solve1(n))<<"" ""<<solve1((solve2(n)))<<endl;
      return 0;
}
```

假设输入的 n 是绝对值不超过 1000 的整数，完成下面的判断题和单选题。

判断题

如果输入的 n 为正整数，solve2 函数的作用是计算 n 所有的因子的平方和（ ）

第 13~14 行的作用是避免 n 的平方根因子 i（或 n/i）进入第 16 行而被计算两次（ ）

如果输入的 n 为质数，solve2(n) 的返回值为 n^2+1

单选题

如果输入的 n 为质数 p 的平方，那么 solve2(n) 的返回值为（ ）

当输入为正整数时，第一项减去第二项的差值一定（ ）

当输入为 5 时，输出为（ ）

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. p^2+p+1

B. n^2+n+1

C. n^2+1

D. p^4+2p^2+1

5.

A. 大于 0

B. 大于等于 0 且不一定大于 0

C. 小于 0

D. 小于等于 0 且不一定小于 0

6.

A. 651 625

B. 650 729

C. 651 676

D. 652 625

23.单选题 [CSP-J2022] （枚举因数）从小到大打印正整数 n 的所有正因数。试补全枚举程序。

```
#include <bits/stdc++.h>
using namespace std;

int main(){
    int n;
    cin >> n;

    vector<int> fac;
    fac.reserve((int)ceil(sqrt(n)));

    int i;
    for (i = 1; i * i < n; ++i){
        if (①){
            fac.push_back(i);
        }
    }

    for (int k = 0; k < fac.size(); ++k){
        cout << ② << """;
    }
    if (③) {
        cout << ④ << """;
    }
    for (int k = fac.size() - 1; k >= 0; --k){
        cout << ⑤ << """;
    }
}
```

①~⑤处应填（ ）

1.

A. n % i == 0

B. n % i == 1

C. n % (i-1) == 0

D. n % (i-1) == 1

2.

A. n / fac[k]

B. fac[k]

C. fac[k]-1

D. n / (fac[k]-1)

3.

A. (i-1)*(i-1)== n

B. (i-1)*i == n

C. i*i == n

D. i*(i-1) == n

4.

A. n-i

B. n-i+1

C. i-1

D. i

5.

A. n / fac[k]

B. fac[k]

C. fac[k]-1

D. n / (fac[k]-1)

2. 求最大公约数和最小公倍数

24.单选题 [CSP-J2019] 319 和 377 的最大公约数是（ ）。

A. 27

B. 33

C. 29

D. 31

25.单选题 [GESP202403【5 级】] 辗转相除法也被称为（ ）

A. 高斯消元法

B. 费马定理

C. 欧几里德算法

D. 牛顿迭代法

26.判断题 [GESP202403【5 级】] 辗转相除法用于求两个整数的最大公约数。

27.单选题 [GESP202406【5 级】] 下面是根据欧几里得算法编写的函数，它计算的是 与 的（ ）。

```
int gcd(int a, int b) {
    while (b != 0) {
        int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}
```

A. 最小公倍数

B. 最大公共质因数

C. 最大公约数

D. 最小公共质因数

28.单选题 [GESP 样题【5 级】/NOIP 普及组 2013] 下列代码中，函数 f 的作用是（ ）。

```
void f(int a, int b){
    return b==0? a:f(b, a % b);
}
```

A. 求 a 和 b 的最大公共质因子

B. 求 a 和 b 的最小公共质因子

C. 求 a 和 b 的最大公约数

D. 求 a 和 b 的最小公倍数

29.单选题 [GESP202406【5 级】] 欧几里得算法还可以写成如下形式:

```
int gcd(int a, int b) {
    return b == 0 ? a : gcd(b, a % b);
}
```

下面有关说法，错误的是（ ）。

A. 本题的 gcd() 实现为递归方式。

B. 本题的 gcd() 代码量少，更容易理解其辗转相除的思想。

C. 当 较大时，本题的 gcd() 实现会多次调用自身，需要较多额外的辅助空间。

D. 当 较大时，相比上题中的 gcd() 的实现，本题的 gcd() 执行效率更高。

30.单选题 [GESP202403【6 级】] 以下代码使用了辗转相除法求解最大公因数，请在横线处填入（ ），使其能正确实现相应功能。

```
int gcd(int a,int b){
    while(b!=0){
        __________;
    }
    return a;
}
```

A. int temp = b; b = a / b; a = temp;

B. int temp = a; a = b / a; b = temp;

C. int temp = b; b = a % b; a = temp;

D. b = a % b; a = b;

31.单选题 [GESP202312【5 级】] 有关下面 C++代码说法正确的是（ ）。

```
int rc;
int foo(int x, int y)
{
    int r;
    if (y == 0)
```

```
        r = x;
    else
    {
        r = foo(y, x % y);
        rc++;
    }
    return r;
}
```

A. 如果 x 小于 10， rc 值也不会超过 20

B. foo 可能无限递归

C. foo 可以求出 x 和 y 的最大公共质因子

D. foo 能够求出 x 和 y 的最小公倍数

32.填空题 [NOIP 普及组 2009]

```
#include <iostream>
using namespace std;

int a, b;
int work(int a, int b) {
    if (a % b)
        return work(b, a % b);
    return b;
}

int main() {
    cin >> a >> b;
    cout << work(a, b) << endl;
    return 0;
}
```

输入：20 12

输出：_____

33.填空题 [NOIP 提高组 2009]

```
#include <iostream>
using namespace std;

int a, b;
```

```
int work(int a, int b) {
    if (a % b)
        return work(b, a % b);
    return b;
}

int main() {
    cin >> a >> b;
    cout << work(a, b) << endl;
    return 0;
}
```

输入：123 321

输出：_____

34.填空题 [NOIP 提高组 2012]

```
#include <iostream>
using namespace std;
int n, i, ans;
int gcd(int a, int b){
    if (a % b == 0)
        return b;
    else
        return gcd(b, a%b);
}
int main(){
    cin>>n;
    ans = 0;
    for (i = 1;i <= n;i++)
        if (gcd(n,i) == i)ans++;
    cout<<ans<<endl;
}
```

输入：120

输出：____

35.单选题 [GESP202312【8 级】] 假设输入参数 m 和 n 满足 ，则下面程序的最差情况的时间复杂度为（ ）。

```
int gcd(int m, int n) {
    while (m > 0) {
```

```
        int t = m;
        m = n % m;
        n = t;
    }
    return n;
}
```

A.O(log(n))

B.O(n)

C.O(n*m)

D.O(m*log(n))

36.填空题 [NOIP 普及组 2018] （最大公约数之和）下列程序想要求解整数 n 的所有约数两两之间最大公约数的和对 10007 求余后的值，试补全程序。举例来说，4 的所有约数是 1,2,4。1 和 2 的最大公约数为 1；2 和 4 的最大公约数为 2；1 和 4 的最大公约数为 1。于是答案为 1 + 2 + 1 = 4。要求 getDivisor 函数的复杂度为 O(√n)，gcd 函数的复杂度为 O(log max(a,b))。

例如:

```
#include <iostream>
using namespace std;
const int N = 110000, P = 10007;
int n;
int a[N], len;
int ans;
void getDivisor() {
    len = 0;
    for (int i = 1; ① <= n; ++i)
        if (n % i == 0) {
            a[++len] = i;
            if ( ② != i) a[++len] = n / i;
        }
    }
}
int gcd(int a, int b) {
    if (b == 0) {
        ③ ;
    }
    return gcd(b, ④ );
}
```

```
int main() {
    cin >> n;
    getDivisor();
    ans = 0;
    for (int i = 1; i <= len; ++i) {
        for (int j = i + 1; j <= len; ++j) {
            ans = ( ⑤ ) % P;
        }
    }
    cout << ans << endl;
    return 0;
}
```

37.单选题 [GESP202406【8 级】] 下面程序的最差时间复杂度为（ ）。

```
int gcd(int m, int n) {
    if (m == 0) return n;
    return gcd(n % m, m);
}
```

A.O(SQRT(n))

B.O(log(n))

C.O(n)

D.O(1)

二、质和数问题

3. 唯一分解定理

38.单选题 [CSP-J2019] 100 以内最大的素数是（）。

A. 89

B. 97

C. 91

D. 93

39.单选题 [NOIP 普及组 2018] 10000 以内，与 10000 互质的正整数有（ ）个。

A. 2000

B. 4000

C. 6000

D. 8000

40.单选题 [GESP202403【5 级】] 唯一分解定理描述的内容是（ ）？

A. 任意整数都可以分解为素数的乘积

B. 每个合数都可以唯一分解为一系列素数的乘积

C. 两个不同的整数可以分解为相同的素数乘积

D. 以上都不对

41.判断题 [GESP 样题【5 级】/GESP 样题【6 级】] 唯一分解定理指的是分解质因数只有唯一的一种算法。

42.判断题 [GESP202312【5 级】] 小杨设计了一个拆数程序，它能够将任意的非质数自然数 N 转换成若干个质数的乘积，这个程序是可以设计出来的。

43.判断题 [GESP202406【7 级】] 唯一分解定理（算术基本定理）指出，每个大于 1 的自然数都可以唯一地分解成若干个素数的乘积。因此，我们可以很容易的对给定的自然数 n 进行质因数分解，时间复杂度仅为 。

44.判断题 [GESP202406【5 级】] 唯一分解定理表明任何一个大于 1 的整数都可以唯一地表示为一系列质数的乘积，即质因数分解是唯一的。

45.单选题 [GESP 样题【6 级】] 现希望存储整数 N 的所有质因子，请问其空间复杂度上界为是（ ）。

A．O(loglogN)

B．O(logN)

C．O(√N)

D．O(N)

46.单选题 [CSP-J2020] （质因数分解）给出正整数 n，请输出将 n 质因数分解的结果，结果从小到大输出。例如：输入 n=120，程序应该输出 2 2 2 3 5，表示 120=2×2×2×3×5。输入保证 2≤n≤10^9。提示：先从小到大枚举变量 i，然后用 i 不停试除 n 来寻找所有的质因子。试补全程序。

```
#include <cstdio>
using namespace std;
int n, i;
int main() {
    scanf(""%d"", &n);
    for (i = ①; ② <= n; i ++) {
        ③ {
            printf(""%d "", i);
            n = n / i;
        }
    }
    if (④) {
        printf(""%d "", ⑤);
    }
    return 0;
}
```

1）①处应填（ ）

2）②处应填（ ）

3）③处应填（ ）

4）④处应填（ ）

5）⑤处应填（ ）

1.

A. 1

B. n-1

C. 2

D. 0

2.

A. n/i

B. n/(i*i)

C. i*i

D. i*i*i

3.

A. if(n%i==0)

B. if(i*i<=n)

C. while(n%i==0)

D. while(i*i<=n)

4.

A. n>1

B. n<=1

C. i<n/i

D. i+i<=n

5.

A. 2

B. n/i

C. n

D. i

4. 判断质数

47.单选题 [GESP202403【5 级】] 下面的代码片段用于判断一个正整数是否为素数。请对以下代码进行修改，使其能正确实现相应功能。（ ）

```
bool isPrime(int num){
    if(num<2){
        return false;
    }
    for(int i=2;i*i<num; ++i)
            if(num%i==0){
                return false;
```

```
            }
    return true;
}
```

A. num < 2 应该改为 num <= 2

B. 循环条件 i * i < num 应该改为 i * i <= num

C. 循环条件应该是 i <= num

D. 循环体中应该是 if (num % i != 0)

48.单选题 [GESP202303【2 级】] 执行以下 C++语言程序后，输出结果是（ ）。

```
#include <iostream>
using namespace std;
int main(){
    int n = 17;
    bool isprime =true;
    for(int i=2;i<= n; i++)
        if(n%i==0)
            isprime = false;
    cout<< isprime<< endl;
    return 0;
}
```

A． false

B． true

C． 0

D．1

49.单选题 [GESP202403【1 级】] 下面 C++代码用于判断键盘输入的整数是否为质数。质数是只能被 1 和它本身整除的数。在横线处应填入代码是（ ）。

```
int N;
cin >>N;
int cnt =0;//记录 N 被整除的次数
for(int i=1; i<N+1; i++)
    if(_______)
        cnt +=1;
if(cnt == 2)
    cout<<""是质数"";
```

```
9  else
10      cout<<""不是质数"";
```

A. N % i

B. N % i == 0

C. N / i == 0

D. N / i

50.单选题 [GESP202312【1 级】] 下面 C++代码用于判断一个数是否为质数(素数),在横线处应填入代码是（ ）。

```
1  cin >>N;
2  int cnt =0;//记录 N 被整除的次数
3  for(int i=1; i<N+1; i++)
4      if(N%i==0)
5          _________;
6  if(cnt == 2)
7      cout<<""是质数"";
8  else
9      cout<<""不是质数"";
```

A. cnt = 1

B. cnt = 2

C. cnt =+ 1

D. cnt += 1

51.单选题 [GESP202309【2 级】] 下面 C++代码用于判断 N 是否为质数（素数），约定输入 N 为大于等于 2 的正整数，请在横线处填入合适的代码（ ）。

```
1  int N=0,i=0;
2  cout<<""请输入一个大于等于 2 的正整数:"";
3  cin >>N;
4  for(i=2;i<N;i++)
5      if(N%i== 0){
6          cout<<""非质数"";
7          ________;//此处填写代码
8      }
9  if(i == N)
```

```
10      cout<<""是质数"";
```

A. break

B. continue

C. exit

D. return

52.单选题 [GESP202312【2 级】] 下面C++代码用于判断 N（大于等于 2 的正整数）是否为质数（素数）。下面对如下代码的说法，正确的是（ ）。

```
1  cin>>N;
2  for(i = 2;i < N/2; i++)
3      if(N %i ==0){
4          cout<<N<<""不是质数"";
5          break;
6      }
7  if(i>= N/2)
8      cout<<N<<""是质数"";
```

A. 代码能正确判断 N 是否为质数。

B. 代码总是不能判断 N 是否质数。

C. 删除第 5行 break ，将能正确判断 N 是否质数。

D. 代码存在漏洞，边界存在问题，应将第 2行和第 7行的 N / 2 改为 N / 2 + 1 。

53.单选题 [GESP202312【5 级】] 下面 C++代码中的 isPrimeA() 和 isPrimeB() 都用于判断参数 N 是否素数，有关其时间复杂度的正确说法是（ ）。

```
1  bool isPrimeA(int N)
2  {
3      if(N<2)
4          return false;
5      for(int i= 2;i<=N/2;i++)
6          if(N%i== 0)
7              return false;
8      return true;
9  }
10 bool isPrimeB(int N){
11     if(N< 2)
12         return false;
```

```
    for(int i=2;i<=sqrt(N);i++)
        if(N%i == 0)
            return false;
    return true;
}
```

A. isPrimeA( ) 的最坏时间复杂度是 O(N/2)， isPrimeB( ) 的最坏时间复杂度是 O(logN) ， isPrimeA() 优于 isPrimeB()

B. isPrimeA() 的最坏时间复杂度是 O(N/2) ， isPrimeB( ) 的最坏时间复杂度是 O(N^0.5) ， isPrimeB() 绝大多数情况下优于 isPrimeA()

C. isPrimeA() 的最坏时间复杂度是 O(N^0.5)， isPrimeB( ) 的最坏时间复杂度是 O(N) ， isPrimeA( ) 优于 isPrimeB( )

D. isPrimeA() 的最坏时间复杂度是 O(logN)， isPrimeB( ) 的最坏时间复杂度是 O(N) ， isPrimeA() 优于 isPrimeB( )

54.单选题 [GESP202309【5 级】] 下面代码中的 isPrimeA() 和 isPrimeB() 都用于判断参数 N 是否素数，有关其时间复杂度的正确说法是（ ）。

```
#include <iostream>
#include <cmath>
using namespace std;
bool isPrimeA(int N) {
    if (N < 2) {
        return false;
    }
    for (int i = 2; i < N; i++) {
        if (N % i == 0) {
            return false;
        }
    }
    return true;
}

bool isPrimeB(int N) {
    if (N < 2) {
        return false;
    }
    int endNum = int(sqrt(N));
    for (int i = 2; i <= endNum; i++) {
        if (N % i == 0) {
```

```
            return false;
        }
    }
    return true;
}

int main() {
    cout << boolalpha;
    cout << isPrimeA(13) <<"" ""<< isPrimeB(13) << endl;
    return 0;
}
```

A. isPrimeA() 的最坏时间复杂度是O(N)，isPrimeB() 的最坏时间复杂度是O(logN)，isPrimeB() 优于 isPrimeA() 。

B. isPrimeA() 的最坏时间复杂度是 O(N)， isPrimeB() 的最坏时间复杂度是 O(N^0.5)， isPrimeB() 优于 isPrimeA() 。

C. isPrimeA() 的最坏时间复杂度是 O(N^0.5)， isPrimeB() 的最坏时间复杂度是 O(N)， isPrimeA() 优于 isPrimeB() 。

D. isPrimeA() 的最坏时间复杂度是O(logN)，isPrimeB() 的最坏时间复杂度是O(N)，isPrimeA() 优于 isPrimeB() 。

55.填空题 [NOIP 普及组 2010]（哥德巴赫猜想）哥德巴赫猜想是指，任一大于 2 的偶数都可写成两个质数之和。迄今 为止， 这仍然是一个著名的世界难题，被誉为数学王冠上的明珠。试编写程序，验证任一大 于 2 且不超过 n 的偶数都能写成两个质数之和。

```
#include<iostream>
using namespace std;
int main() {
    const int SIZE = 1000;
    int n, r, p[SIZE], i, j, k, ans;
    bool tmp;
    cin>>n;
    r = 1;
    p[1] = 2;
    for (i = 3; i <= n; i++) {
            ①    _______;
        for (j = 1; j <= r; j++)
```

```
                if (i %   ② ______) {
                    tmp = false;
                    break;
                }
            if (tmp) {
                r++;
                    ③_________

            }
        }
        ans = 0;
        for (i = 2; i <= n / 2; i++) {
            tmp = false;
            for (j = 1; j <= r; j++)
                for (k = j; k <= r; k++)
                    if (i + i ==   ④ _______    ) {
                        tmp = true;
                        break;
                    }
            if (tmp)
                ans++;
        }
        cout<<ans<<endl;
        return 0;
}
```

若输入 n 为 2010 ，则输出 ⑤ _______ 时表示验证成功，即大于 2 且不超过 2010 的偶数都 满足哥德巴赫猜想。

56.填空题 [NOIP 普及组 2005] 判断质数

题目描述: 给出一个正整数，判断这个数是否是质数。

输入: 一个正整数 n(1 ≤ n ≤ 10000)。

输出: 如果 n 是质数，输出”YES”；否则，输出”NO”。

输入样例: 10

输出样例: NO

程序:

```
#include <stdio.h>

int main() {
```

```
    int ①;
    scanf(""%d"", &n);
    if (n == 2) puts(②);
    else if ( ③ || n % 2 == 0) puts(""NO"");
    else {
        i = 3;
        while (i * i <= n) {
            if (④) {
                puts(""NO"");
                return 0;
            }
            i = i + 2;
        }
        puts(""YES"");
    }
    return 0;
}
```

57.单选题 [GESP202406【1 级】] 下面 C++代码用于判断 N 是否为质数（只能被 1 和它本身整除的正整数）。程序执行后，下面有关描述正确的是（ ）。

```
int N;
cout << "请输入整数：";
cin >> N;
bool Flag = false;
if (N >= 2) {
    Flag = true;
    for (int i = 2; i < N; i++) {
        if (N % i == 0) {
            Flag = false;
            break;
        }
    }
}
if (Flag)
    cout << "是质数";
else
    cout << "不是质数";
```

A. 如果输入负整数，可能输出“是质数”

B. 如果输入 2，将输出“不是质数”，因为此时循环不起作用

C. 如果输入 2，将输出“是质数”，即便此时循环体没有被执行

D. 如果将 if (N >= 2) 改为 if (N > 2) 将能正确判断 N 是否质数

58.单选题 [GESP202406【2 级】] 执行下面的 C++代码，有关说法正确的是（ ）【质数是指仅能被 1 和它本身整除的正整数】。

```
int N;
cin >> N;
bool Flag = true;
for (int i = 2; i < N; i++) {
    if (i * i > N)
        break;
    if (N % i == 0) {
        Flag = false;
        break;
    }
}
if (Flag)
    cout << N << "是质数" << endl;
else
    cout << N << "不是质数" << endl;
```

A. 如果输入正整数，上面代码能正确判断 N 是否为质数

B. 如果输入整数，上面代码能正确判断 N 是否为质数

C. 如果输入大于等于 0 的整数，上面代码能正确判断 N 是否质数

D. 如将 Flag = true 修改为 Flag = N>=2? true:false 则能判断所有整数包括负整数、0、正整数是否为质数

5. 质数筛法

59.判断题 [GESP 样题【6 级】/GESP202309【5 级】] 质数的判定和筛法的目的并不相同，质数判定旨在判断特定的正整数是否为质数，而质数筛法意在筛选出范围内的所有质数。

60.判断题 [GESP202309【5 级】] 找出自然数 N 以内的所有质数，常用算法有埃氏筛法和线性筛法，其中埃氏筛法效率更高。

61.单选题 [GESP202403【6 级】] 线性筛法与埃氏筛法相比的优势是（ ）。

A. 更容易实现

B. 更节省内存

C. 更快速

D. 更准确

62.单选题 [GESP202312【5 级】/GESP202312【6 级】] 小杨想编写一个判断任意输入的整数 N 是否为素数的程序，下面哪个方法不合适？（ ）

A. 埃氏筛法

B. 线性筛法

C. 二分答案

D. 枚举法

63.判断题 [GESP 样题【5 级】] 埃氏筛法用于找出自然数 N 以内的所有质数，其时间复杂度为 O（N*sqrt（N））， 因为判定一个数是否为质数的时间复杂度为 O(N)。

64.判断题 [GESP202403【5 级】] 素数表的埃氏筛法和线性筛法的时间复杂度都是 O(NloglogN)。

65.判断题 [GESP202406【5 级】] 找出自然数 n 以内的所有质数，常用算法有埃拉托斯特尼（埃氏）筛法和线性筛法，其中埃氏筛法效率更高。

66.单选题 [GESP202403【5 级】] 素数的线性筛法时间复杂度为（ ）。

A.O(n）

B.O(nloglogn)

C.O（nlogn）

D.O(n^2)

67.单选题 [GESP202403【8 级】] 下面程序的时间复杂度为（ ）。

```
int primes[MAXP],num=0;
bool isPrime[MAXN]={false};
void sieve(){
    for(intn=2;n<=MAXN; n++){
        if(!isPrime[n])
            primes[num++]= n;
        for(int i=0;i<num &&n*primes[i]<= MAXN; i++){
            isPrime[n*primes[i]]= true;
            if(n%primes[i]==0)
                break;
        }
    }
}
```

A.O(n)

B.O(n*logn)

C.O(n*loglogn)

D.O(n^2)

68.单选题 [GESP202403【5 级】] 在埃拉托斯特尼筛法中，要筛选出不大于 n 的所有素数，最外层循环应该遍历什么范围（ ）？

```
vector<int> sieveOfEratosthenes(int n) {
    std::vector<bool> isPrime(n + 1, true);
    std::vector<int> primes;
    _____________{
        if (isPrime[i]) {
            primes.push_back(i);
            for (int j = i * i; j <= n; j += i) {
                isPrime[j] = false;
            }
        }
    }
    for(int i=sqrt(n)+1;i<=n;i++){
        if(isPrime[i])
            primes.push_back(i);
    }
    return primes;
}
```

A. for (int i = 2; i <= n; ++i)

B. for (int i = 1; i < n; ++i)

C. for (int i = 2; i <= sqrt(n); ++i)

D. for (int i = 1; i <= sqrt(n); ++i)

69.填空题 [NOIP 普及组 2007] #include <iostream.h>

```
#include <iomanip.h>
#include ""math.h""
void main()
{
    int a1[51]={0};
    int i,j,t,t2,n=50;
    for (i=2;i<=sqrt(n);i++)
    if(a1[i]==0)
    {
        t2=n/i;
        for(j=2;j<=t2;j++) a1[i*j]=1;
    }
    t=0;
    for (i=2;i<=n;i++)
    if(a1[i]==0)
    {
        cout<<setw(4)<<i; t++;
        if(t%10==0) cout<<endl;
    }
    cout<<endl;
}
```

输出： _____

70.填空题 [NOIP 提高组 2007]

```
#include <iostream.h>
#include <iomanip.h>
#include ""math.h""
void main()
{
    int a1[51]={0};
    int i,j,t,t2,n=50;
    for (i=2;i<=sqrt(n);i++)
    if(a1[i]==0)
    {
        t2=n/i;
        for(j=2;j<=t2;j++) a1[i*j]=1;
```

```
    }
    t=0;
    for (i=2;i<=n;i++)
    if(a1[i]==0)
    {
        cout<<setw(4)<<i; t++;
        if(t%10==0) cout<<endl;
    }
    cout<<endl;
}
```

输出：_____

71.填空题 [NOIP 提高组 2018]

```
#include <cstdio>
int n, d[100];
bool v[100];
int main() {
    scanf(""%d"", &n);
    for (int i = 0; i < n; ++i) {
        scanf(""%d"", d + i);
        v[i] = false;
    }
    int cnt = 0;
    for (int i = 0; i < n; ++i) {
        if (!v[i]) {
            for (int j = i; !v[j]; j = d[j]) {
                v[j] = true;
            }
            ++cnt;
        }
    }
    printf(""%d"", cnt);
    return 0;
}
```

输入：10 7 1 4 3 2 5 9 8 0 6

输出：（ ）

72.单选题 [CSP-S2023]

```
#include <iostream>
```

```cpp
#include <cmath>
#include <vector>
#include <algorithm>
using namespace std;

long long solve1(int n) {
    vector<bool> p(n + 1, true);
    vector<long long> f(n + 1, 0), g(n + 1, 0);
    f[1] = 1;
    for (int i = 2; i * i <= n; i++) {
        if (p[i]) {
            vector<int> d;
            for (int k = i; k <= n; k *= i)
                d.push_back(k);
            reverse(d.begin(), d.end());
            for (int k : d) {
                for (int j = k; j <= n; j += k) {
                    if (p[j]) {
                        p[j] = false;
                        f[j] = i;
                        g[j] = k;
                    }
                }
            }
        }
    }
    for (int i = sqrt(n) + 1; i <= n; i++) {
        if (p[i]) {
            f[i] = i;
            g[i] = i;
        }
    }
    long long sum = 1;
    for (int i = 2; i <= n; i++) {
        f[i] = f[i / g[i]] * (g[i] * f[i] - 1) / (f[i] - 1);
        sum += f[i];
    }
    return sum;
}

long long solve2(int n) {
    long long sum = 0;
    for (int i = 1; i <= n; i++) {
        sum += i * (n / i);
    }
```

```
47          return sum;
48      }
49
50      int main() {
51          int n;
52          cin >> n;
53          cout << solve1(n) << endl;
54          cout << solve2(n) << endl;
55          return 0;
56      }
```

假设输入的 n 是不超过 1000000 的自然数，完成下面的判断题和单选题:

判断题

将第行删去，输出不变。( )

当输入为 10 时，输出的第一行大于第二行。( )

当输入为 1000 时，输出的第一行与第二行相等。( )

单选题

solve1(n) 的时间复杂度为( )。

solve2(n) 的时间复杂度为( )。

当输入为 5 时，输出的第二行为( )。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. $O(n(\log n)^2)$

B. $O(n)$

C. $O(n\log n)$

D. O(nloglogn)

5.

A. $O(n^2)$

B. O(n)

C. O(nlogn)

D. $O(n\sqrt{n})$

6.

A. 20

B. 21

C. 22

D. 23

73.单选题 [CSP-J2021]

```
#include <iostream>
using namespace std;
const int n = 1000000;
const int N = n + 1;

int m;
int a[N], b[N], c[N], d[N];
int f[N], g[N];

void init()
{
    f[1] = g[1] = 1;
    for (int i = 2; i <= n; i++) {
        if (!a[i]) {
            b[m++] = i;
            c[i] = 1, f[i] = 2;
            d[i] = 1, g[i] = i + 1;
        }
        for (int j = 0; j < m && b[j] * i <= n; j++) {
            int k = b[j];
            a[i * k] = 1;
            if (i % k == 0) {
                c[i * k] = c[i] + 1;
                f[i * k] = f[i] / c[i * k] * (c[i * k] + 1);
                d[i * k] = d[i];
```

```
                    break;
                } else {
                    c[i * k] = 1;
                    f[i * k] = f[i] * 2;
                    d[i * k] = d[i] * g[k];
                }
            }
        }
}

int main()
{
     init();
    int x;
    cin>>x;
    cout<<f[x]<<' '<<g[x]<<endl;
    return 0;
}
```

假设输入的 x 是不超过 1000 的自然数，完成下面的判断题和单选题：

判断题

若输入不为 1，把第 13 行删去不会影响输出的结果。（ ）

第 25 行的 f[i] / c[i * k]可能存在无法整除而向下取整的情况。 （ ）

在执行完 init() 后，f 数组不是单调递增的，但 g 数组是单调递增的。 （ ）

单选题

init 函数的时间复杂度为（ ）。

在执行完 init() 后，f[1],f[2],f[3]…f[100]中有（）个等于 2。

当输入为 1000 时，输出为（）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. O(n)

B. O(nlogn)

C. O(√n)

D. O(n^2)

5.

A. 23

B. 24

C. 25

D. 26

6.

A. 15 1340

B. 15 2340

C. 16 2340

D. 16 1340

74.填空题 [NOIP 普及组 2014]

```
#include <iostream>
using namespace std;
const int SIZE = 100;
int main()
{
    int p[SIZE];
    int n, tot, i, cn;
    tot = 0;
    cin >> n;
    for ( i = 1; i <= n; i++ )
        p[i] = 1;
    for ( i = 2; i <= n; i++ )
    {
        if ( p[i] == 1 )
            tot++;
        cn = i * 2;
        while ( cn <= n )
```

```
        {
            p[cn] = 0;
            cn += i;
        }
    }
    cout << tot << endl;
    return(0);
}
```

输入：30

输出：___

75.单选题 [GESP202406【5 级】] 下述代码实现素数表的线性筛法，筛选出所有小于等于 n 的素数，则横线上应填的代码是( )。

```
vector<int> linear_sieve(int n) {
    vector<bool> is_prime(n + 1, true);
    vector<int> primes;
    is_prime[0] = is_prime[1] = 0; // 0 和 1 两个数特殊处理
    for (int i = 2; i <= n; ++i) {
        if (is_prime[i]) {
            primes.push_back(i);
        }
        _____ { // 在此处填入代码
            is_prime[i * primes[j]] = 0;
            if (i % primes[j] == 0)
                break;
        }
    }
    return primes;
}
```

A. for (int j = 0; j < primes.size() && i * primes[j] <= n; j++)

B. for (int j = 0; j <= sqrt(n) && i * primes[j] <= n; j++)

C. for (int j = 0; j <= n; j++)

D. for (int j = 1; j <= sqrt(n); j++)

76.单选题 [GESP202406【5 级】] 下述代码实现素数表的线性筛法，筛选出所有小于等于 n 的素数，时间复杂度是（ ）。

A.O(n^2)

B.O(nlogn)

C.O(nloglogn)

D.O(n)

```
vector<int> linear_sieve(int n) {
    vector<bool> is_prime(n + 1, true);
    vector<int> primes;
    is_prime[0] = is_prime[1] = 0; // 0 和 1 两个数特殊处理
    for (int i = 2; i <= n; ++i) {
        if (is_prime[i]) {
            primes.push_back(i);
        }
        for (int j = 0; j < primes.size() && i * primes[j] <= n; j++) {
            is_prime[i * primes[j]] = 0;
        }
        if (i % primes[j] == 0)
            break;
    }
    return primes;
}
```

77.单选题 [GESP202406【8 级】] 下面程序的时间复杂度为（ ）。

```
bool notPrime[N] = {false};
void sieve() {
    for (int n = 2; n * n < N; n++)
        if (!notPrime[n])
            for (int i = n * n; i < N; i += n)
                notPrime[i] = true;
}
```

A.O(N)

B.O(NlogN)

C.O(NloglogN)

D.O(N^2)

答案

1.C

2.B

3.C

4.C

5.A

6.A

7.289

8.1009 1008

9.C

10.782

11.-400

12.

1. 正确答案： offset = 4
2. 正确答案： (offset + dayNum[i]) % 7
3. 正确答案： dayNum[m]
4. 正确答案： i
5. 正确答案： (offset + i) % 7"

13.

① begin+m-1

② result>=k (或者 k<=result)

③ !find (或者 find==0)

④ 2*k-i

⑤ m-1

14.DCCDB

15.416

16.5850

17.T

18.D

19.C

20.A

21.B

22.AAABDC

23.ABCDA

24.C

25.C

26.T

27.C

28.C

29.D

30.C

31.A

32.4

33.3

34.16

35.A

36.

1. 正确答案： i * i
2. 正确答案： n / i
3. 正确答案： return a
4. 正确答案： a % b
5. 正确答案： ans + gcd(a[i], a[j])

37.B

38.B

39.B

40.B

41.F

42.T

43.F

44.T

45.B

46.CCCAC

47.B

48.C

49.B

50.D

51.A

52.D

53.B

54.B

55.

1.正确答案： tmp=1 / tmp=true

2.正确答案： p[j]

3.正确答案： p[r]=i

4.正确答案： p[j]+p[k] / p[k]+p[j]

5.正确答案： 1004

56.

① n,i(或者 i, n)

② 'YES'

③n==1(或者 n-1 == 0)

④n%i==0(或者!n%i)

57.C

58.D

59.T

60.F

61.C

62.C

63.F

64.F

65.F

66.A

67.A

68.C

69.

2 3 5 7 11 13 17 19 23 29

31 37 41 43 47

70.

2 3 5 7 11 13 17 19 23 29

31 37 41 43 47

71.6

72.BBADBB

73.ABBACC

74.10

75.A

76.D

77.C

GW 信奥 CSP 初赛习题集 5-3

计算机数学——数制

1. 进制转换

1. 基本概念

1.单选题 [NOIP 普及组 2013] 在十六进制表示法中，字母 A 相当于十进制中的（ ）。

A. 9

B. 10

C. 15

D. 16

2.单选题 [GESP 样题【3 级】] 下列关于十六进制的描述中，正确的是( )

A. 使用 0-9 和 A-F 表示

B. 使用 0-9 和 A-E 表示

C. 使用 1-9 和 A-F 表示

D. 使用 1-9 和 A-E 表示

3.单选题 [GESP202309【3 级】] 下列关于进制的叙述，正确的是（ ）。

A. 只有十进制和二进制能够用来表示小数，八进制和十六进制不可以。

B. 常用的进制包括二进制、八进制、十进制、十六进制，其他进制在日常生活中很少使用。

C. 对任意正整数，其二进制表示不会比它的十进制表示更短。

D. 正整数的八进制表示中，每一位可能出现的最大数字是 8。

4.单选题 [GESP202306【3 级】] 下列关于进制的叙述，不正确的是（ ）。

A. 正整数的二进制表示中只会出现 0 和 1。

B. 10 不是 2 的整数次幂，所以十进制数无法转换为二进制数。

C. 从二进制转换为 8 进制时，可以很方便地由低到高将每 3 位二进制位转换为对应的一位 8 进制位。

D. 从二进制转换为 16 进制时，可以很方便地由低到高将每 4 位二进制位转换为对应的一位 16 进制位。

2. 整数互转

5.单选题 [GESP 样题【3 级】] 对于一个十进制数 37，以下哪个是它的二进制表示( )

A. 10101

B. 100101

C. 101001

D. 1000101

6.单选题 [NOIP 普及组 2005] 和十进制数 23 的值相等的二进制数是（ ）。

A. 10110

B. 11011

C. 11011

D. 10111

E. 10011

7.单选题 [NOIP 普及组 2006/NOIP 普及组 2007] 与十进制数 1770 对应的八进制数是（ ）。

A. 3350

B. 3351

C. 3352

D. 3540

8.单选题 [NOIP 普及组 2004] 十进制数 2004 等值于八进制数（ ）。

A. 3077

B. 3724

C. 2766

D. 4002

E. 3755

9.单选题 [CSP-J2020] 二进制数 1011 转换成十进制数是（ ）。

A. 11

B. 10

C. 13

D. 12

10.单选题 [CSP-S2020] 请选出以下最大的数（ ）

A. (550)10

B. (777)8

C. 2^{10}

D. (22F)16

11.单选题 [NOIP 普及组 2018/NOIP 提高组 2018] 下列四个不同进制的数中，与其它三项数值上不相等的是（ ）。

A. (269)16

B. (617)10

C. (1151)8

D. (1001101011)2

12.单选题 [NOIP 提高组 2015] 下面有四个数据组，每个组各有三个数据，其中第一个数据为八进制数，第二个数据为 十进制数，第三个数据为十六进制数。这四个数据组中三个数据相同的是（ ）。

A. 120 82 50

B. 144 100 68

C. 300 200 C8

D. 1762 1010 3F2

13.单选题 [NOIP 普及组 2012] 十六进制数 9A 在（ ）进制下是 232

A. 四

B. 八

C. 十

D. 十二

14.单选题 [GESP202312【3 级】] 在下列编码中，不能够和二进制 1101 1101 相等的是( )。

A. （221）10 进制

B. （335）8 进制

C. （dd）16 进制

D. （5d）16 进制

15.单选题 [GESP202406【8 级】] 7 进制数 235 转换成 3 进制数是（ ）。

A. 11121

B. 11122

C. 11211

D. 11112

3. 整数运算

16.单选题 [NOIP 普及组 2011/NOIP 提高组 2011] 在二进制下，1101001 +（ ）= 1110110。

A. 1011

B. 1101

C. 1010

D. 1111

17.单选题 [NOIP 普及组 2014/NOIP 提高组 2014] 二进制数 00100100 和 00010101 的和是 ( )。

A. 00101000

B. 001010100

C. 01000101

D. 00111001

18.单选题 [NOIP 普及组 2015] 二进制数 00100100 和 00010100 的和是 ( )

A. 00101000

B. 01000001

C. 01000100

D. 00111000

19.单选题 [NOIP 普及组 2016] 二进制数 00101100 和 00010101 的和是 ( )。

A. 00101000

B. 01000001

C. 01000100

D. 00111000

20.单选题 [CSP-S2021] 二进制数（00101010）2 和（00010110）2 的和为 ( )。

A. 00111100

B. 01000000

C. 00111100

D. 01000010

21.单选题 [CSP-J2023] 八进制（12345670）8 和（07654321）8 的和为

A.222222218

B.211111118

C.221111118

D.222222118

22.不定项选择题 [NOIP 提高组 2005/NOIP 普及组 2005] (3725)8 + (B)16 的运算结果是（ ）。

A. (3736)8

B. (2016)10

C. (11111100000)2

D. (3006)10

E. (7E0)16

23.不定项选择题 [NOIP 提高组 2006/NOIP 普及组 2006] (2010)16 + (32)8 的结果是（ ）。

A. (8234)10

B. (202A)16

C. (100000000110)2

D. (2042)16

24.不定项选择题 [NOIP 提高组 2007/NOIP 普及组 2007] (2070)16 + (34)8 的结果是（ ）。

A. (8332)10

B. (208C)16

C. (100000000110)2

D. (20214)8

25.单选题 [CSP-J2023] 数(101010)2 和(166)8 的和为（ ）

A.(10110000)2

B.(236)8

C.(158)10

D.(A0)16

26.不定项选择题 [NOIP 提高组 2004/NOIP 普及组 2004] (2004)10 + (32)16 的结果是（ ）。

A. (2036)16

B. (2054)10

C. (4006)8

D. (100000000110)2

E. (2036)10

27.不定项选择题 [NOIP 提高组 2008/NOIP 普及组 2008] (2008)10 + (5B)16 的结果是（ ）。

A. (833)16

B. (2099)10

C. (4063)8

D. (100001100011)2

4. 小数转换

28.单选题 [NOIP 提高组 2005] 以下二进制数的值与十进制数 23.456 的值最接近的是（ ）。

A. 10111.0101

B. 11011.1111

C. 11011.0111

D. 10111.0111

E. 10111.1111

29.判断题 [GESP202309【3 级】] 二进制数 101.101 在十进制下是 5.005。

30.单选题 [GESP202306【3 级】/NOIP 普及组 2008/NOIP 提高组 2013] 二进制数 11.01 在十进制下是（ ）。

A. 3.01

B. 3.05

C. 3.125

D. 3.25

31.单选题 [NOIP 提高组 2014] 二进制数 111.101 所对应的十进制数是 ( ).

A.5.625

B.5.5

C.6.125

D.7.625

32.单选题 [CSP-J2021/GESP202406【3 级】] 二进制数 101.11 对应的十进制数是（ ）。

A. 6.5

B. 5.5

C. 5.75

D. 5.25

33.单选题 [GESP202406【8 级】] 二进制数 100.001 转换成十进制数是（ ）。

A. 4.25

B. 4.125

C. 4.5

D. 4.75

34.单选题 [CSP-J2022] 八进制数 32.1 对应的十进制数是（ ）。

A.24.125

B.24.250

C.26.125

D.26.250

35.单选题 [NOIP 提高组 2010] 与 16 进制数 A1.2 等值的 10 进制数是（ ）

A. 101.2

B. 111.4

C. 161.125

D. 177.25

36.单选题 [NOIP 普及组 2015/NOIP 提高组 2015] 与二进制小数 0 .1 相等的十六进制数是（ ）

A. 0.8

B. 0.4

C. 0.2

D. 0.1

37.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 与二进制小数 0.1 相等的八进制数是 ( )。

A. 0.8

B. 0.4

C. 0.2

D. 0.1

38.单选题 [NOIP 普及组 2017] 十进制小数 13.375 对应的二进制数是( )。

A. 1101.011

B. 1011.011

C. 1101.101

D. 1010.01

39.单选题 [NOIP 提高组 2004] 十进制数 100.625 等值于二进制数（ ）。

A. 1001100.101

B. 1100100.101

C. 1100100.011

D. 1001100.11

E. 1001100.01

40.单选题 [NOIP 提高组 2006] 与十进制数 1770.625 对应的八进制数是（ ）。

A. 3352.5

B. 3350.5

C. 3352.1161

D. 3350.1151

E. 前 4 个答案都不对

41.单选题 [NOIP 提高组 2007] 与十进制数 17.5625 对应的 8 进制数是（ ）。

A. 21.5625

B. 21.44

C. 21.73

D. 21.731

E. 前 4 个答案都不对

42.单选题 [NOIP 普及组 2009] 十进制小数 125.125 对应的 8 进制数是

A. 100.1

B. 175.175

C. 175.1

D. 100.175

43.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 与十进制数 28.5625 相等的四进制数是（ ）。

A. 123.21

B. 131.22

C. 130.22

D. 130.21

5. 综合应用

44.单选题 [NOIP 普及组 2016] 如果 256 种颜色用二进制编码来表示，至少需要( )位。

A. 6

B. 7

C. 8

D. 9

45.单选题 [NOIP 普及组 2011] 一个正整数在二进制下有 100 位，则它在十六进制下有（ ）位。

A. 7

B. 13

C. 25

D. 不能确定

46.不定项选择题 [NOIP 提高组 2011] 一个正整数在十六进制下有 100 位，则它在二进制下可能有（ ）位。

A. 399

B. 400

C. 401

D. 404

47.单选题 [NOIP 普及组 2010] 一个自然数在十进制下有 n 位，则它在二进制下的位数与（ ）最接近。

A. 5n

B. $n*\log_2 10$

C. $10*\log_2 n$

D. $10^n*\log_2 n$

48.单选题 [NOIP 普及组 2010] 设 X、Y、Z 分别代表三进制下的一位数字，若等式 XY + ZX = XYX 在三进制下成立， 那么同样在三进制下，等式 XY * ZX = （ ）也成立。

A. YXZ

B. ZXY

C. XYZ

D. XZY

49.单选题 [NOIP 提高组 2010] 如果在某个进制下等式 7*7=41 成立，那么在该进制下等式 12*12= （ ）也成立。

A. 100

B. 144

C. 164

D. 196

50.单选题 [NOIP 普及组 2014] 下列各无符号十进制整数中，能用八位二进制表示的数中最大的是（）。

A. 296

B. 133

C. 256

D. 199

51.不定项选择题 [NOIP 提高组 2014] 下列各无符号十进制整数中，能用八位二进制表示的数有（).

A.296

B.133

C.256

D.199

2. 整数编码

52.判断题 [GESP202306【3 级】] 数据编码方式只有原码、反码、补码三种。

53.判断题 [GESP202403【3 级】] 任意整数 a 的二进制反码与补码都有 1 位不同。

54.判断题 [GESP 样题【3 级】] 二进制数据编码中，负数的补码是通过对原码按位取反并加 1 得到的。

55.不定项选择题 [NOIP 提高组 2010] 在整数的补码表示法中,以下说法正确的是( )

A. 只有负整数的编码最高为 1

B. 在编码的位数确定后，所能表示的最小整数和最大整数的绝对值相同

C. 整数 0 只有唯一的一个编码

D. 两个用补码表示的数相加时，如果在最高位产生进位，则表示运算溢出

56.单选题 [GESP 样题【3 级】] 下列关于负数的原码、反码、补码的描述中，正确的是( )

A. 原码和反码互为按位取反（符号位除外），补码为反码加 1

B. 原码和反码互为按位取反（符号位除外），补码为原码加 1

C. 反码和补码互为按位取反（符号位除外），原码为反码加 1

D. 补码和原码互为按位取反（符号位除外），反码为补码加 1

57.判断题 [GESP202406【3 级】] 补码的优点是可以将减法运算转化为加法运算，从而简化计算机的硬件设计。

58.判断题 [GESP202406【3 级】] 整数-6 的 16 位补码可用十六进制表示为 FFFA。

59.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 在 8 位二进制补码中，10101011 表示的数是十进制下的( )。

A. 43

B. -85

C. -43

D. -84

60.单选题 [NOIP 提高组 2009] 在字长为 16 位的系统环境下，一个 16 位带符号整数的二进制补码为 1111111111101101。其对应的十进制整数应该是：

A. 19

B. -19

C. 18

D. -18

61.单选题 [NOIP 普及组 2010] 一个字长为 8 位的整数的补码是 11111001 ，则它的原码是（ ）

A. 00000111

B. 01111001

C. 11111001

D. 10000111

62.单选题 [GESP202403【3 级】] 如果 16 位短整数 -2 的二进制是""FFFE""，则短整数 -4 的十六进制是( )。

A. FF04

B. FFFA

C. FFFC

D. FFFH

63.单选题 [GESP202403【3 级】] 整数-5 的 16 位补码表示是( )。

A. 1005

B. 1006

C. FFFA

D. FFFB

3. 位运算

64.判断题 [GESP 样题【4 级】] &和&&都是 C++语言的运算符，*和**也都是。

65.判断题 [GESP202306【4 级】] >=和>>=都是 C++语言的运算符。

66.判断题 [GESP202309【3 级】] 在 C++语言中，位运算符也有类似“先乘除、后加减”的优先级规则。因此，使用时应注意合理使用括号。

67.判断题 [GESP 样题【3 级】] C++语言中数字的符号位是不参与位运算的。

68.单选题 [GESP 样题【3 级】] 以下哪个属于 C++语言中的位运算符？( )

A. +

B. -

C. *

D. &

69.单选题 [GESP202306【3 级】] 以下哪个不是 C++语言中的运算符？（ ）

A. &

B. &&

C. *

D. **

70.单选题 [GESP202309【3 级】] 以下哪个不是 C++语言中的运算符？（ ）

A. ~

B. ~~

C. <

D. <<

71.单选题 [GESP202406【7 级】] 在 C++中，关于运算符&，下面说法正确的是（ ）。

A. 2 & 3 的结果是 true

B. 011 & 111 的结果是 3

C. 3 & 6 的结果是 2

D. 110 & 101 的结果是 4

72.单选题 [CSP-S2019] 下面哪个选项是 11 1011 1001 0111 和 01 0110 1110 1011 进行逻辑或运算的结果( )。

A. 11 1111 1101 1111

B. 11 1111 1111 1101

C. 10 1111 1111 1111

D. 11 1111 1111 1111

73.单选题 [CSP-J2019] 二进制数11 1011 1001 0111和01 0110 1110 1011 进行按位与运算的结果是（）。

编者注：原题为“逻辑与”，但是根据题意应当是按位与。

A.01 0010 1000 1011

B.01 0010 1001 0011

C.01 0010 1000 0001

D.01 0010 1000 0011

74.单选题 [NOIP 提高组 2016] 二进制数 00101100 和 01010101 异或的结果是（ ）。

A. 00101000

B. 01111001

C. 01000100

D. 01000100

75.判断题 [GESP202403【4 级】] C++语言中 cout << 9^2 << endl; 会输出 81。

76.判断题 [GESP202403【7 级】] C++语言中，表达式 2 ^ 3 的结果类型为 int 、值为 8 。

77.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 在 C++ 中，表达式 21^2 的值是（ ）

A. 441

B. 42

C.23

D.24

78.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] 在 C++程序中，表达式 23|2^5 的值是（ ）

A. 23

B. 1

C.32

D.18

79.单选题 [NOIP 普及组 2008] 在 C++程序中，表达式 200|10 的值是（ ）

A. 20

B. 1

C. 220

D. 202

80.单选题 [GESP202306【4 级】] 如果 a 为 int 类型的变量，且 a 的值为 6，则执行 a &= 3;之后，a 的值会是（ ）。

A. 3

B. 9

C. 2

D. 7

81.单选题 [GESP202309【4 级】] 如果 a 为 int 类型的变量，且 a 的值为 6，则执行 a = ~a; 之后， a 的值会是（ ）。

A. -6

B. 6

C. -7

D. 7

82.判断题 [GESP202406【3 级】] 如果 为 int 类型的变量，且表达式 ((a | 3) == 3) 的值为 true ，则说明 在从 0 到 3 之间（可能为 0、可能为 3）。

83.判断题 [GESP202406【7 级】] C++语言中，表达式 6 & 5 的结果类型为 int 、值为 1 。

84.单选题 [GESP202403【3 级】] 下面 C++代码执行后的输出是( )。

```
int main(){
    cout<<(3|16)<<endl;
```

```
    cout<<endl;
    return 0;
}
```

A. 3

B. 16

C. 19

D. 48

85.判断题 [GESP202403【4 级】] 定义变量 int a=5，则 cout << &++a 会输出 6 。

86.单选题 [GESP202312【4 级】] 下列 C++语句执行以后结果是 true 的是（ ）。

A. 3&&false

B. 5&&2

C. 101&&000

D. 4&true

87.单选题 [GESP202403【3 级】] 定义整型变量 int a=3, b=16 ，则 a|b 的值和 a+b 的关系是（ ）。

A. 大于

B. 等于

C. 小于

D. 等于或小于

88.单选题 [CSP-S2023] 假设我们有以下的 C++ 代码：

```
int a =5,b=3,c=4;
bool res=a&b||c^b&&a|c;
```

请问，res 的值是什么？（） 提示：在 C++ 中，逻辑运算的优先级从高到低依次为：逻辑非（!）、逻辑与（&&）、逻辑或（||）。位运算的优先级从高到低依次为：位非（~）、

位与（&）、位异或（^）、位或（|）。同时，双目位运算的优先级高于双目逻辑运算；逻辑非与位非优先级相同，且高于所有双目运算符。

A. true

B. false

C. 1

D. 0

89.判断题 [GESP202403【3 级】] 对整型变量 int a = 3 ，执行 C++代码 a<<2 将把 2 输出到 a 中。

90.判断题 [GESP 样题【3 级】] 表达式(7 >> 2)的计算结果为 1.75，且结果类型为 double。

91.判断题 [GESP202309【3 级】/GESP 样题【8 级】] 在 C++语言中，所有 int 类型的值，经过若干次左移操作（ << ）后，它们的值总会变为 0 。

92.判断题 [GESP202403【3 级】] 在 C++语言中， (010<<1) 执行结果是 100 。

93.判断题 [GESP202403【3 级】] 一个 int 类型变量 a ，执行操作 (a<<2>>2) 后的值一定是 a 。

94.单选题 [GESP202309【3 级】] 如果 a 是 int 类型的变量，下列哪个表达式的值一定为 true ？（ ）

A. a + 1000 - 1000 == a

B. a * 2 / 2 == a

C. (a & 1) == 1

D. (a | 1) == a + 1

95.单选题 [GESP202403【3 级】] 定义整数 int x=-5 ，则执行 C++代码 cout << (x == (x<<1>>1)) 输出是（ ）。

A. 0

B. 1

C. -5

D. 5

96.阅读题 [CSP-S2023]

```
#include <iostream>
using namespace std;
unsigned short f(unsigned short x) {
    x ^= x << 6;
    x ^= x >> 8;
    return x;
}
int main() {
    unsigned short x;
    cin >> x;
    unsigned short y = f(x);
    cout << y << endl;
    return 0;
}
```

假设输入的 x 是不超过的自然数，完成下面的判断题和单选题:

判断题

当输入非零时，输出一定不为零。( )

将 f 函数的输入参数的类型改为 unsigned int，程序的输出不变。( )

当输入为 65535 时，输出为 63。( )

当输入为 1 时，输出为 64。( )

单选题

当输入为 512 时，输出为（ ）。

当输入为 64 时，执行完第 5 行后 x 的值为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 33280

B. 33410

C. 33106

D. 33346

6.

A. 8256

B. 4130

C. 4128

D. 4160

97.单选题 [CSP-J2022]

```
#include<iostream>
using namespace std;
int main()
{
    unsigned short x, y;
    cin >> x >> y;
    x = (x | x << 2) & 0x33;
    x = (x | x << 1) & 0x55;
    y = (y | y << 2) & 0x33;
    y = (y | y << 1) & 0x55;
```

```
11      unsigned short z = x | y << 1;
12      cout << z << endl;
13      return 0;
14  }
```

假设输入的 x,y 均是不超过 15 的自然数，完成下面的判断题和单选题

判断题

1.删去第 7 行与第 13 行的 unsigned，程序行为不变,

2.将第 7 行与第 13 行的 short 均改为 char，程序行为不变。

3.程序总是输出一个整数“0”

4.当输入为 2 2 时，输出为 10

5.当输入为 2 2 时，输出为 59

6.当输入为 13 8 时，输出为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 正确

B. 错误

6.

A.0

B.209

C.197

D.226

98.填空题 [CSP-J2021]

```
#include <iostream>
using namespace std;

int n;
int a[1000];

int f(int x)
{
    int ret = 0;
    for (; x; x &= x - 1) ret++;
    return ret;
}

int g(int x)
{
    return x & -x;
}

int main()
{
    cin >> n;
    for (int i = 0; i < n; i++) cin >> a[i];
    for (int i = 0; i < n; i++)
        cout << f(a[i]) << " " << g(a[i]) << " ";
    cout << endl;
    return 0;
}
```

输入的 n 等于 1001 时，程序不会发生下标越界。（ ）

输入的 a[i]必须全为正整数，否则程序将陷入死循环。（ ）

当输入为 5 2 11 9 16 10 时，输出为 3 4 3 17 5。（ ）

当输入为 1 511998 时，输出为 18。（ ）

将源代码中 g 函数的定义（14~17 行）移到 main 函数的后面，程序可以正常编译运行。（ ）

单选题

当输入为 2 -65536 2147483647 时，输出为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 正确

B. 错误

6.

A. 65532 33

B. 65552 32

C. 65535 34

D. 65554 33

99.单选题 [CSP-S2020]

```
#include <iostream>
using namespace std;
int n;
int d[1000];
int main() {
    cin >> n;
    for (int i = 0; i < n; ++i)
```

```
8       cin >> d[i];
9     int ans = -1;
10    for (int i = 0; i < n; ++i)
11      for (int j = 0; j < n; ++j)
12        if (d[i] < d[j])
13            ans = max(ans, d[i] + d[j] - (d[i] & d[j]));
14      cout << ans;
15      return 0;
16  }
```

假设输入的 n 和 d[i]都是不超过 10000 的正整数，完成下面的判断题和单选题：

判断题

n 必须小于 1000,否则程序可能会发生运行错误。（ ）

输出一定大于等于 0。（ ）

若将第 13 行的 j=0 改为 j = i + 1 程序输出可能会改变。 （ ）

将第 14 行的 d[i] < d[j] 改为 d[i] != d[j]，程序输出不会改变。（ ）

单选题

若输入 n 为 100，且输出为 127，则输入的 d[i]中不可能有（ ）。

若输出的数大于 0，则下面说法正确的是(）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 127

B. 126

C. 128

D. 125

6.

A. 若输出为偶数，则输入的 d[i]中最多有两个偶数

B. 若输出为奇数，则输入的 d[i]中至少有两个奇数

C. 若输出为偶数，则输入的 d[i]中至少有两个偶数

D. 若输出为奇数，则输入的 d[i]中最多有两个奇数

100.单选题 [GESP 样题【3 级】] 一个 int 类型的值乘以 8，等价于以下哪个位运算？( )

A. 左移 3 位

B. 右移 3 位

C. 左移 8 位

D. 右移 8 位

101.单选题 [GESP202306【3 级】] 一个 int 类型的值，做以下哪个操作，一定会变回原来的值？（ ）

A. 左移 3 位，再右移 3 位。

B. 右移 3 位，再左移 3 位。

C. 按位或 7，再按位与-8。

D. 按位异或 7，再按位异或 7。

102.判断题 [GESP202309【3 级】/GESP 样题【3 级】] 如果 a 为 int 类型的变量，且表达式 ((a & 1) == 0) 的值为 true ，则说明 a 是偶数。

103.判断题 [GESP202306【3 级】] 如果 a 为 int 类型的变量，且表达式((a | 3) == 3)的值为 true，则说明 a 在从 0 到 3 之间（可能为 0、可能为 3）。

104.单选题 [GESP 样题【3 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式能正确判断“a 等于 0 且 b 等于 0”的是（）

A. ((~a) && (~b))

B. ((a & b) == 0)

C. ((a | b) == 0)

D. ((a ^ b) == 0)

105.单选题 [GESP202306【3 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式不能正确判断“ a 等于 b ”的是（ ）。

A. ((a/b)==1)

B. ((a&b)==a)

C. ((a^b)==0)

D. ((a|b)==b)

106.单选题 [GESP202309【3 级】] 如果 a 和 b 均为 int 类型的变量，下列表达式不能正确判断“ a 等于 b ”的是（ ）。

A. ((a >= b) && (a <= b))

B. ((a >> 1) == (b >> 1))

C. ((a + b) == (a + a))

D. ((a ^ b) == 0)

107.单选题 [GESP202309【5 级】] 如果 a 和 b 均为 int 类型的变量，且 b 的值不为 0 ，那么下列能正确判断“ a 是 b 的 3 倍”的表达式是（）。

A. (a >> 3 == b)

B. (a - b) % 3 == 0

C. (a / b == 3)

D. (a == 3 * b)

108.单选题 [GESP202309【6 级】] 如果 a 和 b 均为 int 类型的变量，且 b 的值不为 0 ，那么下列能正确判断“ a 是 b 的 3 倍”的表达式是（ ）。

A. (a >> 3 == b)

B. (a - b) % 3 == 0

C. (a / b == 3)

D. (a == 3 * b)

109.单选题 [GESP 样题【3 级】] 如果 a 为 int 类型的变量，下列哪个表达式可以正确求出满足“大于等于 a 且是 4 的倍数”的整数中最小的？

A. (a & (~3))

B. (a / 4 * 4)

C. ((a - 1) | 3) + 1

D. (a << 2)

110.单选题 [GESP202306【3 级】] 如果 a 为 int 类型的变量，下列哪个表达式可以正确求出满足“小于等于 a 且是 4 的倍数”的整数中最大的？（ ）

A. (a & (~3))

B. ((a << 2) >> 2)

C. (a ^ 3)

D. ((a - 1) | 3) + 1

111.单选题 [GESP202406【3 级】] 下列代码的输出结果是（ ）。

```
#include <iostream>
using namespace std;
int main() {
    int a = 12;
    int result = a >> 2;
    cout << result << endl;
    return 0;
}
```

A. 12

B. 6

C. 3

D. 1

112.判断题 [GESP202406【3 级】] 执行下面 C++代码后，输出的结果是 8。

int a = 0b1010;

int b = 01100;

int c = a & b;

cout << c <<endl;

113.单选题 [GESP202406【3 级】] 下列代码的输出结果是（ ）。

```
#include <iostream>
using namespace std;
int main() {
    int a = 5; int b = 10;
    a = a ^ b; b = a ^ b;
    a = a ^ b;
    cout << ""a = "" << a << "", b = "" << b << endl;
    return 0;
}
```

A. a = 5, b = 10

B. a = 5, b = 5

C. a = 10, b = 5

D. a = 10, b = 10

114.单选题 [GESP 样题【3 级】] 在下列代码的横线处填写（ ），可以使得输出是“17 11”。

```
#include <iostream>
using namespace std;
int main() {
    int a = 5;
    int b = 10;
    a = a ^ b;
```

```
    b = a ^ b;
    a = a ^ b;
    cout << "a = " << a << ", b = " << b << endl;
    return 0;
}
```

A. a + b

B. a - b

C. a ^ b

D. a & b

115.单选题 [GESP202306【3 级】] 在下列代码的横线处填写（ ），可以使得输出是“24 12”。

```
#include <iostream>
using namespace std;
int main(){
    int a= 12, b = 24;
    ________//在此处填入代码
    b = a ^ b;
    a = a ^ b;
    cout << a << " " << b << endl;
    return 0;
}
```

A. a = a ^ b

B. b = a ^ b

C. a = a + b

D. b = a + b

116.单选题 [GESP202309【3 级】] 在下列代码的横线处填写（ ），可以使得输出是“20 10”。

```
#include <iostream>
using namespace std;
int main() {
    int a = 10, b = 20;
    a = (a << 8) | b;
    // 在此处填入代码
```

```
    cout << a << " " << b << endl;
    return 0;
}
```

A. a = a >> 8; b = a & 0xff

B. b = a >> 8; a = a & 0xff;

C. a = b; b = a & 0xff;

D. b = a; a = b;

117.单选题 [GESP202306【3 级】] 在下列代码的横线处填写（ ），可以使得输出不是“31”。

```
#include <iostream>
using namespace std;
int main(){
    int array[5]={1,2,4,8,16};
    int res =0;
    for(int i=0;i<5; i++)
        _________//在此处填写代码
    cout<< res<< endl;
    return 0;
}
```

A. res = res + array[i]

B. res = res & array[i]

C. res = res | array[i]

D. res = res ^ array[i]

118.单选题 [NOIP 普及组 2018/NOIP 提高组 2018] 为了统计一个非负整数的二进制形式中 1 的个数，代码如下：

```
int CountBit(int x) {
    int ret = 0;
    while (x) {
        ++ret;
        (    ) ;
    }
}
```

则空格内要填入的语句是（ ）。

A. x >>= 1

B. x &= x - 1

C. x |= x >> 1

D. x <<= 1

119.填空题 [NOIP 提高组 2018] 方程 a*b = (a or b) *(a and b)，在 a,b 都取 [0, 31]中的整数时，共有_____组解。(* 表示乘法；or 表示按位或运算；and 表示按位与运算)

120.填空题 [CSP-J2021]

```
#include <iostream>
#include <string>
using namespace std;

char base[64];
char table[256];

void init() {
    for (int i = 0; i < 26; i++) base[i] = 'A' + i;
    for (int i = 0; i < 26; i++) base[26 + i] = 'a' + i;
    for (int i = 0; i < 10; i++) base[52 + i] = '0' + i;
    base[62] = '+';
    base[63] = '/';

    for (int i = 0; i < 256; i++) table[i] = 0xff;
    for (int i = 0; i < 64; i++) table[base[i]] = i;
    table['='] = 0;
}

string decode(string str) {
    string ret;
    int i;
    for (i = 0; i < str.size(); i += 4) {
        ret += table[str[i]] << 2 | table[str[i + 1]] >> 4;
        if (str[i + 2] != '=') {
            ret += (table[str[i + 1]] & 0x0f) << 4 | table[str[i + 2]] >> 2;
        }
        if (str[i + 3] != '=') {
```

```
                ret += (table[str[i + 2]] & 0x3) << 6 | table[str[i + 3]];
            }
        }
        return ret;
}

int main() {
        init();
        cout << int(table[0]) << endl;
        string str;
        cin >> str;
        cout << decode(str) << endl;
        return 0;
}
```

输出的第二行一定是由小写字母、大写字母、数字和 +、 /、= 构成的字符串。（ ）

可能存在输入不同，但输出的第二行相同的情形。（ ）

输出的第一行为 -1。（ ）

单选题

设输入字符串长度为 n，decode 函数的时间复杂度为（ ）

当输入为 Y3Nx 时，输出的第二行为（）。

当输入为 Y2NmlDlwMjE= 时，输出的第二行为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. O(√n)

B. O(n)

C. O(nlogn)

D. O(n^2)

5.

A. csp

B. csq

C. CSP

D. Csp

6.

A. ccf2021

B. ccf2022

C. ccf 2021

D. ccf 2022

121.单选题 [CSP-S2021]

```
#include <iostream>
#include <string>
using namespace std;

char base[64];
char table[256];

void init()
{
    for (int i = 0; i < 26; i++) base[i] = 'A' + i;
    for (int i = 0; i < 26; i++) base[26 + i] = 'a' + i;
    for (int i = 0; i < 10; i++) base[52 + i] = '0' + i;
    base[62] = '+', base[63] = '/';

    for (int i = 0; i < 256; i++) table[i] = 0xff;
    for (int i = 0; i < 64; i++) table[base[i]] = i;
    table['='] = 0;
}

string encode(string str)
{
    string ret;
    int i;
```

```
    for (i = 0; i + 3 <= str.size(); i += 3) {
        ret += base[str[i] >> 2];
        ret += base[((str[i] & 0x03) << 4) | (str[i + 1] >> 4)];
        ret += base[((str[i + 1] & 0x0f) << 2) | (str[i + 2] >> 6)];
        ret += base[str[i + 2] & 0x3f];
    }
    if (i < str.size()) {
        ret += base[(str[i] & 0xff) >> 2];
        if ((i + 1) < str.size()) {
            ret += base[((str[i] & 0x03) << 4) | (str[i + 1] >> 4)];
            ret += base[(str[i + 1] & 0x0f) << 2];
            ret += ""="";
        } else {
            ret += base[(str[i] & 0x03) << 4];
            ret += ""=="";
        }
    }
    return ret;
}

string decode(string str)
{
    string ret;
    int i;
    for (i = 0; i + 4 < str.size(); i += 4) {
        ret += (table[str[i]] << 2) | (table[str[i + 1]] >> 4);
        ret += ((table[str[i + 1]] & 0x0f) << 4) | (table[str[i + 2]] >> 2);
        ret += ((table[str[i + 2]] & 0x03) << 6) | table[str[i + 3]];
    }
    if (str[i + 2] == '=') {
        ret += (table[str[i]] << 2) | (table[str[i + 1]] >> 4);
    } else if (str[i + 3] == '=') {
        ret += (table[str[i]] << 2) | (table[str[i + 1]] >> 4);
        ret += ((table[str[i + 1]] & 0x0f) << 4) | (table[str[i + 2]] >> 2);
    }
    return ret;
}

int main()
{
    init();
    cout << (int(table[0]) & endl;
    int opt;
    string str;
    cin >> opt >> str;
```

```
69      cout << (opt ? decode(str) : encode(str)) << endl;
70      return 0;
71  }
```

假设输入总是合法的（一个整数和一个不含空白字符的字符串，用空格隔开），完成下面的判断题和单选题：

判断题

程序总是先输出一行一个整数，再输出一行一个字符串。（ ）

对于任意不含空白字符的字符串 str1，先执行程序输入 0 str1，得到输出的第二行记为 str2 再执行程序输入 1 str2，输出的第二行必为 str1。（ ）

当输入为 1 SGVsbG93b3JsZA==时，输出的第二行为 HelloWorld。( )

单选题

设输入字符串长度为 n，encode 函数的时间复杂度为（ ）。

输出的第一行为（ ）。

当输入为 0 CSP2021csp 时，输出的第二行为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. $O(\sqrt{n})$

B. $O(n)$

C. $O(nlogn)$

D. $O(n^2)$

5.

A. 0xff

B. 255

C. 0xFF

D. -1

6.

A. Q1NQMjAyMWNzcAv=

B. Q1NQMjAyMGNzcA==

C. Q1NQMjAyMGNzcAv=

D. Q1NQMjAyMWNzcA==

答案

1.B

2.A

3.C

4.B

5.B

6.D

7.C

8.B

9.A

10.C

11.D

12.D

13.B

14.D

15.A

16.B

17.D

18.D

19.B

20.B

21.D

22.BCE

23.AB

24.ABD

25.D

26.BCD

27.ABC

28.D

29.F

30.D

31.D

32.C

33.B

34.C

35.C

36.A

37.B

38.A

39.B

40.A

41.B

42.C

43.D

44.C

45.C

46.AB

47.B

48.B

49.B

50.D

51.BD

52.F

53.F

54.T

55.AC

56.A

57.T

58.T

59.B

60.B

61.D

62.C

63.D

64.F

65.T

66.T

67.F

68.D

69.D

70.B

71.C

72.C

73.D

74.B

75.F

76.F

77.C

78.A

79.D

80.C

81.C

82.T

83.F

84.C

85.F

86.B

87.B

88.A

89.F

90.F

91.T

92.F

93.F

94.A

95.B

96.ABABBD

97.ABBBBB

98.BBBABB

99.BBAAC

100.A

101.C

102.T

103.T

104.C

105.C

106.B

107.D

108.D

109.C

110.A

111.C

112.F

113.C

114.C

115.B

116.B

117.B

118.B

119.454

120.BAABBC

121.BABBDD

GW 信奥 CSP 初赛习题集 5-4

计算机数学——组合数学

1. 排列组合计数

1. 基本计数原理

1.单选题 [GESP 样题【8 级】] 从 A 城到 C 城需要经过 B 城，其中 A 到 B 可选高铁和飞机，B 到 C 可以自驾或打的，请问 A 到 C 有几种交通选择（ ）。

A. 2

B. 4

C. 8

D. 不知道

2.单选题 [GESP202312【8 级】] 小杨要从 A 城到 B 城，又想顺路游览一番。他有两个选项：1、坐高铁路到 C 城游览，再坐高铁或飞机到 B 城；2、坐船到 D 城游览，再坐船、高铁或飞机到 B 城。请问小杨从 A 城到 B 城共有几种交通方案可以选择？（ ）。

A. 2

B. 3

C. 5

D. 6

3.单选题 [GESP202403【8 级】] 为丰富食堂菜谱，炒菜部进行头脑风暴。肉类有鸡肉、牛肉、羊肉、猪肉 4 种，切法有肉排、肉块、肉末 3 种，配菜有圆白菜、油菜、豆腐 3 种，辣度有麻辣、微辣、不辣 3 种。不考虑口感的情况下，选 1 种肉、1 种切法、1 种配菜、1 种辣度产生一道菜（例如：麻辣牛肉片炒豆腐），这样能产生多少道菜？（ ）。

A. 13

B. 42

C. 63

D. 108

4.单选题 [NOIP 普及组 2017] 甲、乙、丙三位同学选修课程，从 4 门课程中，甲选修 2 门，乙、丙各选修 3 门，则不同的选修方案共有( )种。

A. 36

B. 48

C. 96

D. 192

5.单选题 [CSP-S2022] 共有 8 人选修了程序设计课程，期末大作业要求由 2 人组成的团队完成。假设不区分每个团队内 2 人的角色和作用，请问共有多少种可能的组队方案。（ ）。

A. 28

B. 32

C. 56

D. 64

6.单选题 [CSP-J2020] 有五副不同颜色的手套（共 10 只手套，每副手套左右手各 1 只），一次性从中取 6 只手套，请问恰好能配成两副手套的不同取法有（ ）种。

A. 120

B. 180

C. 150

D. 30

7.判断题 [GESP 样题【8 级】] 学校组织班际排球赛，每个班级可以派男女各一个参赛队伍，每队 5 人。班级 A 的每位同学都可以报名，那可以用加法原理来计算 A 班有多少支候选队伍。

8.判断题 [GESP202406【8 级】] 一个袋子中有 3 个完全相同的红色小球、2 个完全相同的蓝色小球。每次从中取出 1 个，再放回袋子，这样进行 3 次后，可能的颜色顺序有 8 种。

2. 组合问题

9.判断题 [GESP 样题【8 级】] 从 15 本不同的书中选 3 本，总共有 455 种方法。

10.填空题 [NOIP 提高组 2008] 书架上有 21 本书，编号从 1 到 21，从其中选 4 本，其中每两本的编号都不相邻的选法一共有_____种。

11.单选题 [CSP-J2023] 一个班级有 10 个男生和 12 个女生。如果要选出一个 3 人的小组，并且小组中必须至少包含 1 个女生，那么有多少种可能的组合?()

A.1420

B.1770

C.1540

D.2200

12.填空题 [NOIP 普及组 2016] 从一个 4×4 的棋盘（不可旋转）中选取不在同一行也不在同一列上的两个方格，共有（ ）种方法。

13.填空题 [NOIP 提高组 2016] 一个 1×8 的方格图形（不可旋转）用黑、白两种颜色填涂每个方格。如果每个方格只能填涂一种颜色，且不允许两个黑格相邻，共有_____种填涂方案。

14.单选题 [CSP-S2020] 从一个 4×4 的棋盘中选取不在同一行也不在同一列上的两个方格，共有（ ）种方法。

A. 60

B. 72

C. 86

D. 64

15.单选题 [CSP-S2021] 有 8 个苹果从左到右排成一排，你要从中挑选至少一个苹果，并且不能同时挑选相邻的两个苹果，一共有（ ）种方案。

A. 36

B. 48

C. 54

D. 64

16.单选题 [CSP-J2023] 小明在某一天中依次有七个空闲时间段，他想要选出至少一个空闲时间段来练习唱歌，但他希望任意两个练习的时间段之间都有至少两个空闲的时间段让他休息。则小明一共有（）种选择时间段的方案。

A. 31

B. 18

C. 21

D. 33

17.单选题 [CSP-S2021] 设一个三位数 n=abc,a、b、c 均为 1~9 之间的整数，若以 a、b、c 作为三角形的三条边可以构成等腰三角形（包括等边），则这样的 n 有（ ）个。

A.81

B.120

C.165

D.216

18.单选题 [NOIP 普及组 2018] 设含有 10 个元素的集合的全部子集数为 S，其中由 7 个元素组成的子集数为 T，则 T/S 的值为（ ）。

A. 5 / 32

B. 15 / 128

C. 1 / 8

D. 21 / 128

3. 排列问题

19.填空题 [NOIP 普及组 2015] 重新排列 1234 使得每一个数字都不在原来的位置上，一共有（ ）种排法。

20.单选题 [GESP202312【8 级】] 5 位同学排队，其中一位同学不能排在第一，则共有多少种可能的排队方式？（ ）。

A. 5

B. 24

C. 96

D. 120

21.单选题 [CSP-J2020] 5 个小朋友并排站成一列，其中有两个小朋友是双胞胎，如果要求这两个双胞胎必须相邻，则有（ ）种不同排列方法?

A. 48

B. 36

C. 24

D. 72

22.填空题 [NOIP 普及组 2008] 书架上有 4 本不同的书 A、B、C、D。其中 A 和 B 是红皮的，C 和 D 是黑皮的。把这 4 本书摆在书架上，满足所有黑皮的书都排在一起的摆法有_____种。满足 A 必须比 C 靠左，所有红皮的书要摆放在一起，所有黑皮的书要摆放在一起，共有_____种摆法。

23.判断题 [GESP202406【8 级】] ABCDE 五个小朋友，排成一队跑步，其中 AB 两人必须排在一起，一共有 48 种排法。

4. 先选再排问题

24.填空题 [NOIP 提高组 2014] 由数字 1,1,2,4,8,8 所组成的不同的四位数的个数是_____.

25.单选题 [CSP-J2021] 由 1,1,2,2,3 这五个数字组成不同的三位数有（ ）种。

A. 18

B. 15

C. 12

D. 24

26.单选题 [CSP-S2019] 由数字 1, 1, 2, 4, 8, 8 组成的不同的 4 位数的个数是（）。

A. 104

B. 102

C. 98

D. 100

27.单选题 [CSP-S2023] 0,1,2,3,4 中选取 4 个数字，能组成（ ）个不同四位数（注：最小的四位数是 1000 最大的四位数是 9999）。

A.96

B.18

C.120

D.84

28.单选题 [CSP-S2022] 小明希望选到形如“省 A·LLDDD”的车牌号。车牌号在“·”之前的内容固定的 5 位号码中，前 2 位必须是大写英文字母，后 3 位必须是阿拉伯数字（L 代表 A 至 Z，D 表示 0 至 9，两个 L 和三个 D 之间可能相同也可能不同）。请问总共有多少个可供选择的车牌号。（ ）

A. 20280

B. 52000

C. 676000

D. 1757600

29.单选题 [CSP-J2019] 一些数字可以颠倒过来看，例如 0,1,8 颠倒过来还是本身，6 颠倒过来是 9，9 颠倒过来看还是 6，其他数字颠倒过来都不构成数字。类似的，一些多位数也可以颠倒过来看，比如 106 颠倒过来是 901。假设某个城市的车牌只由 5 位数字组成，每一位都可以取 0 到 9。请问这个城市最多有多少个车牌倒过来恰好还是原来的车牌？（）

A. 60

B. 125

C. 75

D. 100

30.单选题 [CSP-S2019] 一些数字可以颠倒过来看，例如 0、1、8 颠倒过来还是本身，6 颠倒过来是 9，9 颠倒过来看还是 6，其他数字颠倒过来都不构成数字。类似的，一些多位数也可以颠倒过来看，比如 106 颠倒过来是 901。假设某个城市的车牌只由 5 位数字组成，每一位都可以取 0 到 9。请问这个城市最多有多少个车牌倒过来恰好还是原来的车牌，并且车牌上的 5 位数能被 3 整除（）。

A. 40

B. 25

C. 30

D. 20

31.单选题 [GESP202406【8 级】] GESP 活动期间，主办方从获胜者 ABCDE 五个人中选出三个人排成一队升国旗，其中 A 不能排在队首，请问有多少种排法？

A. 24

B. 48

C. 32

D. 12

32.单选题 [GESP202406【8 级】] 0,1,2,3,4,5 这些数字组成一个三位数，请问没有重复数字的情况下，有多少种组法（ ）。

A. 180

B. 120

C. 80

D. 100

5. 小球小盒模型

33.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 有 7 个一模一样的苹果，放到 3 个一样的盘子中，一共有 ( )种放法。

A. 7

B. 8

C. 21

D. 37

34.单选题 [CSP-J2021] 6 个人，两个人组一队，总共组成三队，不区分队伍的编号。不同的组队情况有（ ）种。

A. 10

B. 15

C. 30

D. 20

35.单选题 [CSP-J2020] 10 个三好学生名额分配到 7 个班级，每个班级至少有一个名额，一共有（ ）种不同的分配方案。

A. 84

B. 72

C. 56

D. 504

36.单选题 [NOIP 提高组 2017] 将 7 个名额分给 4 个不同的班级，允许有的班级没有名额，有( )种不同的分配方案。

A. 60

B. 84

C. 96

D. 120

37.单选题 [CSP-J2019] 把 8 个同样的球放在 5 个同样的袋子里，允许有的袋子空着不放，问共有多少种不同的分法（ ）

提示：如果 8 个球都放在一个袋子里，无论是哪个袋子，都只算同一种分法。

A. 22

B. 24

C. 18

D. 20

38.填空题 [NOIP 普及组 2014] 把 M 个同样的球放到 N 个同样的袋子里，允许有的袋子空着不放，问共有多少种不同 的放置方法？ (用 K 表示)。 例如， M＝7 ，N＝3 时， K＝8 ；在这里认为和是同一种放置方法。 问： M＝8 ，N＝5 时， K＝_____ 。

39.填空题 [NOIP 普及组 2007] （子集划分）将 n 个数{1，2，…，n}划分成 r 个子集。每个数都恰好属于一个子集，任何两个不同的子集没有共同的数，也没有空集。将不同划分方法的总数记为 S(n,r)。例如，S(4,2)=7，这 7 种不同的划分方法依次为{(1),(234)}, {(2),(134)}, {(3),(124)}, {(4),(123)}, {(12),(34)}, {(13),(24)}, {(14),(23)}。当 n=6,r=3 时，S(6,3)=？（提示：先固定一个数，对于其余的 5 个数考虑 S(5,3)与 S(5,2)，再分这两种情况对原固定的数进行分析）。

40.填空题 [NOIP 提高组 2007] 给定 n 个有标号的球，标号依次为 1，2，…，n。将这 n 个球放入 r 个相同的盒子里，不允许有空盒，其不同放置方法的总数记为 S(n,r)。例如，S(4,2)=7，这 7 种不同的放置方法依次为{(1),(234)}, {(2),(134)}, {(3),(124)}, {(4),(123)}, {(12),(34)}, {(13),(24)}, {(14),(23)}。当 n=7,r=4 时，S(7,4)=？

6. 子串模型

41.单选题 [NOIP 普及组 2017] 若串 S=""copyright""，其子串的个数是（ ）。

A. 72

B. 45

C. 46

D. 36

42.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 设字符串 S=”Olympic”，S 的非空子串的数目是（ ）。

A. 28

B. 29

C. 16

D. 17

43.单选题 [NOIP 普及组 2012] 原字符串中任意一段连续的字符组成的新字符串称为子串。则字符串“AAABBBCCC”共有（ ）个不同的非空子串。

A. 3

B. 12

C. 36

D. 45

44.单选题 [CSP-J2022] 一个字符串中任意个连续的字符组成的子序列称为该字符串的子串，则字符串 abcab 有（ ）个内容互不相同的子串。

A.12

B.13

C.14

D.15

45.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 由 3 个 a，1 个 b 和 2 个 c 构成的所有字符串中，包含子串“abc”的共有（ ）个。

A. 20

B. 8

C. 16

D. 12

E. 24

7. 分类讨论思想应用

46.判断题 [GESP202403【8 级】] 一个袋子中有 3 个完全相同的红色小球、2 个完全相同的蓝色小球。每次从中取出 1 个，再放回袋子，这样进行 3 次后，可能的颜色顺序有 7 种。

47.判断题 [GESP202312【8 级】] 一个袋子中有 3 个完全相同的红色小球、2 个完全相同的蓝色小球。每次从中取出 1 个，且不放回袋子，这样进行 3 次后，将取出的小球依次排列，则可能的颜色顺序有 7 种。

48.单选题 [GESP202403【8 级】] 已知袋中有 2 个相同的红球、3 个相同的绿球、5 个相同的黄球。每次取出一个不放回，全部取出。可能产生多少种序列？（ ）。

A. 6

B. 1440

C. 2520

D. 3628800

49.填空题 [NOIP 普及组 2009] 小陈现有 2 个任务 A，B 要完成，每个任务分别有若干步骤如下：A=a1->a2->a3，B=b1->b2->b3->b4->b5。在任何时候，小陈只能专心做某个任务的一个步骤。但是如果愿意，他可以在做完手中任务的当前步骤后，切换至另一个任务，从上次此任务第一个未做的步骤继续。每个任务的步骤顺序不能打乱，例如……a2->b2->a3->b3……是合法的，而……a2->b3->a3->b2……是不合法的。小陈从 B 任务的 b1 步骤开始做，当恰做完某个任务的某个步骤后，就停工回家吃饭了。当他回来时，只记得自己已经完成了整个任务 A，其他的都忘了。试计算小陈饭前已做的可能的任务步骤序列共有 种。

50.填空题 [NOIP 普及组 2011] 每份考卷都有一个 8 位二进制序列号。当且仅当一个序列号含有偶数个 1 时，它才是有效的。例如，00000000、01010011 都是有效的序列号，而 11111110 不是。那么，有效的序列号共有____个。

51.填空题 [NOIP 普及组 2018] 从 1 到 2018 这 2018 个数中，共有___个包含数字 8 的数。包含数字 8 的数是指有某一位是“8”的数，例如“2018”与“188”。

52.填空题 [NOIP 普及组 2010] 队列快照是指在某一时刻队列中的元素组成的有序序列。例如，当元素 1 、2 、3 入队， 元素 1 出队后， 此刻的队列快照是 “2 3” 。

当元素 2、3 也出队后，队列快照是 “” ，即为空。 现有 3 个正整数元素依次入队、出队。已知它们的和为 8，则共有 _____ 种可能的不 同的队列快照（不同队列的相同快照只计一次）。例如，“5 1″ 、”4 2 2″ 、”” 都是可能 的队列快照；而 “7” 不是可能的队列快照，因为剩下的 2 个正整数的和不可能是 1。

53.填空题 [NOIP 提高组 2010] 记 T 为一队列初始为空现有 n 个总和不超过 32 的正整数依次入队如果无论这些数具 体为何值都能找到一种出队的方式使得存在某个时刻队列 T 中的数之和恰好为 9 那么 n 的最小值是 _____。

54.判断题 [GESP202312【8 级】] 杨辉三角，是二项式系数的一种三角形排列，在中国南宋数学家杨辉 1261 年所著的《详解九章算法》一书中出现，是中国数学史上的一项伟大成就。

55.单选题 [GESP 样题【8 级】] 约定杨辉三角形第 0 行只有 1 个元素是 1，第 1 行有 2 个元素都是 1，第四行的所有元素之和是（ ）？

A. 8

B. 16

C. 24

D. 32

2. 杨辉三角

56.单选题 [GESP202403【8 级】/GESP202312【8 级】] 下面程序的时间复杂度为（ ）。

```
int choose(int n,int m){
    if(m==0||m==n)
        return 1;
    return choose(n-1,m-1)+choose(n-1,m);
}
```

A.$O(2^n)$

B.O(2^m*(n-m))

C.O(C(n,m))

D.O(m*(n-m))

57.填空题 [NOIP 提高组 2009]

```
#include <iostream>
using namespace std;
const int maxn = 50;
const int y = 2009;
int main() {
    int n, c[maxn][maxn], i, j, s = 0;
    cin >> n;
    c[0][0] = 1;

    for (i = 1; i <= n; i++) {
        c[i][0] = 1;
        for (j = 1; j < i; j++) {
            c[i][j] = c[i-1][j-1] + c[i-1][j];
        }
        c[i][i] = 1;
    }

    for (i = 0; i <= n; i++) {
        s = (s + c[n][i]) % y;
    }
    cout << s << endl;
    return 0;
}
```

输入：17

输出：

58.单选题 [GESP 样题【7 级】] 下面函数尝试使用动态规划方法求出如下递推公式的函数，则横线处填写下列哪段代码可以完成预期功能？

```
c(n,m)=1(n==0 或 m==0）,c(n-1,m-1)+c(n-1,m)（n>0 且 m>0）
int rec_C[MAXN][MAXM];
int C(int n, int m) {
    for (int i = 0; i <= n; i++) {
        rec_C[i][0] = 1;
```

```
    }
    for (int j = 0; j <= m; j++) {
        rec_C[0][j] = 1;
    }
    _____// 递推计算代码
    rec_C[i][j] = rec_C[i - 1][j - 1] + rec_C[i - 1][j];
    return rec_C[n][m];
}
```

A. for (int i = n; i >= 1; i--) for (int j = 1; j <= m; j++)

B. for (int i = 1; i <= n; i++) for (int j = m; j >= 1; j--)

C. for (int j = 1; j <= m; j++) for (int i = 1; i <= n; i++)

D. for (int j = 1; j <= m; j++) for (int i = n; i >= 1; i--)

答案

1.B

2.C

3.D

4.C

5.A

6.A

7.T

8.T

9.T

10.3060

11.A

12.72

13.55

14.B

15.C

16.B

17.C

18.B

19.9

20.C

21.A

22.12 4

23.T

24.102

25.A

26.C

27.A

28.C

29.C

30.B

31.B

32.D

33.B

34.B

35.A

36.D

37.C

38.18

39.90

40.350

41.C

42.A

43.C

44.B

45.D

46.F

47.T

48.C

49.70

50.128

51.554

52.49

53.18

54.T

55.B

56.D

57.487

58.C

GW 信奥 CSP 初赛习题集 5-5

计算机数学——其他

1. 统筹规划问题

1.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] 周末小明和爸爸妈妈三个人一起想动手做三道菜。小明负责洗菜、爸爸负责切菜、妈妈负责炒菜。假 设做每道菜的顺序都是：先洗菜 10 分钟，然后切菜 10 分钟，最后炒菜 10 分钟。那么做一道菜需要 30 分 钟。注意：两道不同的菜的相同步骤不可以同时进行。例如第一道菜和第二道的菜不能同时洗，也不能同 时切。那么做完三道菜的最短时间需要 ( )分钟。

A. 90

B. 60

C. 50

D. 40

2.填空题 [NOIP 普及组 2009] 有如下的一段程序：

1.a=1;

2.b=a;

3.d=-a;

4.e=a+d;

5.C=2*d;

6.f=b+e-d;

7.g=a*f+C;

现在要把这段程序分配到若干台(数量充足.用电缆连接的 PC 上做并行执行。每台 PC 执行其中的某几个语句，并可随时通过电缆与其他 PC 通讯，交换一些中间结果。假设每台 PC 每单位时间可以执行一个语句，且通讯花费的时间不计。则这段程序最快可以在_____单位时间内执行完毕。注意：任意中间结果只有在某台 PC 上已经得到，

才可以被其他 PC 引用。例如若语句 4 和 6 被分别分配到两台 PC 上执行，则因为语句 6 需要引用语句 4 的计算结果，语句 6 必须在语句 4 之后执行。

3.填空题 [NOIP 提高组 2016] 某中学在安排期末考试时发现，有 7 个学生要参加 7 门课程的考试，下表列出了哪些学生参加哪些考试（用√表示要参加相应的考试）。最少要安排_____个不同的考试时间段才能避免冲突？

考试	学生 1	学生 2	学生 3	学生 4	学生 5	学生 6	学生 7
通用技术	√				√		√
物理	√	√					√
化学		√		√			
生物	√				√	√	
历史			√	√		√	
地理		√	√				√
政治			√			√	

2. 几何问题

4.判断题 [GESP202403【7 级】] 祖冲之是南北朝时期杰出的数学家、天文学家，其主要贡献在数学、天文历法和机械制造三方面。他首次将“圆周率”精算到小数第七位，即在 3.1415926 和 3.1415927 之间。

5.判断题 [GESP 样题【8 级】] 如果一个四边形的对角线互相平分，并且两条对角线的长度都为 8，那么这个四边形的面积一定是 32。

6.不定项选择题 [NOIP 提高组 2010] 一个平面的法线是指与该平面垂直的直线。过点（ 1,1,1 ）、（ 0,3,0 ）、（ 2,0,0 ）的平面 的法线是（ ）

A. 过点（ 1,1,1 ）、（ 2,3,3 ）的直线

B. 过点（ 1,1,1 ）、（ 3,2,1 ）的直线

C. 过点（ 0,3,0 ）、（ -3,1,1 ）的直线

D. 过点（ 2,0,0 ）、（ 5,2,1 ）的直线

7.单选题 [GESP 样题【8 级】] 平面内，通过一点可以作( )条平行于给定直线的直线？

A. 0

B. 1

C. 2

D. 无限多

8.单选题 [GESP202406【4 级】] 10 条直线，最多可以把平面分为多少个区域（ ）。

A. 55

B. 56

C. 54

D. 58

9.填空题 [NOIP 普及组 2012] 如果平面上任取 n 个整点(横纵坐标都是整数)，其中一定存在两个点，它们连线的中点也是整点，那么 n 至少是____。

3. 代数问题

10.单选题 [CSP-S2019] 有一个等比数列，共有奇数项，其中第一项和最后一项分别是 2 和 118098，中间一项是 486，请问以下那个数是可能的公比 ( )。

A. 5

B. 3

C. 4

D. 2

11.单选题 [GESP 样题【8 级】] 对数列 3、4、7、12、19、28、39 求和，除简单累加外，还可以用下面（ ）来直接计算。

A. 等差数列求和

B. 等比数列求和

C. 斐波拉契数列

D. 其他某种有规律序列

12.单选题 [CSP-S2020] 小明想通过走楼梯来锻炼身体，假设从第 1 层走到第 2 层消耗 10 卡热量，接着从第 2 层走到第 3 层消耗 20 卡热量，再从第 3 层走到第 4 层消耗 30 卡热量，依此类推，从第 k 层走到第 k+1 层消耗 10k 卡热量（k>1）。如果小明想从 1 层开始，通过连续向上爬楼梯消耗 1000 卡热量，至少要爬到第几层楼？（ ）

A. 14

B. 16

C. 15

D. 13

13.填空题 [NOIP 普及组 2004] 一个家具公司生产桌子和椅子。现在有 113 个单位的木材。每张桌子要使用 20 个单位的木材，售价是 30 元；每张椅子要使用 16 个单位的木材，售价是 20 元。使用已有的木材生产桌椅（不一定要把木材用光），最多可以卖?元钱。

14.单选题 [NOIP 提高组 2017] 若 f0=0,f1=1,fn+1=(fn+fn-1)/2，则随着 i 的增大，fi 将接近于

A.1/2

B.2/3

C.(√5-1)/2

D.1

15.单选题 [GESP202406【8 级】] 从 1 到 2024 这 2024 个数中，共有（ ）个包含数字 6 的数。

A. 544

B. 546

C. 564

D. 602

答案

1.C

2.5

3.3

4.T

5.F

6.D

7.B

8.B

9.5

10.D

11.D

12.C

13.160

14.B

15.A

GW 信奥 CSP 初赛习题集 6-1

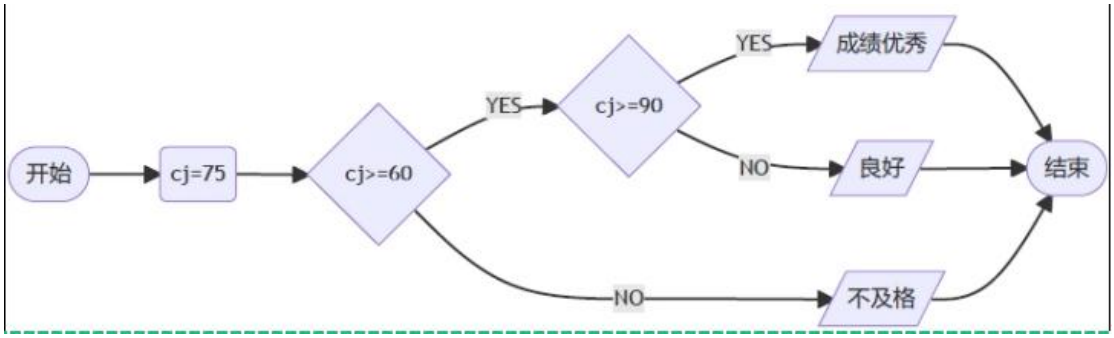
算法——算法概述

1. 算法的描述

1.判断题 [GESP202306【3 级】] 一个算法可以用不同的形式来描述，但要求描述比较规范，因此不能用自然语言描述。

2.判断题 [GESP202309【5 级】] 在特殊情况下流程图中可以出现三角框和圆形框。

3.单选题 [GESP202403【2 级】] 下列流程图的输出结果是？( )



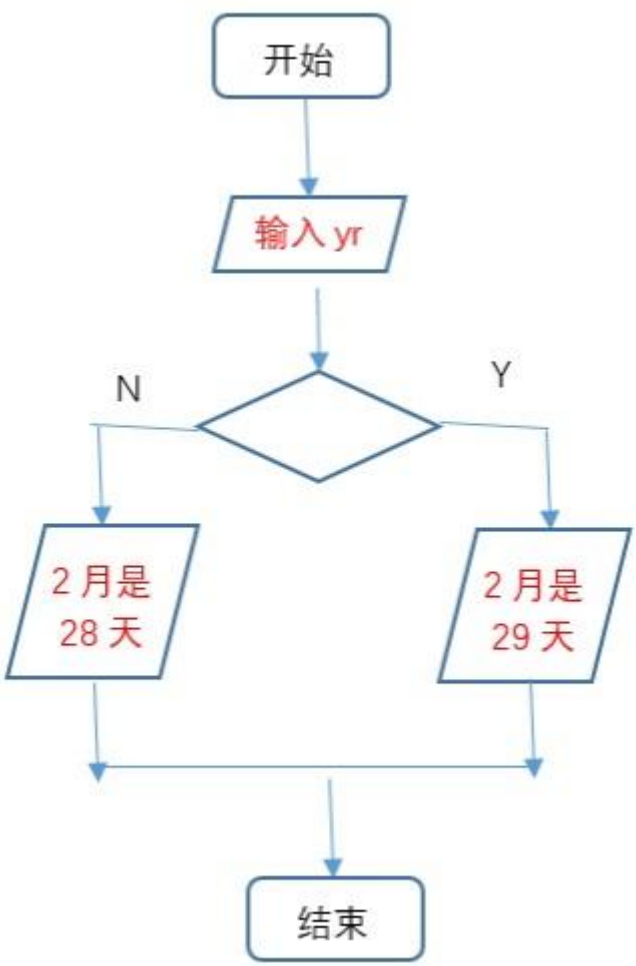
A. 优秀

B. 良好

C. 不及格

D. 没有输出

4.单选题 [GESP202406【2 级】/GESP202406【3 级】] 下面流程图在 yr 输入2024 时，可以判定 yr 代表闰年，并输出 2 月是 29 天 ，则图中菱形框中应该填入（ ）。



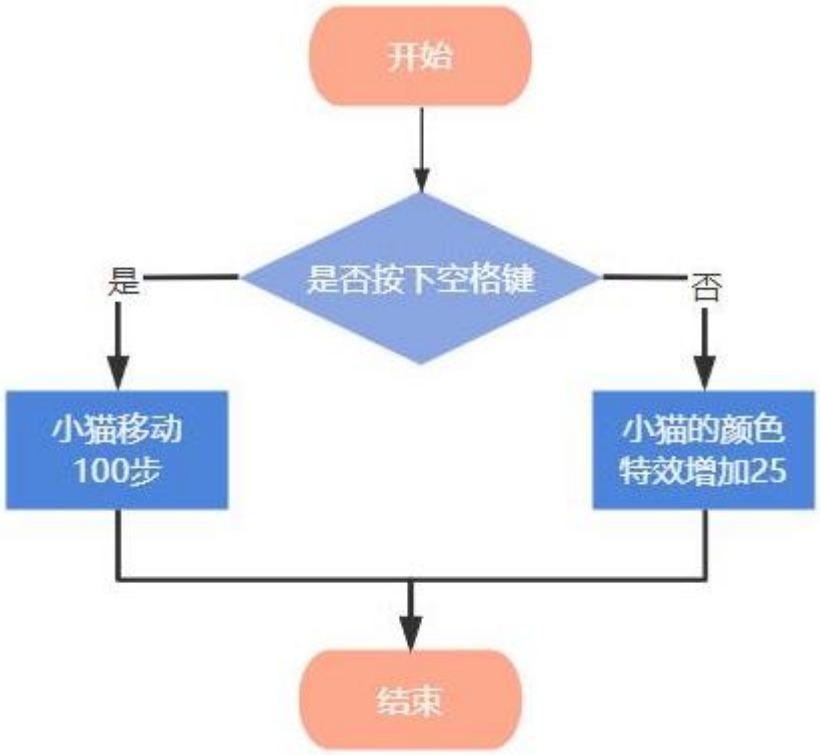
A. (yr%400==0) || (yr%4==0)

B. (yr%400==0) || (yr%4==0 && yr%100!=0)

C. (yr%400==0) && (yr%4==0)

D. (yr%400==0) && (yr%4==0 && yr%100!=0)

5.单选题 [GESP202303【2 级】] 下列流程图，属于计算机的哪种程序结构？（ ）。



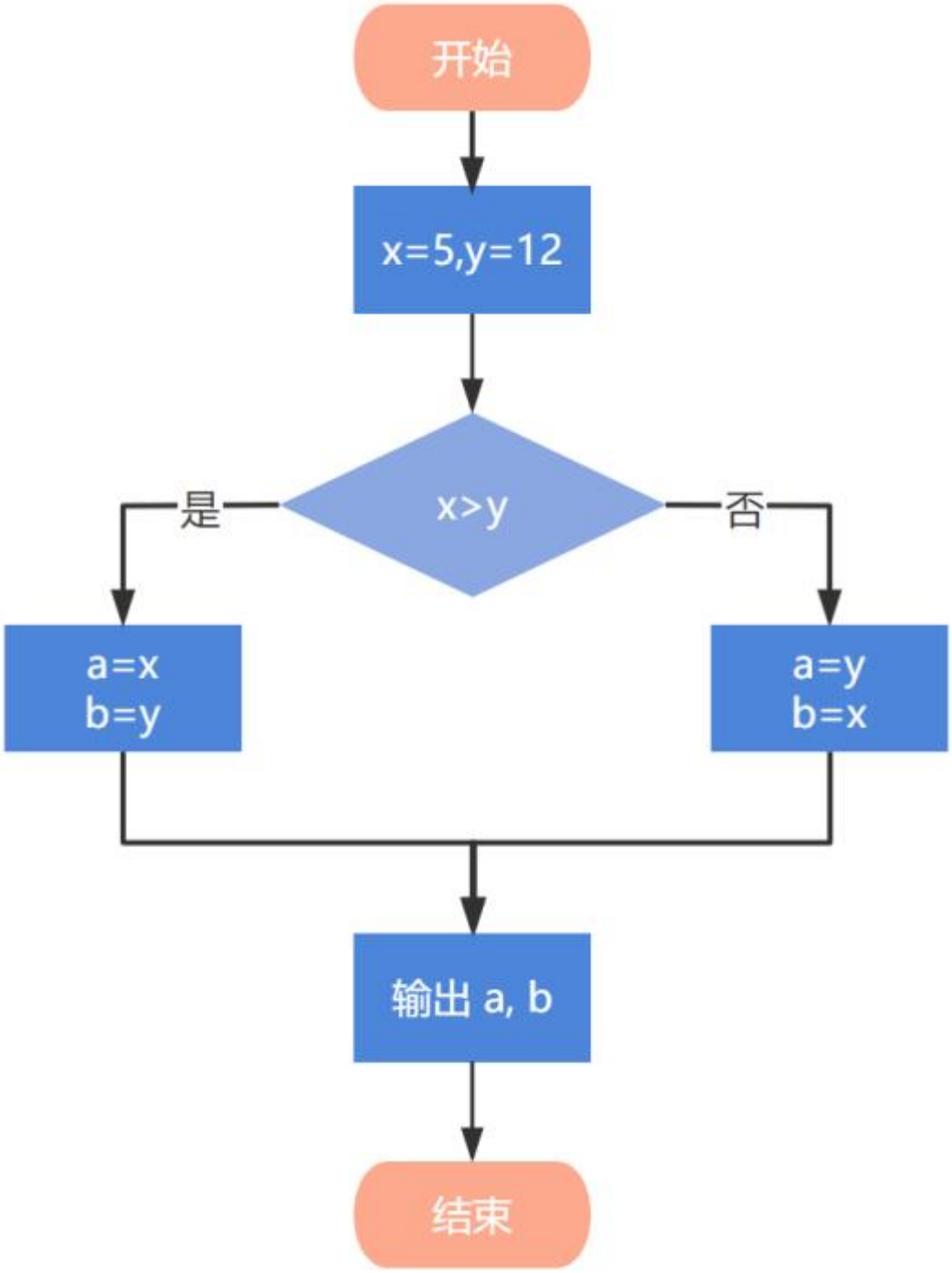
A． 顺序结构

B． 循环结构

C． 分支结构

D． 数据结构

6.单选题 [GESP202309【2 级】] 下列流程图的输出结果是（ ）？



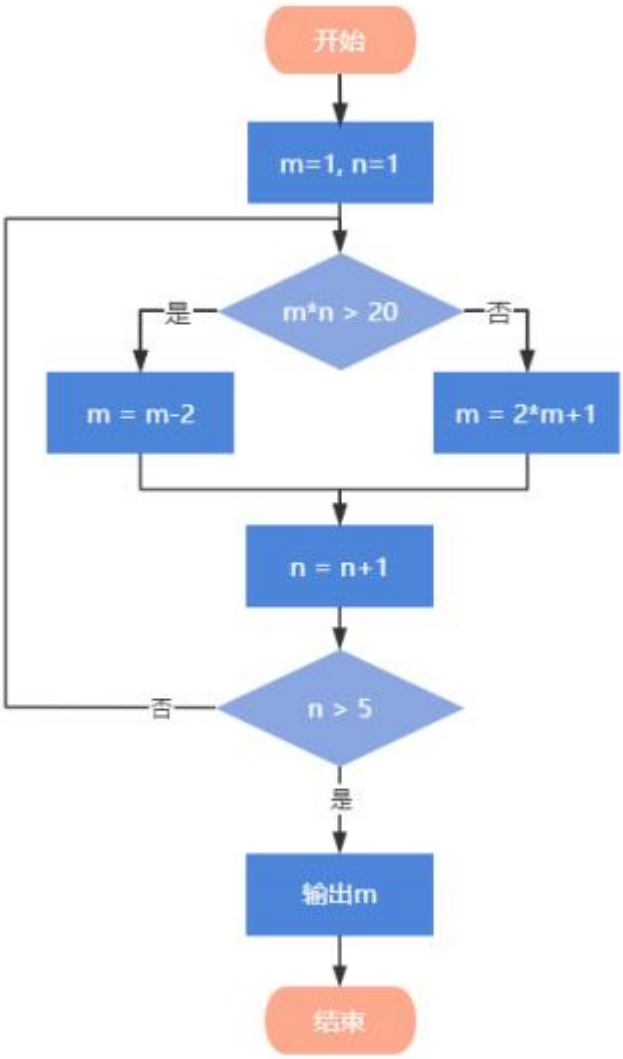
A. 5 12

B. 12 5

C. 5 5

D. 12 12

7.单选题 [GESP202309【4 级】] 下列流程图的输出结果是？( )



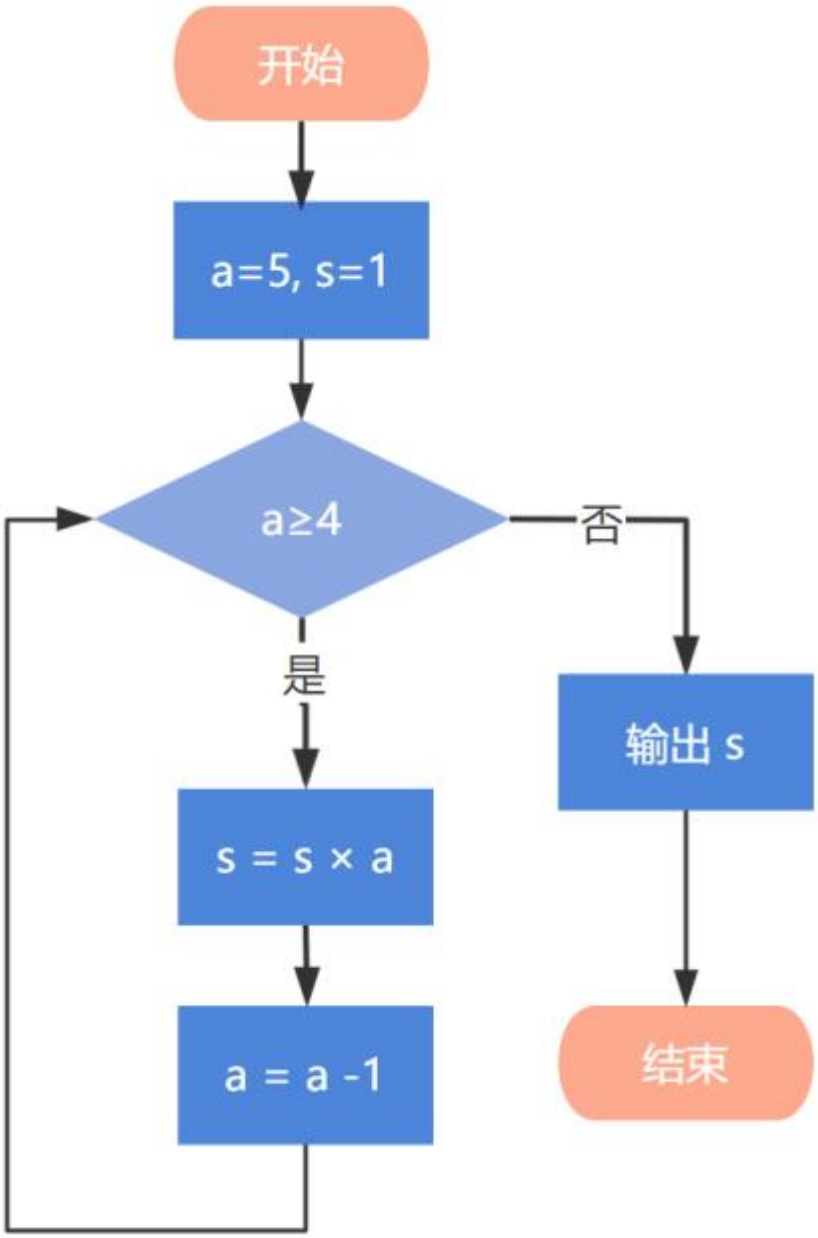
A. 9

B. 7

C. 5

D. 11

8.单选题 [GESP202309【3 级】] 下列流程图的输出结果是？（ ）



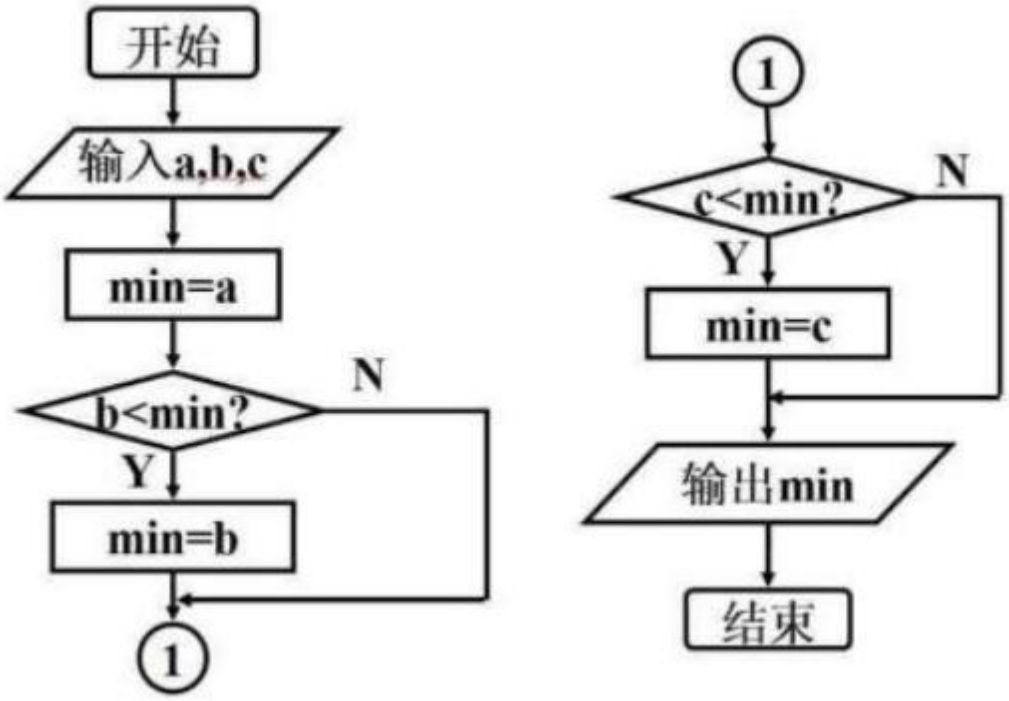
A. 60

B. 20

C. 5

D. 1

9.单选题 [GESP 样题【3 级】] 下面流程图，输入 1 2 3，会输出( )



A. 无输出

B. 1

C. 2

D. 3

10.单选题 [GESP 样题【3 级】] 通常用下列哪种方式来描述算法？

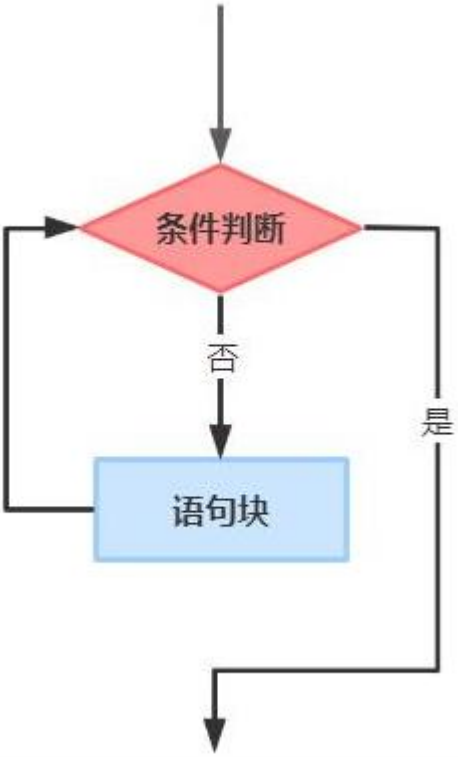
A. 汇编语言

B. 伪代码

C. SQL

D. CSS

11.单选题 [GESP202306【2 级】] 能够实现下面流程图功能的伪代码是（ ）。



A. if 条件判断 then 语句块

B. if 条件判断 then 什么也不做 else 语句块

C. while 条件判断 do 语句块

D. while not 条件判断 do 语句块

12.单选题 [CSP-J2020] 设 A 是 n 个实数的数组，考虑下面的递归算法：

```
XYZ(A[1..n])
if n=1 then return A[1]
else temp←XYZ(A[1..n-1])
if temp < A[n] then
    return temp
else return A[n]
```

请问算法 XYZ 的输出是什么？（）。

A. A 数组的平均

B. A 数组的最小值

C. A 数组的中值

D. A 数组的最大值

13.单选题 [CSP-J2020] 输入:数组 L，n≧k。输出:按非递减顺序排序的 L

```
算法 Bubblesort:
FLAG ← n //标记被交换的最后元素位置
while FLAG >1 do
    k← FLAG -1
    FLAG ← 1
    for j=1 to k do
        if L(j)>L(j+1)then do
            L(j)↔L(j+1)
            FLAG ← j
```

对 n 个数用以上冒泡排序算法进行排序，最少需要比较多少次?（ ）。

A.n^2

B.n-2

C.n-1

D.n

14.单选题 [NOIP普及组2016/NOIP提高组2016] 给定含有 n 个不同的数的数组 L=。如果 L 中存在 xi(1<i<n) 使="""" 得="""" x1<x2<...<xi-1xi+1>...>xn，，则称 L 是单峰的，并称 xi 是 L 的“峰顶”。现在已知 L 是单峰的，请把 a-c 三行代码补全到算法中使得算法正确找到 L 的峰顶。

```
a.Search(k+1, n)
b.Search(1, k-1)
c.return L[k]
Search(1, n)
1) k → [n/2]
2) if L[k] > L[k-1] and L[k] > L[k+1]
3) then __________
4) else if L[k] > L[k-1] and L[k] < L[k+1]
5) then __________
6) else __________
```

正确的填空顺序是（ ）。

A. c,a,b

B. c,b,a

C. a,b,c

D. b,a,c

15.单选题 [NOIP 提高组 2017] 在 n(n≥3) 枚硬币中有一枚质量不合格的硬币（质量过轻或质量过重），如果只有一架天平可以用来称重且称重的硬币数没有限制，下面是找出这枚不合格的硬币的算法。请把 a-c 三行代码补全到算法中。

```
a.A←XUY
b.A←Z
c.n← | A |
```

```
算法 Coin(A,n)
1）k← ┗n/3┛
2）将 A 中硬币分成 X,Y,Z 三个集合，使得 |X|=|Y|=k, |Z|=n-2k
3）if W(X)!=W(Y) ,W(X), W(Y),分别为 X 或 Y 的重量
4）then___
5）else___
6）______
7）if n>2 then goto 1
8）if n=2 then 任取 A 中 1 枚硬币与拿走硬币比较，若不等，则它不合格；若相等，则 A 中剩下的硬币不合格
9）if n=1 then A 中硬币不合格
```

正确的填空顺序是（）。

A. b,c,a

B. c,b,a

C. c,a,b

D. a,b,c

2. 算法复杂度

16.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 在下列关于计算机算法的说法中，不正确的是（ ）。

A. 一个正确的算法至少要有一个输入

B. 算法的改进，在很大程度上推动了计算机科学与技术的进步

C. 判断一个算法的好坏的主要标准是算法的时间复杂性与空间复杂性

D. 目前仍然存在许多涉及到国计民生的重大课题，还没有找到能够在计算机上实施的有效算法

17.单选题 [NOIP 普及组 2011/NOIP 提高组 2011] 在使用高级语言编写程序时，一般提到的“空间复杂度”中的“空间”是指（ ）。

A. 程序运行时理论上所占的内存空间

B. 程序运行时理论上所占的数组空间

C. 程序运行时理论上所占的硬盘空间

D. 程序源文件理论上所占的硬盘空间

18.单选题 [NOIP 提高组 2011] 在使用高级语言编写程序时，一般提到的“空间复杂度”中的“空间”是指（ ）。

A. 程序运行时理论上所占的内存空间

B. 程序运行时理论上所占的数组空间

C. 程序运行时理论上所占的硬盘空间

D. 程序源文件理论上所占的硬盘空间

答案

1.F

2.F

3.B

4.B

5.C

6.B

7.A

8.B

9.B

10.B

11.D

12.B

13.C

14.A

15.D

16.A

17.A

18.A

GW 信奥 CSP 初赛习题集 6-2

算法——搜索算法

1. 枚举法

1. 枚举法概述

1.判断题 [GESP202403【3 级】] 下面 C++代码可以计算 1 到 100 的累加和，采用的是穷举法。

```
#include <iostream>
using namespace std;
int main()
{
    int i, sum=0;
    for(int i = 1; i <= 100 ; i++)
        sum += i;
    cout << sum << endl;
    return 0;
}
```

2.单选题 [GESP202312【8 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int main() {
    const int N = 30;
    int cnt = 0;
    for (int a = 1; a <= N; a++) {
        for (int b = a; a + b <= N; b++) {
            for (int c = b; a + b + c <= N; c++) {
                if (a * a + b * b == c * c) {
                    cnt++;
                }
            }
        }
    }
```

```
    cout << cnt << endl;
    return 0;
}
```

A. 3

B. 6

C. 11

D. 22

3.单选题 [GESP202403【8 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int main() {
    int cnt = 0;
    for (int x = 0; x <= 10; x++)
        for (int y = 0; y <= 10; y++)
            for (int z = 0; z <= 10; z++)
                if (x + y + z == 15)
                    cnt++;
    cout << cnt << endl;
    return 0;
}
```

A. 90

B. 91

C. 96

D. 100

4.单选题 [GESP202312【8 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int main() {
    int cnt = 0;
    for (int a = 1; a <= 10; a++)
        for (int b = 1; b <= 10; b++)
            for (int h = 1; h <= 10; h++)
                if ((a + b) * h == 20)
                    cnt++;
```

```
    cout << cnt << endl;
    return 0;
}
```

A. 12

B. 18

C. 36

D. 42

5.填空题 [NOIP 普及组 2011]（子矩阵）给输入一个 n1*m1 的矩阵 a，和 n2*m2 的矩阵 b，问 a 中是否存在子矩阵和 b 相等。 若存在，输出所有子矩阵左上角的坐标：若不存在输出“ There is no answer ”。

```
#include <iostream>
using namespace std;
const int SIZE = 50;
int n1, m1, n2, m2, a[SIZE][SIZE], b[SIZE][SIZE];
int main(){
    int i, j, k1, k2;
    bool good, haveAns;
    cin>>n1>>m1;
    for (i = 1; i <= n1; i++)
        for (j = 1; j <= m1; j++)
            cin>>a[i][j];
    cin>>n2>>m2;
    for (i = 1; i <= n2; i++)
        for (j = 1; j <= m2; j++)
            ①;
    haveAns = false;
    for (i = 1; i <= n1 - n2 + 1; i++)
        for (j = 1; j <= ②; j++){
            ③;
            for (k1 = 1; k1 <= n2; k1++)
                for(k2 = 1; k2 <= ④; k2++)  {
                    if (a[i + k1 - 1][j + k2 - 1] != b[k1][k2])
                        good = false;
                }

            if (good) {
                cout<<i<<' '<<j<<endl;
                ⑤;
```

```
                }
        }
        if (!haveAns)
            cout<<""There is no answer""<<endl;
        return 0;
}
```

6.单选题 [CSP-J2021]（矩形计数） 平面上有 n 个关键点，求有多少个四条边都和 x 轴或者 y 轴平行的矩形，满足四个顶点都是关键点。给出的关键点可能有重复，但完全重合的矩形只计一次。

试补全枚举算法。

```
#include <iostream>
using namespace std;

struct point {
    int x, y, id;
};

bool equals(point a, point b) {
    return a.x == b.x && a.y == b.y;
}

bool cmp(point a, point b) {
    return ①;
}

void sort(point A[], int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 1; j < n; j++) {
            if (cmp(A[j], A[j - 1])) {
                point t = A[j];
                A[j] = A[j - 1];
                A[j - 1] = t;
            }
        }
    }
}

int unique(point A[], int n) {
    int t = 0;
    for (int i = 0; i < n; i++) {
```

```
            if (②) {
                A[t++] = A[i];
            }
        }
        return t;
}

bool binary_search(point A[], int n, int x, int y) {
        point p;
        p.x = x;
        p.y = y;
        p.id = n;
        int a = 0, b = n - 1;
        while (a < b) {
            int mid = ③;
            if (④) {
                a = mid + 1;
            } else {
                b = mid;
            }
        }
        return equals(A[a], p);
}

const int MAXN = 1000;
point A[MAXN];

int main() {
        int n;
        cin >> n;
        for (int i = 0; i < n; i++) {
            cin >> A[i].x >> A[i].y;
            A[i].id = i;
        }
        sort(A, n);
        n = unique(A, n);
        int ans = 0;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (⑤&& binary_search(A, n, A[i].x, A[j].y) &&binary_search(A, n, A[j].x, A[i].y))
{
                    ans++;
                }
            }
        }
```

```
75        cout << ans << endl;
76        return 0;
77    }
```

①处应填（ ）

A. a.x != b.x ? a.x < b.x : a.id < b.id

B. a.x != b.x ? a.x < b.x : a.y < b.y

C. equals(a, b) ? a.id < b.id : a.x < b.x

D. equals(a, b) ? a.id < b.id : (a.x != b.x ? a.x < b.x : a.y < b.y)

②处应填（ ）

A. i == 0 || cmp(A[i], A[i - 1])

B. t == 0 || equals(A[i], A[t - 1])

C. i == 0 || !cmp(A[i], A[i - 1])

D. t == 0 || !equals(A[i], A[t - 1])

③处应填（ ）

A. b - (b - a) / 2 + 1

B. a + b + 1) >> 1

C. (a + b) >> 1

D. a + (b - a + 1) / 2

④处应填（ ）

A. !cmp(A[mid], p)

B. cmp(A[mid], p)

C. cmp(p, A[mid])

D. !cmp(p, A[mid])

⑤处应填（ ）

A. A[i].x == A[j].x

B. A[i].id < A[j].id

C. A[i].x == A[j].x && A[i].id < A[j].id

D. A[i].x < A[j].x && A[i].y < A[j].y

1.

A. a.x != b.x ? a.x < b.x : a.id < b.id

B. a.x != b.x ? a.x < b.x : a.y < b.y

C. equals(a, b) ? a.id < b.id : a.x < b.x

D. equals(a, b) ? a.id < b.id : (a.x != b.x ? a.x < b.x : a.y < b.y)

2.

A. i == 0 || cmp(A[i], A[i - 1])

B. t == 0 || equals(A[i], A[t - 1])

C. i == 0 || !cmp(A[i], A[i - 1])

D. t == 0 || !equals(A[i], A[t - 1])

3.

A. b - (b - a) / 2 + 1

B. (a + b + 1) >> 1

C. (a + b) >> 1

D. a + (b - a + 1) / 2

4.

A. !cmp(A[mid], p)

B. cmp(A[mid], p)

C. cmp(p, A[mid])

D. !cmp(p, A[mid])

5.

A. A[i].x == A[j].x

B. A[i].id < A[j].id

C. A[i].x == A[j].x && A[i].id < A[j].id

D. A[i].x < A[j].x && A[i].y < A[j].y

2. 枚举法时间复杂度

7.单选题 [CSP-S2022] 对于给定的 n，分析以下代码段对应的时间复杂度，其中最为准确的时间复杂度为（ ）。

```
int i, j, k = 0;
for (i = 0; i < n; i++) {
    for (j = 1; j < n; j*=2) {
        k = k + n / 2;
    }
}
```

A. O(n)

B. O(nlogn)

C. O(nsqrtn)

D. O(n^2)

8.单选题 [GESP202403【7 级】] 下面 count_triple 函数的时间复杂度为( )

```
int count_triple(int n){
    int cnt =0;
    for(int a=1;a<= n; a++)
        for(int b=a;a+b<= n; b++)
            for(int c=b;a+b+c<=n; c++)
                if(a*a+b*b==c*c)
                    cnt++;
    return cnt;
}
```

A.O(N)

B.O(N^2)

C.O(N^3)

D.O(N^4)

9.单选题 [GESP202406【7 级】] 下面 count_triple 函数的时间复杂度为( )。

```
int count_triple(int n) {
    int cnt = 0;
    for (int a = 1; a <= n; a++)
        for (int b = a; a + b <= n; b++) {
            int c = sqrt(a * a + b * b);
            if (a + b + c > n)
                break;
            if (a * a + b * b == c * c)
                cnt++;
```

```
            }
        return cnt;
}
```

A.O(n)

B.O(n^2)

C.O(n^3)

D.O(n^4)

3. 排列组合枚举

10.单选题 [GESP202406【8 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int main() {
    int cnt = 0;
    for (int x = 0; x <= 10; x++)
        for (int y = 0; y <= 10; y++)
            for (int z = 0; z <= 10; z++)
                if (x + y + z <= 15)
                    cnt++;
    cout << cnt << endl;
    return 0;
}
```

A. 90

B. 91

C. 710

D. 711

11.单选题 [GESP 样题【8 级】] 下面程序输出的 n 的值是（ ）

```
#include <iostream>
#include <string>
#include <cmath>
using namespace std;

int main()
```

```
{
    int a,b,c,d, n=0;
    for(a=1;a<5;a++)
        for(b=1;b<5;b++)
            for(c=1;c<5;c++)
                for(d=1;d<5;d++)
                    if(a!=b && a!=c && a!=d && b!=c && b!=d && c!=d)
                        if(a!=4)
                        {
                            cout << a*1000+b*100+c*10+d << "" "";
                            n++;
                        }
    cout << endl << n << endl;
    return 0;
}
```

A. 4

B. 12

C. 18

D. 24

12.填空题 [NOIP 普及组 2009]

```
#include <iostream>
using namespace std;
const int maxn = 50;
void getnext(char str[]) {
    int l = strlen(str), i, j, k, temp;
    k = l - 2;
    while (k >= 0 && str[k] > str[k + 1]) k--;
    i = k + 1;
    while (i < l && str[i] > str[k]) i++;
    temp = str[k];
    str[k] = str[i - 1];
    str[i - 1] = temp;
    for (i = l - 1; i > k; i--)
        for (j = k + 1; j < i; j++)
            if (str[j] > str[j + 1]) {
                temp = str[j];
                str[j] = str[j + 1];
                str[j + 1] = temp;
            }
    return;
```

```
}
int main() {
    char a[maxn];
    int n;
    cin >> a >> n;
    while (n > 0) {
        getnext(a);
        n--;
    }
    cout << a << endl;
    return 0;
}
```

输入：NOIP 3

13.填空题 [NOIP 普及组 2005]

```
#include <stdio.h>
#include <string.h>

int main() {
    char str[60];
    int len, i, j, chr[26];
    char mmin = 'z';
    scanf("%s", str);
    len = strlen(str);
    for (i = len - 1; i >= 1; i--)
        if (str[i - 1] < str[i]) break;
    if (i == 0) {
        printf("No result!\n");
        return 0;
    }
    for (j = 0; j < i - 1; j++) putchar(str[j]);
    memset(chr, 0, sizeof(chr));
    for (j = i; j < len; j++) {
        if (str[j] > str[i - 1] && str[j] < mmin)
            mmin = str[j];
        chr[str[j] - 'a']++;
    }
    chr[mmin - 'a']--;
    chr[str[i - 1] - 'a']++;
    putchar(mmin);
    for (i = 0; i < 26; i++)
        for (j = 0; j < chr[i]; j++)
```

```
                putchar(i + 'a');
        putchar('\n');
        return 0;
}
```

14.填空题 [NOIP 提高组 2005]

```
#include <stdio.h>
#include <string.h>
int main() {
        char str[60];
        int len, i, j, chr[26];
        char mmin = 'z';
        scanf("%s", str);
        len = strlen(str);
        for (i = len - 1; i >= 1; i--)
                if (str[i - 1] < str[i]) break;
        if (i == 0) {
                printf("No result!\n");
                return 0;
        }
        for (j = 0; j < i - 1; j++) putchar(str[j]);
        memset(chr, 0, sizeof(chr));
        for (j = i; j < len; j++) {
                if (str[j] > str[i - 1] && str[j] < mmin)
                        mmin = str[j];
                chr[str[j] - 'a']++;
        }
        chr[mmin - 'a']--;
        chr[str[i - 1] - 'a']++;
        putchar(mmin);
        for (i = 0; i < 26; i++)
                for (j = 0; j < chr[i]; j++)
                        putchar(i + 'a');
        putchar('\n');
        return 0;
}
```

15.填空题 [NOIP 提高组 2018]

```
#include <iostream>
using namespace std;
const int N = 110;
```

```
bool isUse[N];
int n, t;
int a[N], b[N];
bool isSmall() {
    for (int i = 1; i <= n; ++i)
        if (a[i] != b[i]) return a[i] < b[i];
    return false;
}
bool getPermutation(int pos) {
    if (pos > n) {
        return isSmall();
    }
    for (int i = 1; i <= n; ++i) {
        if (!isUse[i]) {
            b[pos] = i; isUse[i] = true;
            if (getPermutation(pos + 1)) {
                return true;
            }
            isUse[i] = false;
        }
    }
    return false;
}
void getNext() {
    for (int i = 1; i <= n; ++i) {
        isUse[i] = false;
    }
    getPermutation(1);
    for (int i = 1; i <= n; ++i) {
        a[i] = b[i];
    }

}
int main() {
    scanf(""%d%d"", &n, &t);
    for (int i = 1; i <= n; ++i) {
        scanf(""%d"", &a[i]);
    }
    for (int i = 1; i <= t; ++i) {
        getNext();
    }
    for (int i = 1; i <= n; ++i) {
        printf(""%d"", a[i]);
        if (i == n) putchar('\n'); else putchar(' ');
    }
```

```
    return 0;
}
```

输入 1：6 10 1 6 4 5 3 2

输入 2：6 200 1 5 3 4 2 6

16.填空题 [NOIP 普及组 2012]

（排列数）输入两个正整数 n，m（1<n<20,1<m<n），在 1~n 中任取 m 个数，按字典序从小到大输出所有这样的排列。例如：

输入： 3 2

输出：

1 2

1 3

2 1

2 3

3 1

3 2

```
#include<iostream>
#include<cstring>
using namespace std;
const int SIZE = 25;
bool used[SIZE];
int data[SIZE];
int n, m, i, j, k;
bool flag;
int main(){
    cin>>n>>m;
    memset(used, false, sizeof(used));
    for (i = 1; i <= m; i++){
        data[i] = i;
        used[i] = true;
    }
    flag = true;
    while (flag){
        for (i = 1; i <= m-1; i++)cout<<data[i]<<" ";
        cout << data[m] << endl;
        flag =①;
```

```
            for (i = m; i >= 1; i--){
                ②;
                for (j = data[i]+1; j <= n; j++)
                    if (!used[j]){
                        used[j] = true;
                        data[i] =③;
                        flag = true;
                        break;
                    }
                if (flag){
                    for (k = i+1; k <= m; k++)
                        for (j = 1; j <=④; j++)
                            if (!used[j]){
                                data[k] = j;
                                used[j] = true;
                                break;
                            }
                    ⑤;
                }
            }
        }
}
```

17.填空题 [NOIP 普及组 2006]（ 全排列 ）下面程序的功能是利用递归方法生成从 1 到 n(n<10) 的 n 个数的全部可能的排列（ 不一定 按升序输出 ）。例如，输入 3，则应该输出（ 每行输出 5 个排列 ）：

123 132 213 231 321 312

程序：

```
#include<stdio.h>
int n,a[10];
long count=0;
void perm(int k)
{
    int j,p,t;
    if(______ ①______)
    {
      count++;
      for(p=1;p<=n;p++)
        printf(""%1d"",a[p]); /* ""%1d"" 中是数字 1，不是字母 l */
      printf("" "");
      if(______ ②______)
```

```
            printf(""\n"");
        return;
    }
    for(j=k;j<=n;j++)
    {
        t=a[k];
        a[k]=a[j];
        a[j]=t;
        ______③______;
        t=a[k]; ______④______;
    }
}
int main()
{
    int i;
    printf(""Entry n:\n"");
    scanf(""%d"",&n);
    for(i=1;i<=n;i++)
      a[i]=i;
     ______⑤______;
}
```

18.填空题 [NOIP 提高组 2006] （选排列）下面程序的功能是利用递归方法生成从 1 到 n(n<10)的 n 个数中取 k(1<=k<=n)个数的 全部可能的排列（不一定按升序输出）。

例如，当 n=3，k=2 时， 应该输出（每行输出 5 个排列）：

12 13 21 23 32

31

程序：

```
#include <stdio.h>
int n,k,a[10];
long count=0;
void perm2(int j)
{
    int i,p,t;
    if( ① )
    {
        for(i=k;i<=n;i++)
    {
```

```
            count++;
            t=a[k]; a[k]=a[i]; a[i]=t;
            for( ② )
                printf(""%1d"",a[p]); /* ""%1d"" 中是数字 1，不是字母 l */
            printf("" "");
            t=a[k];a[k]=a[i];a[i]=t;
            if(count%5==0) printf(""\n"");
        }
        return;
        }
        for(i=j;i<=n;i++)
        {
            t=a[j];a[j]=a[i];a[i]=t;
            ③ ;
            t=a[j]; ④ ;
        }
}
int main()
{
        int i;
        printf(""\nEntryn,k (k<=n):\n"");
        scanf(""%d%d"",&n,&k);
        for(i=1;i<=n;i++) a[i]=i;
        ⑤ ;
}
```

19.填空题 [NOIP 提高组 2012]

（排列数）输入两个正整数 n，m（1<n<20,1<m<n），在 1~n 中任取 m 个数，按字典序从小到大输出所有这样的排列。例如：

输入： 3 2

输出：

1 2

1 3

2 1

2 3

3 1

3 2

```
#include<iostream>
#include<cstring>
using namespace std;
const int SIZE = 25;
bool used[SIZE];
int data[SIZE];
int n, m, i, j, k;
bool flag;
int main(){
    cin>>n>>m;
    memset(used, false, sizeof(used));
    for (i = 1; i <= m; i++){
        data[i] = i;
        used[i] = true;
    }
    flag = true;
    while (flag){
        for (i = 1; i <= m-1; i++)cout<<data[i]<<" ";
        cout << data[m] << endl;
        flag =①;
        for (i = m; i >= 1; i--){
            ②;
            for (j = data[i]+1; j <= n; j++)
                if (!used[j]){
                    used[j] = true;
                    data[i] =③;
                    flag = true;
                    break;
                }
            if (flag){
                for (k = i+1; k <= m; k++)
                    for (j = 1; j <=④; j++)
                        if (!used[j]){
                            data[k] = j;
                            used[j] = true;
                            break;
                        }
                ⑤;
            }
        }
    }
}
```

二、深搜广搜

20.单选题 [NOIP 普及组 2011] （ ）是一种选优搜索法，按选优条件向前搜索，以达到目标。当探索到某一步时，发现原先选择并不优或达不到目标，就退回一步重新选择。

A. 回溯法

B. 枚举法

C. 动态规划

D. 贪心

21.判断题 [GESP202309【6 级】] DFS 是深度优先算法的英文简写。

22.判断题 [GESP 样题【6 级】] 宽度优先搜索算法的英文简写是 BFS。

23.单选题 [NOIP 提高组 2011/NOIP 普及组 2011/CSP-S2020] 广度优先搜索时，需要用到的数据结构是（ ）。

A. 链表

B. 队列

C. 栈

D. 散列表

24.判断题 [GESP202406【6 级】/GESP202403【6 级】] 在深度优先搜索中，通常使用队列来辅助实现。

25.判断题 [GESP 样题【6 级】] 深度优先遍历一般需要借助数据结构栈来实现，广度优先遍历一般需要借助数据结构队列来实现。

26.判断题 [GESP202403【7 级】] 围棋游戏中，判断落下一枚棋子后是否会提掉对方的子，可以使用泛洪算法来实现。

27.判断题 [GESP202406【7 级】] 泛洪算法的递归方法容易造成溢出，因此大的二维地图算法中，一般不用递归方法。

28.判断题 [GESP202406【4 级】] 一个一维数组，至少含有一个自然数 N，是一个合法的数列。可以在一维数组末尾加入一个自然数 M，M 不能超过一维数组末尾元素的一半，形成一个新的合法的一维数组，如果 N=6，那么可以有 6 个不同的合法数组。

29.单选题 [GESP202403【7 级】] 下面的程序属于哪种算法( )。

```
void queen(int n) {
    int pos[8];
    for (int i = 0; i < 8; i++) {
        pos[n] = i;
        bool attacked = false;
        for (int j = 0; j < n; j++) {
            if (pos[n] == pos[j] || pos[n] + n == pos[j] + j || pos[n] - n == pos[j] - j) {
                attacked = true;
                break;
            }
        }
        if (attacked)
            continue;
        if (n == 7) {
            return;
        } else {
            queen(n + 1);
        }
    }
}
```

A. 贪心算法

B. 动态规划

C. 深度优先搜索

D. 广度优先搜索

30.填空题 [NOIP 提高组 2007] （连续邮资问题）某国发行了 n 种不同面值的邮票，并规定每封信最多允许贴 m 张邮票，在这些约束下，为了能贴出{1,2,3,…,maxvalue}

连续整数集合的所有邮资，并使 maxvalue 的值最大，应该如何设计各邮票的面值？例如，当 n=5,m=4 时，面值设计为 {1,3,11,15,32}，可使 maxvalue 达到最大值 70（或者说，用这些面值的 1 至 4 张邮票可以表示不超过 70 的所有邮资，但无法表示邮资 71。而用其他面值的 1 至 4 张邮票如果可以表示不超过 k 的所有邮资，必有 k≤70）。下面是用递归回溯求解连续邮资问题的程序。数组 x[1:n] 表示 n 种不同的邮票面值，并约定各元素按下标是严格递增的。数组 bestx[1:n] 存放使 maxvalue 达到最大值的邮票面值（最优解），数组 y[maxl] 用于记录当前已选定的邮票面值 x[1:i] 能贴出的各种邮资所需的最少邮票张数。请将程序补充完整。

```
#include <stdio.h>
#define NN 20
#define maxint 30000
#define maxl 500 /*邮资的最大值*/
int n, m, bestx[NN], x[NN], y[maxl], maxvalue = 0;
void result()
{
 //输出结果： 最大值：maxvalue 及最优解：bestx[1:n]（略）
}


void backtrace( int i, int r )
{
 int j, k, z[maxl];
 for ( j = 0; j <= ①; j++ )
  if ( y[j] < m )
   for ( k = 1; k <= m - y[j]; k++ )
    if ( y[j] + k <= y[②] )
     y[③] = y[j] + k;
 while ( y[r] < maxint )
  r++;
 if ( i > n )
 {
  if ( r - 1 > maxvalue )
  {
   maxvalue = ④;
   for ( j = 1; j <= n; j++ )
    bestx[j] = x[j];
  }
  return;
 }
 for ( k = 0; k < maxl; k++ )
  z[k] = y[k];
```

```
  for ( j = ⑤; j <= r; j++ )
  {
    x[i] = j;
    ⑥;
    for ( k = 0; k < maxl; k++ )
      y[k] = z[k];
  }
}


void main()
{
  int j;
  printf( ""input n,m:\n"" );
  scanf( “ % d % d ”, &n, &m );
  for ( j = 1; j < maxl; j++ )
    y[j] = maxint;
  y[0] = 0; x[0] = 0; x[1] = 1;
  backtrace( 2, 1 );
  result();
}
```

31.填空题 [NOIP 普及组 2009] (国王放置) 在 n*m 的棋盘上放置 k 个国王， 要求 k 个国王互相不攻击， 有多少种不同 的 放 置 方 法 。 假 设 国 王 放 置 在 第 (x,y) 格 ， 国 王 的 攻 击 的 区 域 是 :(x-1,y-1), (x-1,y),(x-1,y+1),(x,y-1),(x,y+1),(x+1,y-1),(x+1,y),(x+1,y+1) 。读入三个数 n,m,k，输出答案。题目 利用回溯法求解。棋盘行标号为 0~n-1，列标号为 0~m-1。

```
#include <stdio.h>
#include <string.h>
int n,m,k,ans;
int hash[5][5];
void work(int x,int y,int tot)
{
      int i,j;
      if (tot==k)
      {
            ans++;
            return;
      }
do
{
```

```
    while (hash[x][y])
    {
        y++;
        if (y==m)
        {
            x++;
            y= ① ;
        }
        if (x==n)
        return;
    }
    for (i=x-1;i<=x+1;i++)
    if (i>=0&&i<n)
        for (j=y-1;j<=y+1;j++)
    if (j>=0&&j<m)
        ② ;
    ③ ;
    for (i=x-1;i<=x+1;i++)
    if (i>=0&&i<n)
    for (j=y-1;j<=y+1;j++)
    if (j>=0&&j<m)
        ④ ;
    y++;
    if (y==m)
    {
        x++;
        y=0;
    }
    if (x==n)
    return;
}
while (1);
}
int main()
{
    scanf(""%d%d%d"",&n,&m,&k);
    ans=0;
    memset(hash,0,sizeof(hash));
    ⑤ ;
    printf(""%d\n"",ans);
    return 0;
}
```

32.填空题 [NOIP 提高组 2009]（ 寻找等差数列 ）有一些长度相等的等差数列（数列中每个数都为 0~59 的整数），设 长度均为 L，将等差数列中的所有数打乱顺序放在一起。现在给你这些打乱后的数，问原先， L 最大可能为多大？先读入一个数 n（1<=n<=60），再读入 n 个数，代表打乱后的数。输出等 差数列最大可能长度 L。

```
#include <iostream>
using namespace std;
int hash[60];
int n, x, ans, maxnum;
int work(int now)
{
    int first, second, delta, i;
    int ok;
    while ( ① && !hash[now])
    ++now;
    if (now > maxnum)
        return 1;
    first = now;
    for (second = first; second <= maxnum; second++)
    if (hash[second])
    {
        delta = ② ;
        if (first + delta * ③ > maxnum)
        break;
        if (delta == 0)
            ok = ( ④ );
        else{
        ok = 1;
        for (i = 0; i < ans; i++)
        ok = ⑤ && (hash[first+delta*i]);
        }
        if (ok)
        {
            for (i = 0; i < ans; i++)
            hash[first+delta*i]--;
            if (work(first))
            return 1;
            for (i = 0; i < ans; i++)
            hash[first+delta*i]++;
        }
    }
    return 0;
}
int main()
```

```
{
    int i;
    memset(hash, 0, sizeof(hash));
    cin >> n;
    maxnum = 0;
    for (i = 0; i < n; i++){
        cin >> x;
        hash[x]++;
        if (x > maxnum)
        maxnum = x;
    }
    for (ans = n; ans >= 1; ans--)
    if ( n%ans==0 && ⑥ ) {
        cout << ans << endl;
        break;
    }
    return 0;
}
```

33.填空题 [NOIP 普及组 2010/NOIP 提高组 2010] （过河问题） 在一个月黑风高的夜晚，有一群人在河的右岸，想通过唯一的一根独木桥 走到河的左岸。 在这伸手不见五指的黑夜里， 过桥时必须借助灯光来照明， 很不幸的是，他 们只有一盏灯。 另外，独木桥上最多承受两个人同时经过，否则将会坍塌。 每个人单独过桥 都需要一定的时间，不同的人需要的时间可能不同。两个人一起过桥时，由于只有一盏灯，所以需要的时间是较慢的那个人单独过桥时所花的时间。现输入 n（ 2≤n≤100）和这 n 个 人单独过桥时需要的时间，请计算总共最少需要多少时间，他们才能全部到达河的左岸。例如，有 3 个人甲、乙、丙，他们单独过桥的时间分别为 1、 2、4，则总共最少需要 的时间为 7 。具体方法是：甲、乙一起过桥到河的左岸，甲单独回到河的右岸将灯带回，然 后甲、丙再一起过桥到河的左岸，总时间为 2+1+4=7 。

```
#include<iostream>
using namespace std;
const int SIZE = 100;
const int INFINITY = 10000;
const bool LEFT = true;
const bool RIGHT = false;
const bool LEFT_TO_RIGHT = true;
const bool RIGHT_TO_LEFT = false;
int n, hour[SIZE];
```

```
bool pos[SIZE];
int max(int a, int b) {
    if (a > b)
        return a;
    else
        return b;
}
int go(bool stage) {
    int i, j, num, tmp, ans;
    if (stage == RIGHT_TO_LEFT) {
        num = 0;
        ans = 0;
        for (i = 1; i <= n; i++)
            if (pos[i] == RIGHT) {
                num++;    //人数加 1
                if (hour[i] > ans)
                    ans = hour[i];
            }
        if ( ① __________      )
            return ans;
        ans = INFINITY;
        for (i = 1; i <= n - 1; i++)
            if (pos[i] == RIGHT)
                for (j = i + 1; j <= n; j++)
                    if (pos[j] == RIGHT) {
                        pos[i] = LEFT;
                        pos[j] = LEFT;
                        tmp = max(hour[i], hour[j]) +   ② __________) ;
                        if (tmp < ans)
                            ans = tmp;
                        pos[i] = RIGHT;
                        pos[j] = RIGHT;
                    }
        return ans;
    }
    if (stage == LEFT_TO_RIGHT) {
        ans = INFINITY;
        for (i = 1; i <= n; i++)
            if (   ③_________) {
                pos[i] = RIGHT;
                tmp =   ④ _________;
                if (tmp < ans)
                    ans = tmp;
                            ⑤__________    ;

```

```
                }
            return ans;
        }
        return 0;
}
int main() {
        int i;
        cin>>n;
        for (i = 1; i <=n; i++) {
            cin>>hour[i];
            pos[i] = RIGHT;
        }
        cout<<go(RIGHT_TO_LEFT)<<endl;
        return 0;
}
```

34.单选题 [CSP-S2022] （容器分水） 有两个容器，容器 1 的容量为为 a 升，容器 2 的容量为 b 升；同时允许下列的三种操作，分别为：

FILL(i)：用水龙头将容器 i(i∈1,2)灌满水；

DROP(i)：将容器 i 的水倒进下水道；

POUR(i,j)：将容器 i 的水倒进容器 j（完成此操作后，要么容器 j 被灌满，要么容器 i 被清空）。

求只使用上述的两个容器和三种操作，获得恰好 c 升水的最少操作数和操作序列。上述 a、b、c 均为不超过 100 的正整数，且 c≤max{a,b}。

试补全程序。

```
#include <bits/stdc++.h>
using namespace std;
const int N = 110;

int f[N][N];
int ans;
int a, b, c;
int init;

int dfs(int x, int y) {
        if (f[x][y] != init)
            return f[x][y];
        if (x == c || y == c)
```

```
14              return f[x][y] = 0;
15          f[x][y] = init - 1;
16          f[x][y] = min(f[x][y], dfs(a, y) + 1);
17          f[x][y] = min(f[x][y], dfs(x, b) + 1);
18          f[x][y] = min(f[x][y], dfs(0, y) + 1);
19          f[x][y] = min(f[x][y], dfs(x, 0) + 1);
20          int t = min(a - x, y);
21          f[x][y] = min(f[x][y], ①);
22          t = min(x, b - y);
23          f[x][y] = min(f[x][y], ②);
24          return f[x][y];
25      }
26
27      void go(int x, int y) {
28          if (③)
29              return;
30          if (f[x][y] == dfs(a, y) + 1) {
31              cout << ""FILL(1)"" << endl;
32              go(a, y);
33          } else if (f[x][y] == dfs(x, b) + 1) {
34              cout << ""FILL(2)"" << endl;
35              go(x, b);
36          } else if (f[x][y] == dfs(0, y) + 1) {
37              cout << ""DROP(1)"" << endl;
38              go (0, y);
39          } else if (f[x][y] == dfs(x, 0) + 1) {
40              cout << ""DROP(2)"" << endl;
41              go(x, 0);
42          } else {
43              int t = min(a - x, y);
44              if(f[x][y] == ④) {
45                  cout << ""POUR(2,1)"" << endl;
46                  go(x + t, y - t);
47              } else {
48                  t = min(x, b - y);
49                  if (f[x][y] == ⑤) {
50                      cout << ""POUR(1,2)"" << endl;
51                      go(x - t, y + t);
52                  } else
53                      assert(0);
54              }
55          }
56      }
57
58      int main() {
```

```
59        cin >> a >> b >> c;
60        ans = 1 << 30;
61        memset(f, 127, sizeof f);
62        init = **f;
63        if ((ans = dfs (0, 0)) == init - 1)
64              cout << ""impossible"";
65        else {
66              cout << ans << endl;
67              go (0, 0);
68        }
69  }
```

①~⑤处应填（ ）

1.

A. dfs(x + t, y - t) + 1

B. dfs(x + t, y - t) - 1

C. dfs(x - t, y + t) + 1

D. dfs(x - t, y + t) - 1

2.

A. dfs(x + t, y - t) + 1

B. dfs(x + t, y - t) - 1

C. dfs(x - t, y + t) + 1

D. dfs(x - t, y + t) - 1

3.

A. x == c || y == c

B. x == c && y == c

C. x >= c || y >= c

D. x >= c && y >= c

4.

A. dfs(x + t, y - t) + 1

B. dfs(x + t, y - t) - 1

C. dfs(x - t, y + t) + 1

D. dfs(x - t, y + t) - 1

5.

A. dfs(x + t, y - t) + 1

B. dfs(x + t, y - t) - 1

C. dfs(x - t, y + t) + 1

D. dfs(x - t, y + t) - 1

35.单选题 [CSP-J2020]

```
#include <algorithm>
#include <iostream>
using namespace std;

int n;
int d[50][2];
int ans;

void dfs(int n, int sum) {
    if (n == 1) {
        ans = max(sum, ans);
        return;
    }
    for (int i = 1; i < n; ++i) {
        int a = d[i - 1][0], b = d[i - 1][1];
        int x = d[i][0], y = d[i][1];
        d[i - 1][0] = a + x;
        d[i - 1][1] = b + y;
        for (int j = i; j < n - 1; ++j) {
            d[j][0] = d[j + 1][0];
            d[j][1] = d[j + 1][1];
        }
        int s = a + x + abs(b - y);
        dfs(n - 1, sum + s);
        for (int j = n - 1; j > i; --j) {
            d[j][0] = d[j - 1][0];
            d[j][1] = d[j - 1][1];
        }
        d[i - 1][0] = a;
        d[i - 1][1] = b;
        d[i][0] = x;
        d[i][1] = y;
    }
```

```
}

int main() {
    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> d[i][0];
    for (int i = 0; i < n; ++i)
        cin >> d[i][1];
    ans = 0;
    dfs(n, 0);
    cout << ans << endl;
    return 0;
}
```

假设输入的 n 是不超过 50 的正整数，d[i][0]、d[i][1] 都是不超过 10000 的正整数，完成下面的判断题和单选题：

判断题

若输入 n 为 0，此程序可能会死循环或发生运行错误。（ ）

若输入 n 为 20，接下来的输入全为 0，则输出为 0。（ ）

输出的数一定不小于输入的 d[i][0] 和 d[i][1] 的任意一个。（ ）

单选题

若输入的 n 为 20，接下来的输入是 20 个 9 和 20 个 0，则输出为（ ）。

若输入的 n 为 30，接下来的输入是 30 个 0 和 30 个 5，则输出为（ ）。

若输入的 n 为 15，接下来的输入是 15 到 1，以及 15 到 1，则输出为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 1890

B. 1881

C. 1908

D. 1917

5.

A. 2000

B. 2010

C. 2030

D. 2020

6.

A. 2440

B. 2220

C. 2240

D. 2420

36.单选题 [CSP-J2022] 现有用字符标记像素颜色的 8×8 图像。颜色填充的操作描述如下：给定起始像素的位置待填充的颜色，将起始像素和所有可达的像素（可达的定义：经过一次或多次的向上、下、左、右四个方向移动所能到达且终点和路径上所有像素的颜色都与起始像素颜色相同），替换为给定的颜色。

试补全程序。

```
#include<bits/stdc++.h>
using namespace std;

const int ROWS = 8;
const int COLS = 8;

struct Point {
    int r, c;
    Point(int r, int c): r(r), c(c) {}
};

bool is_valid(char image[ROWS][COLS], Point pt,
                int prev_color, int new_color) {
```

```
    int r = pt.r;
    int c = pt.c;
    return (0 <= r && r < ROWS && 0 <= c && c < COLS &&
            ① && image[r][c] != new_color);
}

void flood_fill(char image[ROWS][COLS], Point cur, int new_color) {
    queue<Point> queue;
    queue.push(cur);

    int prev_color = image[cur.r][cur.c];
    ②;

    while (!queue.empty()) {
        Point pt = queue.front ();
        queue.pop ();

        Point points[4] = {③, Point(pt.r - 1, pt.c),
                           Point(pt.r, pt.c + 1), Point(pt.r, pt.c - 1)};
        for (auto p ; points) {
            if (is_valid(image, p, prev_color, new_color)) {
                ④;
                ⑤;
            }
        }
    }
}

int main() {
    char image[ROWS][COLS] = {{'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g'},
                              {'g', 'g', 'g', 'g', 'g', 'g', 'r', 'r'},
                              {'g', 'r', 'r', 'g', 'g', 'r', 'g', 'g'},
                              {'g', 'b', 'b', 'b', 'b', 'r', 'g', 'r'},
                              {'g', 'g', 'g', 'b', 'b', 'r', 'g', 'r'},
                              {'g', 'g', 'g', 'b', 'b', 'b', 'b', 'r'},
                              {'g', 'g', 'g', 'g', 'g', 'b', 'g', 'g'},
                              {'g', 'g', 'g', 'g', 'g', 'b', 'b', 'g'}};

    Point cur(4, 4);
    char new_color = 'y';

    flood_fill(image, cur, new_color);

    for (int r = 0; r < ROWS; r++) {
        for (int c = 0; c < COLS; c++) {
```

```
                cout << image[r][c] << '';
            }
            cout << endl;
        }
//输出:
// g g g g g g g g
// g g g g g g r r
// g r r g g r g g
// g y y y y r g r
// g g g y y r g r
// g g g y y y y r
// g g g g g y g g
// g g g g g y y g

        return 0;
}
```

①~⑤处应填（ ）

1.

A. image[r][c] == prev_color

B. `image[r][c] != prev_color

C. image[r][c] == new_color

D. image[r][c] != new_color

2.

A. image[cur.r+1][cur.c] = new_color

B. image[cur.r][cur.c] = new_color

C. image[cur.r][cur.c+1] = new_color

D. image[cur.r][cur.c] = prev_color

3.

A. Point(pt.r, pt.c)

B. Point(pt.r, pt.c+1)

C. Point(pt.r+1, pt.c)

D. Point(pt.r+1, pt.c+1)

4.

A. prev_color = image[p.r][p.c]

B. new_color = image[p.r][p.c]

C. image[p.r][p.c] = prev_color

D. image[p.r][p.c] = new_color

5.

A. queue.push(p)

B. queue. push (pt)

C. queue.push(cur)

D. queue. push(Point (ROWS, COLS))

答案

1.F

2.A

3.B

4.B

5.

1. 正确答案： cin>>b[i][j]
2. 正确答案： m1 - m2 + 1
3. 正确答案： good = true
4. 正确答案： m2
5. 正确答案： haveAns = true

6.BDCBD

7.B

8.C

9.B

10.D

11.C

12.NPOI

13.zzzaaabbbcccy

14.zzzaaabbbcccy

15.

输出 1：2 1 3 5 6 4

输出 2：3 2 5 6 1 4

16.

1. 正确答案：false
2. 正确答案：used[data[i]] = false
3. 正确答案：j
4. 正确答案：n
5. 正确答案：break

17.

1. 正确答案：k==n
2. 正确答案：count%5==0
3. 正确答案：perm(k+1)
4. 正确答案：a[k]=a[j];a[j]=t
5. 正确答案：perm(1)

18.

1. 正确答案：j=k 或 k=j
2. 正确答案：p:=1 to k
3. 正确答案：perm2(j+1)
4. 正确答案：a[j]:=a;a:=t
5. 正确答案：perm2(1)

19.

1. 正确答案：false
2. 正确答案：used[data[i]] = false
3. 正确答案：j
4. 正确答案：n
5. 正确答案：break

20.A

21.T

22.T

23.B

24.F

25.T

26.T

27.T

28.T

29.C

30.

正确答案： x[i-2]*(m-1)

正确答案： j+x[i-1]*k

正确答案： j+x[i-1]*k

正确答案： r-1

正确答案： x[i-1]+1

正确答案： backtrace(i+1,r)

31.

1.正确答案： 0

2.正确答案： hash[i][j]++ / hash[i][j]= hash[i][j]+1 / ++hash[i][j]

3.正确答案： work(x,y,tot+1)

4.正确答案： hash[i][j]-- / hash[i][j]= hash[i][j]-1 / --hash[i][j]

5.正确答案： work(0,0,0)

32.

1.正确答案： now<=maxnum / !(now>maxnum)

2.正确答案： second-first

3.正确答案： (ans-1)

4.正确答案： hash[first]>=ans / hash[second]>=ans / hash[first+delta]>=ans

5.正确答案： ok

6.正确答案： work(0) / work(0)==1 / work(0)>0

33.

1.正确答案： num <= 2 / num < 3 / num == 2

2.正确答案： go(LEFT_TO_RIGHT)

3.正确答案： pos[i] == LEFT / LEFT == pos[i]

4.正确答案： hour[i] + go(RIGHT_TO_LEFT) / go(RIGHT_TO_LEFT) + hour[i]

5.正确答案： pos[i] = LEFT

34.ACAAC

35.BABBCC

36.ABCDA

GW 信奥 CSP 初赛习题集 6-3

算法——分治算法

1. 分治算法概述

1.判断题 [GESP202403【5 级】] 分治算法的核心思想是将一个大问题分解成多个相同或相似的子问题进行解决，最后合并得到原问题的解。

2.单选题 [NOIP 普及组 2012] （ ）就是把一个复杂的问题分成两个或者更多的相同或相似的子问题，再把子问题分成更小的子问题……直到最后的子问题可以简单的直接求解。而原问题的解就是子问题解的并。

A. 动态规划

B. 贪心

C. 分治

D. 搜索

3.单选题 [GESP202406【5 级】] 关于分治算法，以下哪个说法正确？

A. 分治算法将问题分成子问题，然后分别解决子问题，最后合并结果。

B. 归并排序不是分治算法的应用。

C. 分治算法通常用于解决小规模问题。

D. 分治算法的时间复杂度总是优于 O(nlong(n))。

4.单选题 [GESP 样题【5 级】/GESP 样题【6 级】] 关于分治算法，下列说法错误的是（ ）。

A. 分治算法的核心思想是分而治之，即把问题转化为多个规模更小的子问题求解。

B. 分治算法可以不使用递归实现。

C. 分治算法的时间复杂度是 O(logN)，其中 N 表示问题的规模。

D. 分治算法通常较容易在多核处理器上实现加速

5.不定项选择题 [NOIP 提高组 2016] 下列算法中运用分治思想的有（ ）。

A. 快速排序

B. 归并排序

C. 冒泡排序

D. 计数排序

6.单选题 [GESP 样题【5 级】] 下列哪个算法并没有体现分治思想？（ ）。

A. 二分查找

B. 埃氏筛法。

C. 归并排序。

D. 快速排序。

7.单选题 [GESP202403【5 级】] 归并排序的基本思想是（ ）。

A. 动态规划

B. 分治

C. 贪心算法

D. 回溯算法

8.判断题 [GESP202406【5 级】] 在进行全国人口普查时，将其分解为对每个省市县乡来进行普查和统计。这是典型的分治策略。

2. 二分法

1. 二分查找

9.判断题 [GESP202403【5 级】] 二分查找要求被搜索的序列是有序的，否则无法保证正确性。

10.判断题 [GESP 样题【5 级】] 二分查找法可以应用于有序序列（如升序排序的整数数组），也可以应用于无序序列（如乱序的整数数组）。

11.判断题 [GESP202312【5 级】] 小杨在生日聚会时拿一块 H*W 的巧克力招待来的 K 个小朋友，保证每位小朋友至少能获得一块相同大小的巧克力。那么小杨想分出来最大边长的巧克力可以使用二分法。

12.判断题 [GESP202312【8 级】] 给定 double 类型的变量 x ，且其值大于等于 0，我们可以通过二分法求出√x 的近似值。

13.判断题 [GESP202403【8 级】] 给定 double 类型的变量 x ，且其值大于等于 1，我们可以通过二分法求出 logx 的近似值。

14.单选题 [NOIP 普及组 2013]

```
#include <iostream>
using namespace std;
int main(){
    const int SIZE = 100;
    int n, f, i, left, right, middle, a[SIZE];
    cin>>n>>f;
    for (i = 1; i <= n; i++)
        cin>>a[i];
    left = 1;
    right = n;
    do {
        middle = (left + right) / 2;
        if (f <= a[middle])
            right = middle;
        else
            left = middle + 1;
    } while (left < right);
    cout<<left<<endl;
    return 0;
}
```

输入：

12 17

2 4 6 9 11 15 17 18 19 20 21 25

输出：

15.单选题 [GESP202312【5 级】] 下面 C++代码用于有序 list 的二分查找，有关说法错误的是（ ）。

```
int _binarySearch(vector<int> lst, int Low, int High, int Target)
{
    if (Low > High)
        return -1;
    int Mid = (Low + High) / 2;
    if (Target == lst[Mid])
        return Mid;
    else if (Target < lst[Mid])
        return _binarySearch(lst, Low, Mid - 1, Target);
    else
        return _binarySearch(lst, Mid + 1, High, Target);
}

int bSearch(vector<int> lst, int Val)
{
    return _binarySearch(lst, 0, lst.size(), Val);
}
```

A. 代码采用二分法实现有序 list 的查找

B. 代码采用分治算法实现有序 list 的查找

C. 代码采用递归方式实现有序 list 的查找

D. 代码采用动态规划算法实现有序 list 的查找

16.单选题 [GESP202406【5 级】] 根据下述二分查找法，在排好序的数组 1，3，6，9，17，31，39，52，61，79，81，90，96 中查找数值 82，和 82 比较的数组元素分别是（ ）。

```
int binary_search(vector<int>& nums, int target) {
    int left = 0;
    int right = nums.size() - 1;
```

```
    while (left <= right) {
        int mid = (left + right) / 2;
        if (nums[mid] == target) {
            return mid;
        } else if (nums[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1; // 如果找不到目标元素，返回-1
}
```

A. 52, 61, 81, 90

B. 52, 79, 90, 81

C. 39, 79, 90, 81

D. 39, 79, 90

17.填空题 [NOIP 普及组 2006] （寻找假币） 现有 80 枚硬币，其中有一枚是假币，其重量稍轻，所有真币的重量都相同，如果使用不带砝码的天平称重，最少需要称几次，就可以找出假币？你还要指出第 1 次的称重方法。请写出你的结果：

18.单选题 [CSP-J2019] 设有 100 个已排好序的数据元素，采用折半查找时，最大比较次数为（ ）

A. 7

B. 10

C. 6

D. 8

19.单选题 [NOIP 普及组 2008] 对有序数组{5, 13, 19, 21, 37, 56, 64, 75, 88，92，100}进行二分查找，成功查找元素 19 的查找长度（比较次数）是（ ）。

A. 1

B. 2

C. 3

D. 4

20.单选题 [NOIP 提高组 2008] 对有序数组{5, 13, 19, 21, 37, 56, 64, 75, 88, 92, 100}进行二分查找，等概率的情况下查找成功的平均查找长度（平均比较次数）是（ ）。

A. 35/11

B. 34/11

C. 33/11

D. 32/11

E. 34/10

21.单选题 [NOIP 普及组 2014] 设有 100 个数据元素，采用折半搜索时，最大比较次数为（ ）。

A. 6

B. 7

C. 8

D. 10

22.单选题 [NOIP 普及组 2009] 有一个由 4000 个整数构成的顺序表，假定表中的元素已经按升序排列，采用二分查找定位一个元素。则最多需要几次比较就能确定是否存在所查找的元素：

A. 11 次

B. 12 次

C. 13 次

D. 14 次

23.单选题 [GESP 样题【5 级】/GESP 样题【6 级】] 小明想了一个 1 到 100 之间的整数。你可以做多次猜测，每次猜测之后，如果你没有猜中，小明会告诉你，你猜的

数比他想的数大还是小。你希望你在运气最坏的情况下花费最少的次数猜中，请问你运气最坏的情况下会猜（ ）次？（包括最后猜中的那次）。

A. 5

B. 6

C. 7

D. 100

24.单选题 [GESP 样题【5 级】] 阅读下面 C++实现的二分查找代码，下列说法中错误的是（ ）。

```
int binarySearch(int *arr, int l, int r, int x) {
    if (r >= l) {
        int mid = l + (r - l) / 2;
        if (arr[mid] == x)
            return mid;
        else if (arr[mid] > x)
            return binarySearch(arr, l, mid - 1, x);
        else
            return binarySearch(arr, mid + 1, r, x);
    } else
        return -1;
}
```

A. 上面代码实现的二分查找，最少只需查找一次即可得到结果

B. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 0，则它实际被调用 3 次

C. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 3，则它实际被调用 3 次

D. 如果调用该函数在列表{2, 3, 4, 10, 12}中查找元素 10，则它实际被调用 3 次

25.单选题 [GESP202403【5 级】] 给定序列：1，3，6，9，17，31，39，52，61，79，81，90，96。使用以下代码进行二分查找查找元素 82 时，需要循环多少次，即最后输出的 times 值为（ ）。

```
int binarySearch(const std::vector<int>& arr, int target) {
    int left = 0;
    int right = arr.size() - 1;
    int times = 0;
    while (left <= right) {
```

```
            times++;
            int mid = left + (right - left) / 2;
            if (arr[mid] == target) {
                cout << times << endl;
                return mid;
            } else if (arr[mid] < target) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        cout << times << endl;
        return -1;
}
```

A. 2

B. 5

C. 3

D. 4

26.单选题 [GESP202312【5 级】] 在上题的 binarySearch 算法中，如果 lst 中有 N 个元素，其时间复杂度是（ ）。

```
int binarySearch(int *arr, int l, int r, int x) {
    if (r >= l) {
        int mid = l + (r - l) / 2;
        if (arr[mid] == x)
            return mid;
        else if (arr[mid] > x)
            return binarySearch(arr, l, mid - 1, x);
        else
            return binarySearch(arr, mid + 1, r, x);
    } else {
        return -1;
    }
}
```

A.O(N)

B.O(logN)

C.O(NlogN)

D.O（N^2）

27.单选题 [GESP202403【7 级】] 下面 search 函数的平均时间复杂度为( )。

```
int search(int n, int *p, int target) {
    int low = 0, high = n;
    while (low <= high) {
        int middle = (low + high) / 2;
        if (target == p[middle]) {
            return middle;
        } else if (target > p[middle]) {
            low = middle + 1;
        } else {
            high = middle - 1;
        }
    }
    return -1;
}
```

A.O(n)

B.O(log(n))

C.O(1)

1. 可能无法返回

2. 二分的综合应用

28.填空题 [NOIP 普及组 2015] (中位数）给定 n（n 为奇数且小于 1000）个整数，整数的范围在 0~m（$0<m<2^{31}$）之间，请使用二分法求这 n 个整数的中位数。所谓中位数，是指将这 n 个数排序之后，排在正中间的数。

```
#include <iostream>
using namespace std;
const int MAXN = 1000;
int n,i,lbound,rbound,mid,m,count;
int x[MAXN];
int main()
{
    cin >> n >> m;
    for(i = 0; i < n; i++)
```

```
        cin >> x[i];
    lbound = 0;rbound = m;
    while(①) {
        mid=(lbound+rbound)/2;
        ②;
        for(i = 0; i < n; i++)
        {
            if(③)
                ④;
        }
        if(count > n/2)
            lbound = mid + 1;
        else
            ⑤;
    }
    cout << rbound << endl;
    return 0;
}
```

29.填空题 [NOIP 普及组 2016] （郊游活动）有 n 名同学参加学校组织的郊游活动，已知学校给这 n 名同学的郊游总经费为 A 元，与此 同时第 i 位同学自己携带了 Mi 元。为了方便郊游，活动地点提供 B(≥n) 辆自行车供人租用，租用第 j 辆自行车的价格为 Cj 元，每位同学可以使用自己携带的钱或者学校的郊游经费，为了方便账务管理，每位同学只能为自己租用自行车，且不会借钱给他人，他们想知道最多有多少位同学能够租用到自行车。本题采用二分法。对于区间 [l, r]， 我们取中间点 mid 并判断租用到自行车的人数能否达到 mid。判断的过程是利用贪心算法实现的。

```
#include <iostream>
using namespace std;
#define MAXN 1000000

int n, B, A, M[MAXN], C[MAXN], l, r, ans, mid;

bool check(int nn) {
    int count = 0, i, j;
    i =  ①  ;
    j = 1;
    while (i <= n) {
        if( ② )
            count += C[j] - M[i];
```

```
            i++;
            j++;
        }
    return ③ ;
}

void sort(int a[], int l, int r) {
    int i = l, j = r, x = a[(l + r) / 2], y;
    while (i <= j) {
        while (a[i] < x) i++;
        while (a[j] > x) j--;
        if (i <= j) {
            y = a[i];a[i] = a[j]; a[j] = y;
            i++; j--;
        }
    }
    if (i < r) sort(a, i, r);
    if (l < j) sort(a, l, j);
}
int main() {
    int i;
    cin >> n >> B >> A;
    for (i = 1; i <= n; i++)
        cin >> M[i];
    for (i = 1; i <= B; i++)
        cin >> C[i];
    sort(M, 1, n);
    sort(C, 1, B);
    l = 0;
    r = n;
    while (l <= r) {
        mid = (l + r) / 2;
        if ( ④ ) {
            ans = mid;
            l = mid + 1;
        }
        else
            r = ⑤ ;
    }
    cout << ans << endl;
    return 0;
}
```

30.填空题 [NOIP 普及组 2017] （切割绳子）有 n 条绳子，每条绳子的长度已知且均为正整数。绳子可以以任意正整数长度切割，但不可以连接。现在要从这些绳子中切割出 m 条长度相同的绳段，求绳段的最大长度是多少。

输入：第一行是一个不超过 100 的正整数 n，第二行是 n 个不超过 10^6 的正整数，表示每条绳子的长度，第三行是一个不超过 10^8 的正整数 m。

输出 ：绳段的最大长度，若无法切割，输出 Failed。

```
#include <iostream>
using namespace std;
int n, m, i, lbound, ubound, mid, count;
int   len[100]; //绳子长度
int main()
{
      cin >> n;
      count = 0;
      for (i = 0; i < n; i++) {
            cin >> len[i];
            ①;
      }
      cin >> m;
      if (②) {
            cout << ""Failed"" << endl;
            return 0;
      }
      lbound = 1 ;
      ubound = 1000000 ;
      while (③) {
            mid = ④;
            count =0 ;
            for (i = 0; i < n; i++ )
                  ⑤;
            if (count < m)
                  ubound = mid - 1 ;
            else
                  lbound = mid ;
      }
      cout << lbound << endl;
      return 0;
}
```

31.填空题 [NOIP 普及组 2005/NOIP 提高组 2005] 木材加工

题目描述:

木材厂有一些原木，现在想把这些木头切割成一些长度相同的小段木头（木头有可能有剩余），需要得到的小段的数目是给定了的。当然，我们希望得到的小段越长越好，你的任务是计算能够得到的小段木头的最大长度。木头长度的单位是 cm。原木的长度都是正整数，我们要求切割得到的小段木头的长度也是正整数。

输入:第一行是两个正整数 N 和 K(1 ⩽ N ⩽ 10000，1 ⩽ K ⩽ 10000)，N 是原木的数目，K 是需要得到的小段的数目。接下来的 N 行，每行有一个 1 到 10000 之间的正整数，表示一根原木的长度。

输出:输出能够切割得到的小段的最大长度。如果连 1cm 长的小段都切不出来，输出”0”。

输入样例:

3 7

232

124

456

输出样例:

114

```
#include <stdio.h>
int n, k, len[10000];
int isok(int t) {
    int num = 0, i;
    for (i = 0; i < n; i++) {
        if (num >= k) break;
        num += len[i] / t;
    }
    if (num >= k) return 1;
    else return 0;
}

int main() {
    int i, left, right, mid;
    scanf("%d%d", &n, &k);
    right = 0;
    for (i = 0; i < n; i++) {
        scanf("%d", &len[i]);
```

```
            if (right < len[i]) right = len[i];
        }
        right++;
        left = 0;
        while (left + 1 < right) {
            mid = (left + right) / 2;
            if (isok(left) == 0) right = mid;
            else left = mid;
        }
        printf("%d ", left);
        return 0;
}
```

32.单选题 [CSP-S2023]

```
#include <vector>
#include <algorithm>
#include <iostream>

using namespace std;

bool f0(vector<int> &a, int m, int k) {
    int s = 0;
    for (int i = 0, j = 0; i < a.size(); i++) {
        while (a[i] - a[j] > m)
            j++;
        s += i - j;
    }
    return s >= k;
}

int f(vector<int> &a, int k) {
    sort(a.begin(), a.end());

    int g = 0;
    int h = a.back() - a[0];
    while (g < h) {
        int m = g + (h - g) / 2;
        if (f0(a, m, k)) {
            h = m;
        }
        else {
            g = m + 1;
        }
```

```
        }

        return g;
}

int main() {
        int n, k;
        cin >> n >> k;
        vector<int> a(n, 0);
        for (int i = 0; i < n; i++) {
                cin >> a[i];
        }
        cout << f(a, k) << endl;
        return 0;
}
```

假设输入总是合法的且 1≤ai≤10^8,n<=10000,1≤k≤n(n−1)/2，完成下面的判断题和单选题:

判断题

将第 24 行的 m 改为 m－1，输出有可能不变，而剩下情况为少 1。( )

将第 22 行的 g + (h－g) / 2 改为 (h + g) >> 1，输出不变。( )

当输入为 5 7 2 -4 5 1 -3，输出为 5。( )

单选题

设 a 数组中最大值减最小值加 1 为 A，则 f 函数的时间复杂度为（ ）。

将第 10 行中的 > 替换为 >=，那么原输出与现输出的大小关系为（ ）。

当输入为 5 8 2 -5 3 8 -12，输出为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. O(nlogA)

B. O((n^2)logA)

C. O(nlog(nA))

D. O(nlogn)

5.

A. 一定小于

B. 一定小于等于且不一定小于

C. 一定大于等于且不一定大于

D. 以上三种情况都不对.

6.

A. 13

B. 14

C. 8

D. 15

33.单选题 [CSP-J2023] (寻找被移除的元素)问题: 原有长度为 n+1 公差为 1 的等差数列，将数列输到程序的数组时移除了一个元素，导致长度为 n 的连续数组可能不再连续，除非被移除的是第一个或最后一个元素。需要在数组不连续时，找出被移除的元素。试补全程序。

```
#include <iostream>
#include <vector>
using namespace std;
int find_missing(vector<int>& nums) {
    int left = 0, right = nums.size() - 1;
    while (left < right){
        int mid = left + (right - left) / 2;
        if (nums[mid] == mid + ①) {
            ②;
        } else {
            ③;
        }
    }
```

```
        return ④;
}
int main() {
        int n;
        cin >> n;
        vector<int> nums(n);
        for (int i = 0; i < n; i++) cin >> nums[i];
        int missing_number = find_missing(nums);
        if (missing_number == ⑤) {
                cout << ""Sequence is consecutive"" << endl;
        }else{
                cout << ""Missing number is "" << missing_number << endl;
        }
        return 0;
}
```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. 1

B. nums[0]

C. right

D. left

2.

A. left=mid+1

B. right=mid-1

C. right=mid

D. left=mid

3.

A. left=mid+1

B. right=mid-1

C. right=mid

D. left=mid

4.

A. left+nums[0]

B. right+nums[0]

C. mid+nums[0]

D. right+1

5.

A. nums[0]+n

B. nums[0]+n-1

C. nums[0]+n+1

D. nums[n-1]

34.单选题 [CSP-J2022]

```
#include<iostream>
using namespace std;
int n, k;
int solve1()
{
    int l = 0, r = n;
    while (l <= r) {
        int mid = (l + r) / 2;
        if (mid * mid <= n) l = mid + 1;
        else r = mid - 1;
    }
    return l - 1;
}
double solve2(double x)
{
    if (x == 0) return x;
    for (int i = 0; i < k; i++)
        x = (x + n / x) / 2;
    return x;
}
int main()
{
    cin >> n >> k;
    double ans = solve2(solve1());
    cout << ans << ' ' << (ans * ans == n) << endl;
```

```
26      return 0;
27  }
```

假设 int 为 32 位有符号整数类型，输入的 n 是不超过 47000 的自然数、k 是不超过 int 表示范围的自然数，完成下面的判断题和单选题：

1.该算法最准确的时间复杂度分析结果为 O(log n + k)。

2.当输入为 9801 1 时，输出的第一个数为 99

3.对于任意输入的 n，随着所输入 k 的增大，输出的第二个数会变成 1。

4.该程序有存在缺陷。当输入的 n 过大时，第 12 行的乘法有可能溢出，因此应当将 mid 强制转换为 64 位整数再计算。

单选题

5.当输入为 2 1 时，输出的第一个数最接近()。

6.当输入为 3 10 时，输出的第一个数最接近()

7.当输入为 256 11 时，输出的第一个数()。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 1

B. 1.414

C. 1.5

D. 2

6.

A. 1.7

B. 1.732

C. 1.75

D. 2

7.

A. 等于 16

B. 接近但小于 16

C. 接近但大于 16

D. 前三种情况都有可能

35.填空题 [NOIP 提高组 2013]

```
#include <iostream>
#include <cstring>
using namespace std;
const int SIZE = 100;
int n, m, p, a[SIZE][SIZE], count;
void colour(int x, int y){
	count++;
	a[x][y] = 1;
	if ((x > 1) && (a[x - 1][y] == 0)) colour(x - 1, y);
	if ((y > 1) && (a[x][y - 1] == 0)) colour(x, y - 1);
	if ((x < n) && (a[x + 1][y] == 0)) colour(x + 1, y);
	if ((y < m) && (a[x][y + 1] == 0)) colour(x, y + 1);
}
int main(){
	int i, j, x, y, ans;
	memset(a, 0, sizeof(a));
	cin>>n>>m>>p;
	for (i = 1;i <= p;i++) {
		cin>>x>>y;
		a[x][y] = 1;
	}
	ans = 0;
	for (i = 1;i <= n;i++)
		for (j = 1;j <= m;j++)
			if (a[i][j] == 0) {
```

```
                    count = 0;
                    colour(i, j);
                    if (ans < count)
                        ans = count;
                }
    cout<<ans<<endl;
    return 0;
}
```

输入：

6 5 9

1 4

2 3

2 4

3 2

4 1

4 3

4 5

5 4

6 4

输出：_____

36.填空题 [NOIP 普及组 2012]

```
#include <iostream>
using namespace std;
int n, i, j, a[100][100];
int solve(int x, int y){
    int u, v;
    if (x == n)return a[x][y];
    u = solve(x + 1, y);
    v = solve(x + 1, y + 1);
    if (u > v)
        return a[x][y] + u;
    else
        return a[x][y] + v;
}
int main(){
```

```
    cin>>n;
    for (i = 1; i <= n; i++)
        for (j = 1; j <= i; j++)
            cin>>a[i][j];
    cout<<solve(1, 1)<<endl;
    return 0;
}
```

输入：

5 2

-1 4

2 -1 -2

-1 6 4 0

3 2 -1 5 8

输出：_____

37.填空题 [NOIP 普及组 2007]（棋盘覆盖问题）在一个 2^k×2^k 个方格组成的棋盘中恰有一个方格与其它方格不同（图中标记为 −1 的方格），称之为特殊方格。现 L 型（占 3 个小方格）纸片覆盖棋盘上除特殊方格的所有部分，各纸片不得重叠，于是，用到的纸片数恰好是 (4^k−1)/3。在下表给出的一个覆盖方案中，k=2，相同的 3 各数字构成一个纸片。下面给出的程序使用分治法设计的，将棋盘一分为四，依次处理左上角、右上角、左下角、右下角，递归进行。请将程序补充完整。

2 2 3 3

2 -1 1 3

4 1 1 5

4 4 5 5

```
#include <iostream.h>
#include <iomanip.h>
int board[65][65], tile; /* tile 为纸片编号 */

void chessboard(int tr, int tc, int dr, int dc, int size)
/* dr,dc 依次为特殊方格的行、列号 */
{
    int t, s;
```

```
    if (size == 1) {
        board[tr][tc] = tile++;
        return;
    }
    t = tile++;
    s = size / 2;
    if (dr < tr + s && dc < tc + s)
        chessboard(tr, tc, dr, dc, s);
    else {
        board[tr + s - 1][tc + s - 1] = t;
        chessboard(tr, tc, tr + s - 1, tc + s - 1, s);
    }
    if (dr < tr + s && dc >= tc + s)
        chessboard(tr, tc + s, dr, dc, s);
    else {
        board[tr + s - 1][tc + s] = t;
        chessboard(tr, tc + s, tr + s - 1, tc + s, s);
    }
    if (dr >= tr + s && dc < tc + s)
        chessboard(tr + s, tc, dr, dc, s);
    else {
        board[tr + s][tc + s - 1] = t;
        chessboard(tr + s, tc, tr + s, tc + s - 1, s);
    }
    if (dr >= tr + s && dc >= tc + s)
        chessboard(tr + s, tc + s, dr, dc, s);
    else {
        board[tr + s][tc + s] = t;
        chessboard(tr + s, tc + s, tr + s, tc + s, s);
    }
}

void prtl(int b[][65], int n)
{
    int i, j;
    for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++)
            cout << setw(3) << b[i][j];
        cout << endl;
    }
}

void main()
{
    int size, dr, dc;
```

```
54      cout << "input size(4/8/16/64):" << endl;
55      cin >> size;
56      cout << "input the position of special block(x,y):" << endl;
57      cin >> dr >> dc;
58      board[dr][dc] = -1;
59      tile++;
60      chessboard(1, 1, dr, dc, size);
61      prtl(board, size);
62  }
```

38.填空题 [NOIP 提高组 2004] 逻辑游戏

题目描述:

一个同学给了我一个逻辑游戏。他给了我图 1，在这个图上，每一段边界都已经进行了编号。我的任务是在图中画一条连续的曲线，使得这条曲线穿过每一个边界一次且仅穿过一次，而且曲线的起点和终点都在这整个区域的外面。这条曲线是容许自交的

对于图 1，我的同学告诉我画出这样的一条曲线（图 2）是不可能的，但是对于有的图形（比如图 3），画出这样一条曲线是可行的。对于给定的一个图，我想知道是否可以画出满足要求的曲线。

输入:

输入的图形用一个 n×n 的矩阵表示的。矩阵的每一个单元里有一个 0 到 255 之间（包括 0 和 255）的整数。处于同一个区域的单元里的数相同，相邻区域的数不同（但是不相邻的区域里的数可能相同）。

输入的第一行是 n（0<n<100）。以下的 n 行每行包括 n 个整数，分别给出对应的单元里的整数（这 n 个整数之间用空格分开）。图 4 给出了输入样例对应的图形。

输出:

当可以画出满足题意的曲线的时候，输出“YES”；否则，输出“NO”。

输入样例:

3

1 1 2

1 2 2

1 1 2

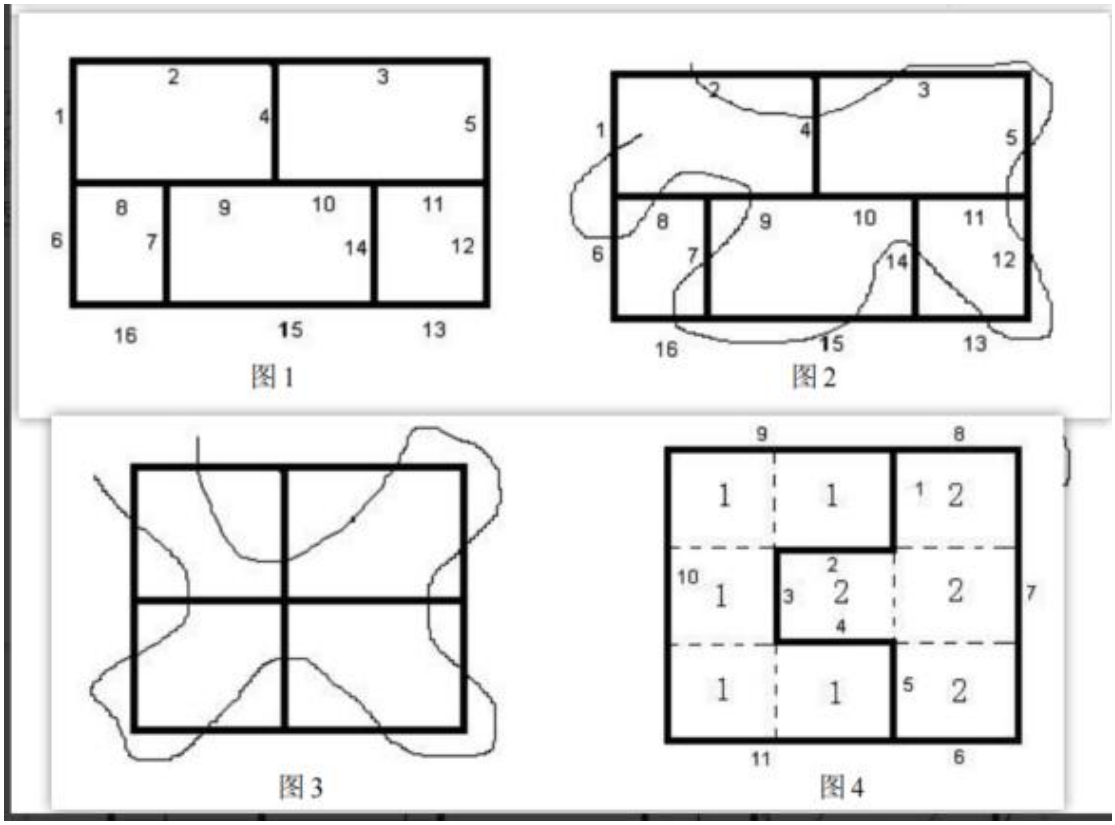
输出样例:

YES

程序:

```
#include <stdio.h>
#include <math.h>

int orig, n, ns, a[102][102], bun;
int d[] = {1, 0, -1, 0, 0, 1, -1, 0}; // Added -1, 0 to complete the array

void plimba(int x, int y) {
    int i, x1, y1;
    a[x][y] = -a[x][y];
    if (abs(a[x - 1][y]) != orig && (a[x - 1][y - 1] != a[x - 1][y] || abs(a[x][y - 1]) != orig))
ns++;
    if (abs(a[x + 1][y]) != orig && (a[x + 1][y - 1] != a[x + 1][y] || abs(a[x][y - 1]) != orig))
ns++;
    if (abs(a[x][y - 1]) != orig && (a[x - 1][y - 1] != a[x][y - 1] || abs(a[x - 1][y]) != orig))
ns++;
    if (abs(a[x][y + 1]) != orig && (a[x - 1][y + 1] != a[x][y + 1] || abs(a[x - 1][y]) != orig))
ns++;
    for (i = 0; i < 4; i++) {
        x1 = x + d[2 * i]; y1 = y + d[2 * i + 1];
        if (x1 >= 1 && x1 <= n && y1 >= 1 && y1 <= n && a[x1][y1] > -1)
            plimba(x1, y1);
    }
}

int main() {
    int i, j;
    bun = 1;
    scanf("%d", &n);
    for (i = 0; i <= n + 1; i++)
        for (j = 0; j <= n + 1; j++) a[i][j] = 0;
    a[0][0] = -1; a[n + 1][0] = -1;
    a[0][n + 1] = -1; a[n + 1][n + 1] = -1;
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++) scanf("%d", &(a[i][j]));
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++) {
            if (a[i][j] > -1) {
                ns = 0;
                orig = a[i][j];
                plimba(i, j);
                if (ns % 2 == 1) bun = 0;
```

```
                }
            }
        if (bun) printf("YES\n"); else printf("NO\n");
        return 0;
}
```



39.单选题 [CSP-S2023] (最大值之和)给定整数序列 ao,a_1 ,a_2 ……an-1,求该序列所有非空连续子序列的最大值之和。上述参数满足 1≤n≤10^5 和 1≤ai≤10^8。一个序列的非空连续子序列可以用两个下标 l 和 r(其中 0≤l≤r≤n)表示，对应的序列为 al,al+1,……ar。两个非空连续子序列不同，当且仅当下标不同。

例如，当原序列为[1,2,1,2]时，要计算子序列 [1],[2],[1],[2],[1,2],[2,1],[1,2],[1,2,1],[2,1,2],[1,2,1,2]的最大值之和，答案为 18。注意[1,1]和[2,2]虽然是原序列的子序列，但不是连续子序列，所以不应该被计算。另外，注意其中有一些值相同的子序列，但由于他们在原序列中的下标不同，属于不同的非空连续子序列，所以会被分别计算。解决该问题有许多算法，以下程序使用分治算法，时间复杂度 O(n log n).

尝试补全程序

```
#include <iostream>
#include <algorithm>
#include <vector>

const int MAXN = 100000;
int n;
int a[MAXN];
long long ans;

void solve(int l, int r) {
    if (l + 1 == r) {
        ans += a[l];
        return;
    }
    int mid = (l + r) >> 1;
    std::vector<int> pre(a + mid, a + r);
    for (int i = 0; i < r - mid; ++i) pre[i] = a[mid + i]; // Fill pre vector with elements from a[mid] to a[r-1]
    std::sort(pre.begin(), pre.end()); // Sort the pre vector
    std::vector<long long> sum(r - mid + 1);
    for (int i = 0; i < r - mid; ++i) sum[i + 1] = sum[i] + pre[i]; // Calculate prefix sums for pre vector
    for (int i = mid - 1, j = mid, max = 0; i >= l; --i) {
        while (j < r && a[j] < a[i]) ++j; // Find the first element in the right half that is not less than a[i]
        max = std::max(max, a[i]); // Update the maximum element seen so far
        ans += max - a[i]; // Add the difference between the maximum and the current element to the answer
        ans += sum[j - mid] - sum[i - mid]; // Add the sum of the first j - mid elements in the pre vector to the answer
    }
    solve(l, mid); // Recursively solve the left half
    solve(mid, r); // Recursively solve the right half
}

int main() {
    std::cin >> n;
    for (int i = 0; i < n; ++i) std::cin >> a[i];
    solve(0, n); // Start the recursive solution from 0 to n-1
    std::cout << ans << std::endl;
    return 0;
}

```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. pre[i] = std::max(pre[i - 1], a[i - 1])

B. pre[i + 1] = std::max(pre[i],pre[i + 1])

C. pre[i] = std::max(pre[i -1], a[i])

D. pre[i] = std::max(pre[i], pre[i - 1])

2.

A. a[j] < max

B. a[j] < a[i]

C. pre[j - mid] < max

D. pre[j - mid] > max

3.

A. (long long)(j - mid) * max

B. (long long)(j - mid) * (i - 1) * max

C. sum[j - mid]

D. sum[j - mid] * (i - 1)

4.

A. (long long)(r - j) * max

B. (long long)(r - j) * (mid - i) * max

C. sum[r - mid] - sum[j - mid]

D. (sum[r - mid] - sum[j - mid]) * (mid - i)

5.

A. solve(0，n)

B. solve(0，n - 1)

C. solve(1，n)

D. solve(1，n－1)

40.单选题 [CSP-J2019] 1.（矩阵变幻）有一个奇幻的矩阵，在不停的变幻，其变幻方式为：

数字 0 变成矩阵

0 0

0 1

数字 1 变成矩阵

1 1

1 0

最初该矩阵只有一个元素 0，变幻 n 次后，矩阵会变成什么样？

例如，矩阵最初为：[0]

[0]；

矩阵变幻 1 次后：

0 0

0 1

矩阵变幻 2 次后：

0 0 0 0

0 1 0 1

0 0 1 1

0 1 1 0

输入一行一个不超过 10 的正整数 n。输出变幻 n 次后的矩阵。试补全程序。

提示：

<< 表示二进制左移运算符，例如 (11)2<< 2=(1100)2

而 ^ 表示二进制异或运算符，它将两个参与运算的数中的每个对应的二进制位一进行比较，若两个二进制位相同，则运算结果的对应二进制位为 0，反之为 1。

```
#include <cstdio>
using namespace std;
int n;
const int max_size = 1 << 10;
```

```
5
6  int res[max_size][max_size];
7
8  void recursive(int x, int y, int n, int t) {
9      if (n == 0) {
10         res[x][y] = ①;
11         return;
12     }
13     int step = 1 << (n - 1);
14     recursive(②, n - 1, t);
15     recursive(x, y + step, n - 1, t);
16     recursive(x + step, y, n - 1, t);
17     recursive(③, n - 1, !t);
18 }
19
20 int main() {
21     scanf(""%d"", &n);
22     recursive(0, 0, ④);
23     int size = ⑤;
24     for (int i = 0; i < size; i++) {
25         for (int j = 0; j < size; j++)
26             printf(""%d"", res[i][j]);
27         puts("""");
28     }
29     return 0;
30 }
```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. n%2

B. 0

C. t

D. 1

2.

A. x-step,y-step

B. x,y-step

C. x-step,y

D. x,y

3.

A. x-step,y-step

B. x+step,y+step

C. x-step,y

D. x,y-step

4.

A. n-1,n%2

B. n,0

C. n,n%2

D. n-1,0

5.

A. 1<<(n+1)

B. 1<<n

C. n+1

D. 1<<(n-1)

答案

答案

1.T

2.C

3.A

4.C

5.ABC

6.B

7.B

8.T

9.T

10.F

11.F

12.T

13.T

14.7

15.D

16.C

17.4 次，第一步分成 3 组：27，27，26，将前 2 组放到天平上。

18.A

19.B

20.C

21.B

22.B

23.B

24.D

25.D

26.B

27.B

28.

1. 正确答案： lbound < rbound
2. 正确答案： count = 0
3. 正确答案： x[i] > mid
4. 正确答案： count++
5. 正确答案： rbound = mid

29.

1. 正确答案： n - nn + 1

2. 正确答案： M[i] < C[j]

3. 正确答案： count <= A

4. 正确答案： check(mid)

5. 正确答案： mid - 1

30.

1. 正确答案： count = count + len[i]

2. 正确答案： count < m

3. 正确答案： lbound < ubound

4. 正确答案： (lbound + ubound + 1) / 2

5. 正确答案： count = count +len[i] / mid

31.

（1）、num + len / t

（2）、num >= k

（3）、left = 0

（4）、left + 1

（5）、!isok(mid) （或者 isok(mid) == 0）

32.AAACBB

33.BACAD

34.AABBCBA

35.7

36.14

37.

正确答案： return

正确答案： (dr<tr+s)&&(dc<tc+s) / dr<tr+s&&dc<tc+s

正确答案： chessboard(tr,tc,tr+s-1,tc+s-1,s)

正确答案： chessboard(tr,tc+s,tr+s-1,tc+s,s)

正确答案： chessboard(tr+s,tc,tr+s,tc+s-1,s)

正确答案： chessboard(tr+s,tc+s,tr+s,tc+s,s)

38.

(1) 0,-1

（2） a[x-1][y-1]

（3） a[x-1][y-1]

（4） d[2*i+1]

（5） a[x1][y1]==orig (或者 orig==a[x1][y1])

（6） orig=a[i][j]"

39.DBACA

40.CDBBB

GW 信奥 CSP 初赛习题集 6-4

算法——贪心算法

题型

1.单选题 [CSP-J2021] 有四个人要从 A 点坐一条船过河到 B 点，船一开始在 A 点。该船一次最多可坐两个人。 已知这四个人中每个人独自坐船的过河时间分别为 1,2,4,8，且两个人坐船的过河时间为两人独自过河时间的较大者。则最短（ ）时间可以让四个人都过河到 B 点（包括从 B 点把船开回 A 点的时间）。

A. 14

B. 15

C. 16

D. 17

2.单选题 [CSP-J2019] 新学期开学了，小胖想减肥，健身教练给小胖制定了两个训练方案。

方案一：每次连续跑 3 公里可以消耗 300 千卡（耗时半小时）；

方案二：每次连续跑 5 公里可以消耗 600 千卡（耗时 1 小时）。

小胖每周周一到周四能抽出半小时跑步，周五到周日能抽出一小时跑步。

另外，教练建议小胖每周最多跑 21 公里，否则会损伤膝盖。

请问如果小胖想严格执行教练的训练方案，并且不想损伤膝盖，每周最多通过跑步消耗多少千卡？（）

A. 3000

B. 2500

C. 2400

D. 2520

答案

1-2 B C

GW 信奥 CSP 初赛习题集 6-5

算法——动态规划算法

1. 动态规划基本概念

1.判断题 [GESP202406【7 级】] 动态规划有递推实现和递归实现，对于很多问题，通过记录子问题的解，两种实现的时间复杂度是相同的。

2.单选题 [GESP202406【6 级】] 在求解最优化问题时，动态规划常常涉及到两个重要性质，即最优子结构和( )。

A.重叠子问题

B. 分治法

C. 贪心策略

D. 回溯算法

2. 动态规划的应用

1. 走楼梯模型

3.单选题 [GESP202309【6 级】] 青蛙每次能跳 1 或 2 步。下面是青蛙跳到第 N 步台阶 C++实现代码。该段代码采用的算法是（ ）。

```
void jumpFrog(int N) {
    if (N <= 3)
        return N;
    return jumpFrog(N - 1) + jumpFrog(N - 2);
}
```

A. 递推算法

B. 贪心算法

C. 动态规划算法

D. 分治算法

4.单选题 [GESP202406【6 级】] 青蛙每次能跳 1 或 2 步，下面代码计算青蛙跳到第 n 步台阶有多少种不同跳法。则下列说法，错误的是( )。

```
int jump_recur(int n) {
    if (n == 1) return 1;
    if (n == 2) return 2;
    return jump_recur(n - 1) + jump_recur(n - 2);
}
int jump_dp(int n) {
    vector<int> dp(n + 1); // 创建一个动态规划数组，用于保存已计算的值
    // 初始化前两个数
    dp[1] = 1;
    dp[2] = 2;
    // 从第三个数开始计算斐波那契数列
    for (int i = 3; i <= n; ++i) {
        dp[i] = dp[i - 1] + dp[i - 2];
    }
    return dp[n];
}
```

A. 函数 jump_recur()采用递归方式。

B. 函数 jump_dp()采用动态规划方法。

C. 当 n 较大时，函数 jump_recur()存在大量重复计算，执行效率低。

D. 函数 jump_recur()代码量小，执行效率高。

2. 棋盘模型

5.单选题 [NOIP 提高组 2017/CSP-S2019] 在正实数构成的数字三角形排列形式如图所示，第一行的数为 a11；第二行的数从左到右依次为 a21,a22;…第 n 行的数为 an1,an2,…，ann。从 a11 开始，每一行的数 aij 只有两条边可以分别通向下一行的两个数 a(i+1)j 和 a(i+1)(j+1)。用动态规划算法找出一条从 a11 向下通到 an1,an2,…,ann 中某个数的路径，使得该路径上的数之和达到最大。令 C[i,j]是从 a11 到 aij 的路径上的数的最大和，并且 C[i,0]=C[0,j]=0，则 C[i,j]=( )。

a11

a21 a22

a31 a32 a33

……

an1 an2 …… ann

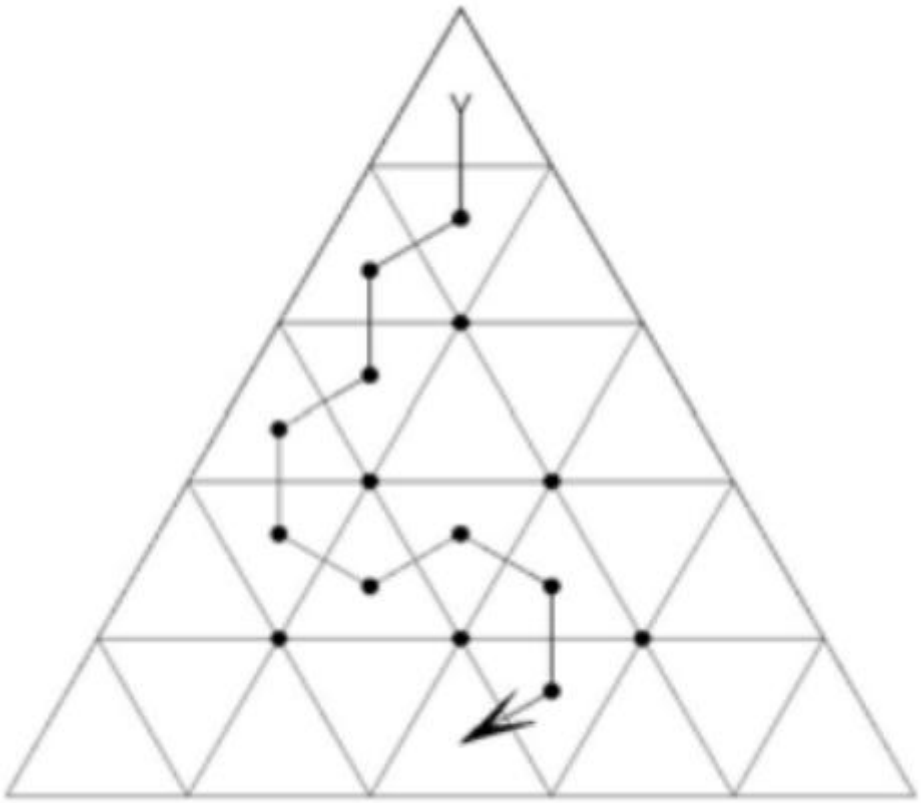
A. max{C[i-1,j-1],C[i-1,j]}+aij

B. C[i-1,j-1]+c[i-1,j]

C. max{C[i-1,j-1],C[i-1,j]}+1

D. max{C[i,j-1],C[i-1,j]}+aij

6.d [NOIP 提高组 2006] 将边长为 n 的正三角形每边 n 等分，过每个分点分别做另外两边的平行线，得到若干个正三角形，我们称为小三角形。正三角形的一条通路是一条连续的折线，起点是最上面的一个小三角形，终点是最下面一行位于中间的小三角形。在通路中，只允许由一个小三角形走到另一个与其有公共边的且位于同一行或下一行的小三角形，并且每个小三角形不能经过两次或两次以上（图中是 n=5 时一条通路的例子）。设 n=10，则该正三角形的不同的通路的总数_____。



7.单选题 [GESP202403【8 级】] 下面程序的输出为（ ）。

```
#include <iostream>
using namespace std;
int a[10][10];
int main() {
    int m = 5, n = 4;
    for (int x = 0; x <= m; x++)
        a[x][0] = 1;
    for (int y = 1; y <= n; y++)
        a[0][y] = 1;
    for (int x = 1; x <= m; x++)
        for (int y = 1; y <= n; y++)
            a[x][y] = a[x - 1][y] + a[x][y - 1];
    cout << a[m][n] << endl;
    return 0;
}
```

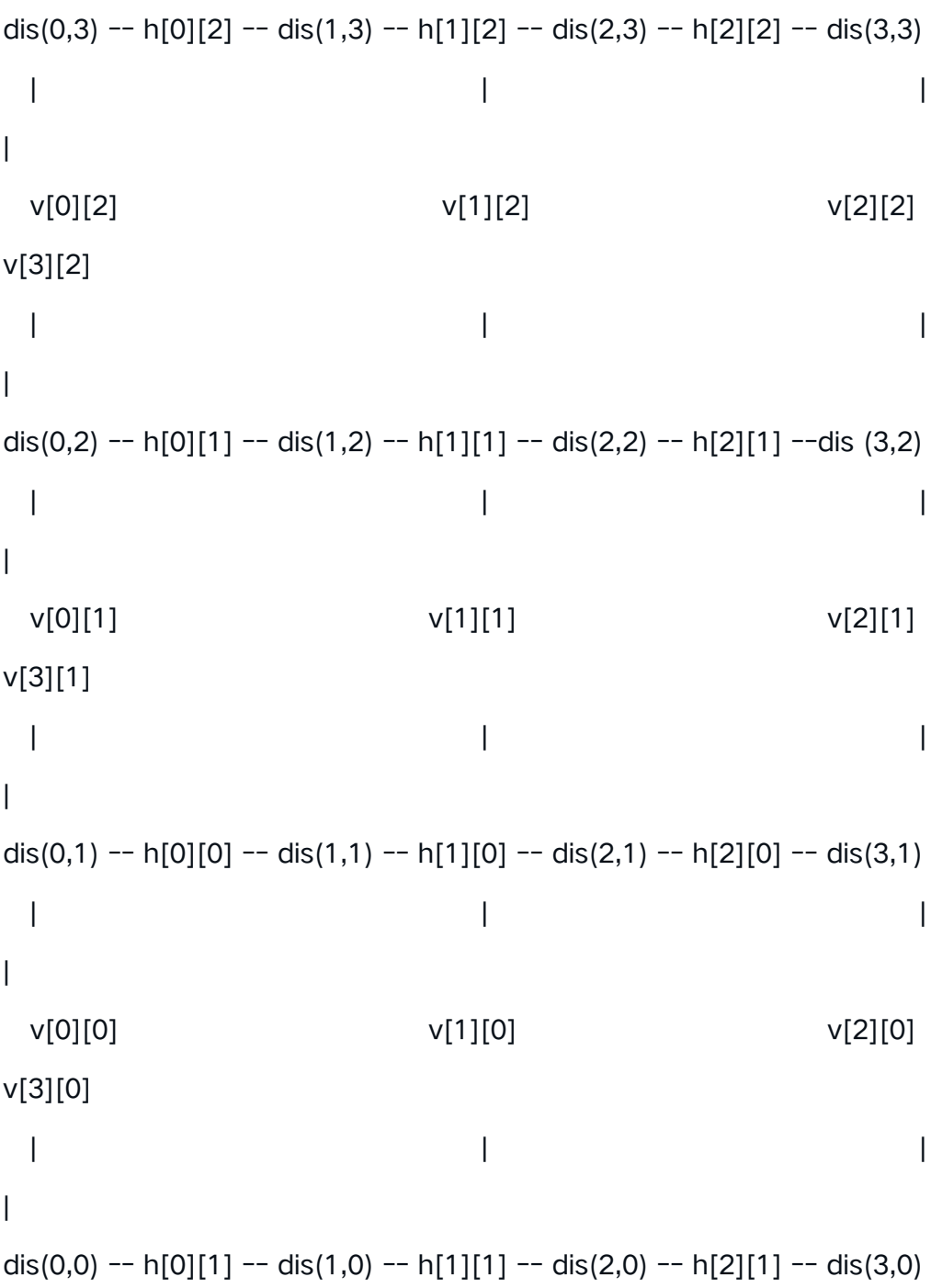
A. 4

B. 5

C. 126

D. 3024

8.单选题 [GESP202312【8 级】] 下面的程序中，二维数组 h 和 v 分别代表如下图所示的网格中的水平边的时间消耗和垂直边的时间消耗。程序使用动态规划计算从左下角到右上角的最小时间消耗，则横线处应该填写下列哪个选项的代码？（ ）。




```
int dis[MAXY][MAXX];
int shortest_path(int x, int y) {
```

```
    dis[0][0] = 0;
    for (int i = 0; i < y; i++)
        dis[i + 1][0] = dis[i][0] + v[i][0];
    for (int j = 0; j < x; j++)
        dis[0][j + 1] = dis[0][j] + h[0][j];
    for (int i = 0; i < y; i++)
        for (int j = 0; j < x; j++)
            // 在此处填写代码
    return dis[y][x];
}
```

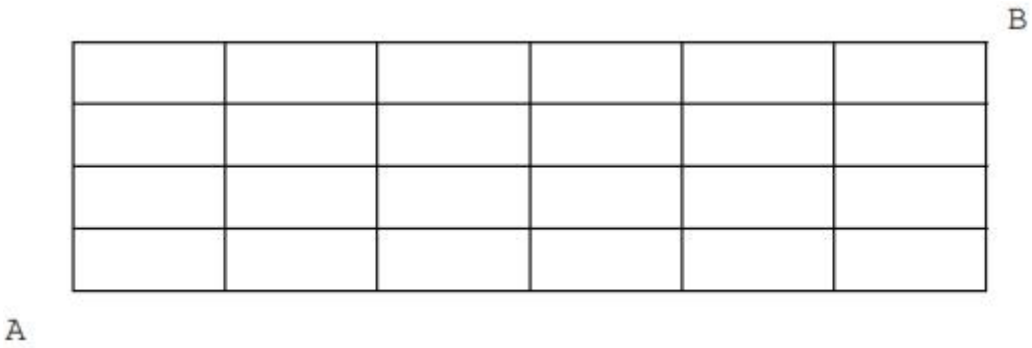
A. dis[i][j] = min(dis[i - 1][j] + v[i - 1][j], dis[i][j - 1] + h[i][j - 1]);

B. dis[i][j] = min(dis[i - 1][j] + h[i - 1][j], dis[i][j - 1] + v[i][j - 1]);

C. dis[i + 1][j + 1] = min(dis[i][j + 1] + v[i][j + 1], dis[i + 1][j] + h[i + 1][j]);

D. dis[i + 1][j + 1] = min(dis[i][j + 1] + h[i][j + 1], dis[i + 1][j] + v[i + 1][j]);

9.填空题 [NOIP 普及组 2007] （最短路线）某城市 的街道是一个很规整的矩形网格（见下图），有 7 条南北向的纵街，5 条东西向的横街。现要从西南角的 A 走到东北角的 B，最短的走法共有多少种？



3. 序列模型

10.判断题 [GESP202312【7 级】] 给定一个数字序列 A1，A2，A3，...，An ，要求 i 和 j （ 1<=i<=j<=n ），使 Ai+…+Aj 最大，可以使用动态规划方法来求解。

11.单选题 [GESP202406【7 级】] 关于序列 {2,7,1,5,6,4,3,8,9} ，以下说法错误的是（ ）。

A. {2,5,6,8,9} 是它的最长上升子序列

B. {1,5,6,8,9} 是它的最长上升子序列

C. {7,5,4,3} 是它的最长下降子序列

D. {1,5,6,8,9} 是它的唯一最长上升子序列

12.单选题 [GESP202312【7 级】] 下面代码可以用来求最长上升子序列（LIS）的长度，如果输入是： 5 1 7 3 5 9 ，则输出是( )。

```
int a[2023], f[2023];

int main() {
    int n, i, j, ans = -1;
    cin >> n;
    for (i = 1; i <= n; i++) {
        cin >> a[i];
        f[i] = 1;
    }

    for (i = 1; i <= n; i++)
        for (j = 1; j < i; j++)
            if (a[j] < a[i])
                f[i] = max(f[i], f[j] + 1);

    for (i = 1; i <= n; i++) {
        ans = max(ans, f[i]);
        cout << f[i] << "" "";
    }

    cout << ans << endl;
    return 0;
}
```

A. 9 7 5 1 1 9

B. 1 2 2 3 4 4

C. 1 3 5 7 9 9

D. 1 1 1 1 1 1

13.单选题 [NOIP 普及组 2013/NOIP 提高组 2013]

```
#include <iostream>
using namespace std;
```

```
int main(){
    const int SIZE = 100;
    int height[SIZE], num[SIZE], n, ans;
    cin>>n;
    for (int i = 0; i < n; i++) {
        cin>>height[i];
        num[i] = 1;
        for (int j = 0; j < i; j++) {
            if ((height[j] < height[i]) && (num[j] >= num[i]))
                num[i] = num[j]+1;
        }
    }
    ans = 0;
    for (int i = 0; i < n; i++) {
        if (num[i] > ans) ans = num[i];
    }
    cout<<ans<<endl;
}
```

输入：

6

2 5 3 11 12 4

输出：

14.单选题 [CSP-S2019]

```
#include <cstdio>
using namespace std;
int n;
int a[100];

int main() {
    scanf(""%d"", &n);
    for (int i = 1; i <= n; ++i)
        scanf(""%d"", &a[i]);
    int ans = 1;
    for (int i = 1; i <= n; ++i) {
        if (i > 1 && a[i] < a[i - 1])
            ans = i;
        while (ans < n && a[i] >= a[ans + 1])
            ++ans;
        printf(""%d\n"", ans);
    }
```

```
18        return 0;
19    }
```

判断题

第 16 行输出 ans 时，ans 的值一定大于 i。（ ）

程序输出的 ans 小于等于 n。（ ）

若将第 12 行的 < 改为 !=，程序输出的结果不会改变。（ ）

当程序执行到第 16 行时，若 ans−i>2，则 a[i+1]⩽a[i]。（ ）

选择题

若输入的 a 数组是一个严格单调递增的数列，此程序的时间复杂度（ ）

最坏情况下，此程序的时间复杂度是（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. O(logn)

B. $O(n^2)$

C. O(nlogn)

D. O(n)

6.

A. $O(n^2)$

B. O(logn)

C. O(n)

D. O(nlogn)

15.填空题 [NOIP 普及组 2012]

```
#include <iostream>
#include <string>
using namespace std;
int n, ans, i, j;
string s;
char get(int i){
    if (i < n)
        return s[i];
    else
        return s[i-n];
}
int main(){
    cin >> s;
    n = s.size();
    ans = 0;
    for (i = 1; i <= n-1; i++){
        for (j = 0; j <= n-1; j++)
            if (get(i+j) < get(ans+j)){
                ans = i; break;
            }else if (get(i+j) > get(ans+j))
                break;
    }
    for (j = 0; j <= n-1; j++)
        cout<<get(ans+j);
    cout<<endl;
}
```

输入： CBBADADA

输出： ______

16.单选题 [GESP202406【7 级】] 已知两个序列 s1= {1,3,4,5,6,7,7,8,1} 、s2= {3,5,7,4,8,2,9,5,1} ，则它们的最长公共子序列是（ ）。

A. {3,5,7,8,1}

B. {3,4,5,7,8}

C. {5,7,8}

D. {3,5,7,9,1}

17.单选题 [NOIP 提高组 2005] 字符串“ababacbab”和字符串“abcba”的最长公共子串是（ ）。

A. abcba

B. cba

C. abc

D. ab

E. bcba

18.单选题 [CSP-S2023] 最长公共子序列长度常常用来衡量两个序列的相似度。其定义如下：给定两个序列 X=x1,x2,x3,⋯,xm 和 Y=y1,y2,y3 ,⋯,yn，最长公共子序列（LCS）问题的目标是找到一个最长的新序列 Z=z1,z2,z3,⋯,zk,使得序列 Z 既是序列 X 的子序列，又是序列 Y 的子序列，且序列 Z 的长度 k 在满足上述条件的序列里是最大的。（注：序列 A 是序列 B 的子序列，当且仅当在保持序列 B 元素顺序的情况下，从序列 B 中删除若干个元素，可以使得剩余的元素构成序列 A。）则序列 ABCAAAABA 和 ABABCBABA 的最长公共子序列长度为（ ）

A.4

B.5

C.6

D.7

19.单选题 [GESP202312【7 级】] 下面代码段可以求两个字符串 s1 和 s2 的最长公共子串（LCS），下列相关描述不正确的是（ ）。

```
while (cin >> s1 >> s2) {
    memset(dp, 0, sizeof(dp));
    int n1 = strlen(s1), n2 = strlen(s2);
    for (int i = 1; i <= n1; ++i) {
        for (int j = 1; j <= n2; ++j) {
```

```
                if (s1[i - 1] == s2[j - 1])
                    dp[i][j] = dp[i - 1][j - 1] + 1;
                else
                    dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
            }
        }
        cout << dp[n1][n2] << endl;
}
```

A. 代码的时间复杂度为 $O(N^2)$

B. 代码的空间复杂度为 $O(N^2)$

C. 空间复杂度已经最优

D. 采用了动态规划求解

20.单选题 [GESP202403【6 级】] 以下动态规划算法的含义与目的是（ ）。

```
int function(vector<int>& nums) {
    int n = nums.size();
    if (n == 0)
        return 0;
    if (n == 1)
        return nums[0];
    vector<int> dp(n, 0);
    dp[0] = nums[0];
    dp[1] = max(nums[0], nums[1]);
    for (int i = 2; i < n; ++i) {
        dp[i] = max(dp[i - 1], nums[i] + dp[i - 2]);
    }
    return dp[n - 1];
}
```

A. 计算数组 nums 中的所有元素的和

B. 计算数组 nums 中相邻元素的最大和

C. 计算数组 nums 中不相邻元素的最大和

D. 计算数组 nums 中的最小元素

21.单选题 [CSP-J2023]

```
#include<iostream>
#include<vector>
```

```
#include<algorithm>
using namespace std;
int f(string x,string y){
    int m=x.size();
    int n=y.size();
    vector<vector<int>>v(m+1,vector<int>(n+1,0));
   for(int i=1;i<=m;i++){
        for(int j=1;j<=n;j++){
            if(x[i-1]==y[j-1]){
                v[i][j]=v[i-1][j-1]+1;
            }else{
                v[i][j]=max(v[i-1][j],v[i][j-1]);
            }
        }
    }
    return v[m][n];
}
bool g(string x,string y){
    if(x.size() != y.size()){
        return false;
    }
    return f(x+x,y)==y.size();
}
int main(){
    string x,y;
    cin>>x>>y;
    cout<<g(x,y)<<endl;
    return 0;
}
```

判断题

f 函数的返回值小于等于 min{n,m}。( )

f 函数的返回值等于两个输入字符串的最长公共子串的长度。( )

当输入两个完全相同的字符串时，g 函数的返回值总是 true。( )

单选题

将第 19 行中的 v[m][n] 替换为 v[n][m]，那么该程序（ ）。

当输入为 csp-j p-jcs 时，输出为（ ）。

当输入为 csppsc spsccp 时，输出为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 行为不变

B. 只会改变输出

C. 一定非正常退出

D. 可能非正常退出

5.

A. 0

B. 1

C. T

D. F

6.

A. T

B. F

C. 0

D. 1

22.单选题 [CSP-S2019] t 是 s 的子序列的意思是：从 s 中删去若干个字符，可以得到 t。特别的，如果 s == t，那么 t 也是 s 的子序列；空串是任何串的子序列。例如“acd”是“abcde”的子序列，“acd”是“acd”的子序列，但“acd”不是“abcde”的子序列。

S[x..y]表示 s[x]…s[y]共 y-x+1 个字符构成的字符串，若 x>y 则 s[x..y]是空串。t[x..y]同理。

```cpp
#include <iostream>
#include <string>
using namespace std;
const int max1 = 202;
string s, t;
int pre[max1], suf[max1];

int main() {
    cin >> s >> t;
    int slen = s.length(), tlen = t.length();

    for (int i = 0, j = 0; i < slen; ++i) {
        if (j < tlen && s[i] == t[j]) ++j;
        pre[i] = j; // t[0..j-1] 是 s[0..i] 的子序列
    }

    for (int  i = slen - 1 , j = tlen - 1; i >= 0; --i) {
        if(j >= 0 && s[i] == t [j]) --j;
        suf[i]= j; // t[j+1..tlen-1] 是 s[i..slen-1] 的子序列
    }

    suf[slen] = tlen -1;
    int ans = 0;
    for (int i = 0, j = 0, tmp = 0; i <= slen; ++i){
        while(j <= slen && tmp >= suf[j] + 1) ++j;
        ans = max(ans, j - i - 1);
        tmp = pre[i];
    }
    cout << ans << endl;
    return 0;
}
```

提示 t[0...pre[]-1]是 s[0...i 的子序列;t[suf[i]+1...tlen-1]是 s[i...slen-1]的子序列。

判断题

程序输出时，suf 数组满足:对任意 0<i< slen,suf[i]<suf[i+1]。()

当 t 是 s 的子序列时，输出一定不为 0。()

程序运行到第 23 行时，j-i-1 一定不小于 0。()

当 t 是 s 的子序列时，pre 数组和 suf 数组满足:对任意 0≤i<slen,pre[i]>suf[i+1]+1. ()

选择题

5.若 tlen=10，输出为 0，则 slen 最小为()

6.若 tlen=10，输出为 2，则 slen 最小为()

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 10

B. 12

C. 0

D. 1

6.

A. 0

B. 10

C. 12

D. 1

23.填空题 [NOIP 普及组 2011] 定义字符串的基本操作为：删除一个字符、插入一个字符和将一个字符修改成另一个字符这三种操作。将字符串 A 变成字符串 B 的最少操作步数，称为字符串 A 到字符串 B 的编辑距离。字符串 ""ABCDEFG"" 到字符串 ""BADECG"" 的编辑距离为____。

24.填空题 [NOIP 提高组 2011] 定义一种字符串操作，一次可以将其中一个元素移到任意位置。举例说明，对于字符串 BCA 可以将 A 移到 B 之前，变字符串 ABC。如果要将字符串 DACHEBGIF 变成 ABCDEFGHI 最少需要_____次操作。

25.单选题 [CSP-J2023] （编辑距离）给定两个字符串，每次操作可以选择删除（Delete）、插入（Insert）、替换（Replace），一个字符，求将第一个字符串转换为第二个字符串所需要的最少操作次数。

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;
int min(int x, int y, int z) {
    return min(min(x, y), z);
}
int edit_dist_dp(string str1, string str2) {
    int m = str1.length();
    int n = str2.length();
    vector<vector<int>> dp(m + 1, vector<int>(n + 1));
    for (int i = 0; i <= m; i++) {
        for (int j = 0; j <= n; j++) {
            if (i == 0)
                dp[i][j] = ①;
            else if (j == 0)
                dp[i][j] = ②;
            else if (③)
                dp[i][j] = ④;
            else
                dp[i][j] = 1 + min(dp[i][j - 1], dp[i - 1][j], ⑤);
        }
    }
    return dp[m][n];
}
int main() {
    string str1, str2;
    cin >> str1 >> str2;
    cout << ""Mininum number of operation:"" << edit_dist_dp(str1, str2) << endl;
    return 0;
}
```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. j

B. i

C. m

D. n

2.

A. j

B. i

C. m

D. n

3.

A. str1[i-1]==str2[j-1]

B. str1[i]==str2[j]

C. str1[i-1]!=str2[j-1]

D. str1[i]!=str2[j]

4.

A. dp[i-1][j-1]+1

B. dp[i-1][j-1]

C. dp[i-1][j]

D. dp[i][j-1]

5.

A. dp[i][j] + 1

B. dp[i-1][j-1]+1

C. dp[i-1][j-1]

D. dp[i][j]

26.填空题 [NOIP 普及组 2009/NOIP 提高组 2009] （最大连续子段和）给出一个数列（元素个数不多于 100），数列元素均为负整数、 正整数、 0。请找出数列中的一个连续子数列， 使得这个子数列中包含的所有元素之和最大， 在和最大的前提下还要求该子数列包含的元素个数最多， 并输出这个最大和以及该连续子数 列中元素的个数。例如数列为 4，-5，3，2， 4 时，输出 9 和 3；数列为 1 2 3 -5 0 7 8 时，输 出 16 和 7。

```
#include <stdio.h>
int a[101];
int n,i,ans,len,tmp,beg;
int main(){
    scanf(""%d"",&n);
    for (i=1;i<=n;i++)
    scanf(""%d"",&a[i]);
    tmp=0;
    ans=0;
    len=0;
    beg= ① ;
    for (i=1;i<=n;i++)
    {
    if (tmp+a[i]>ans)
    {
        ans=tmp+a[i];
        len=i-beg;
    }
    else if ( ②&&i-beg>len)
        len=i-beg;
    if (tmp+a[i] ③ )
    {
        beg= ④ ;
        tmp=0;
    }
    else
        ⑤ ;
    }
printf(""%d %d\n"",ans,len);
return 0;
}
```

27.填空题 [NOIP 普及组 2014/NOIP 提高组 2014]（最大子矩阵和）给出 m 行 n 列的整数矩阵,求最大的子矩阵和(子矩阵不能为空)。输入第一行包含两个整数 m 和 n，

即矩阵的行数和列数。之后 m 行,每行 n 个整数，描述整个矩阵。程序最终输出最大的子矩阵和。

```
#include <iostream>
using namespace std;
const int SIZE = 100;
int matrix[SIZE + 1][SIZE + 1];
int rowsum[SIZE + 1][SIZE + 1]; //rowsum[i][j]记录第 i 行前 j 个数的和
int m, n, i, j, first, last, area, ans;
int main()
{
    cin >> m >> n;
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            cin >> matrix[i][j];
    ans = matrix ①;
    for(i = 1; i <= m; i++)
        ②
    for(i = 1; i <= m; i++)
        for(j = 1; j <= n; j++)
            rowsum[i][j] = ③;
    for(first = 1; first <= n; first++)
        for(last = first; last <= n; last++)
        {
            ④;
            for(i = 1; i <= m; i++)
            {
                area += ⑤;
                if(area > ans)
                    ans = area;
                if(area < 0)
                    area = 0;
            }
        }
    cout << ans << endl;
    return 0;
```

28.填空题 [NOIP 提高组 2015] (双子序列最大和)给定一个长度为 n(3≤n≤1000) 的整数序列，要求从中选出两个连续子序列，使得这两个连续子序列的序列和之和最大，最终只需输出这个最大和。一个连续子序列的序列和为该连续子序列中所有数之和。要求：每个连续子序列长度至少为 1，且两个连续子序列之间至少间隔 1 个数。

```
#include <iostrea m>
using namespace std;
const int MAXN = 1000;
int n, i, ans, sum;
int x[MAXN];
int lmax[MAXN];
// lmax[i] 为仅含 x[i] 及 x[i] 左侧整数的连续子序列的序列和中，最大的序列和
int rmax[MAXN];
// rmax[i] 为仅含 x[i] 及 x[i] 右侧整数的连续子序列的序列和中，最大的序列和

int main() {
    cin >> n;
    for (i = 0; i < n; i++) cin >> x[i];
    lmax[0] = x[0] ;
    for (i = 1; i < n; i++)
        if (lmax[i - 1] <= 0)
            lmax[i] = x[i];
        else
            lmax[i] = lmax[i - 1] + x[i];
    for (i = 1; i < n; i++)
        if (lmax[i] < lmax[i - 1])
            lmax[i] = lmax[i - 1];
    ①;
    for (i = n - 2; i >= 0; i --)
        if (rmax[i + 1] <= 0)
            ②;
        else
            ③;
    for (i = n - 2; i >= 0; i --)
        if (rmax[i] < rmax[i + 1])
            ④;
    ans = x[ 0] + x [2];
    for (i = 1; i < n - 1; i++) {
        sum = ⑤;
        if (sum > ans)
            ans = sum;
    }
    cout << ans << endl;
    return 0;
}
```

29.单选题 [CSP-S2020] （最优子序列）取 m = 16，给出长度为 n 的整数序列 a1,a2,…,an(0≤ai≤2^m)。对于一个二进制数 x，定义其分值 w(x) 为 x+popcnt(x)，

其中 popcnt(x) 表示 x 二进制表示中 1 的个数。对于一个子序列 b1,b2,⋯,bk，定义其子序列分值 S 为 w(b1⊕b2)+w(b2⊕b3)+w(b3⊕b4)+⋯w(bk−1⊕bk)。其中⊕表示按位异或。对于空子序列，规定其子序列分值为 0。求一个子序列似的其子序列的分值最大，输出这个最大值。输入第一行包含一个整数 n(1≤n≤40000)。接下来一行包含 n 个整数 a1,a2,⋯,an。

提示：考虑优化朴素的动态规划算法，将前 m/2 位和后 m /2 位分开计算。

Max[x][y] 表示当前的子序列下一个位置的高 8 位是 x、最后一个位置的低 8 位是 y 时的最大价值。试补全程序。

```
#include <iostream>

using namespace std;

typedef long long LL;

const int MAXN = 40000, M = 16, B = M >> 1, MS = (1 << B) - 1;
const LL INF = 1000000000000000LL;
LL Max[MS + 4][MS + 4];

int w(int x)
{
    int s = x;
    while(x)
    {
        ①;
        s++;
    }
    return s;
}

void to_max(LL &x, LL y)
{
    if(x < y)
        x = y;
}

int main()
{
    int n;
    LL ans = 0;
    cin >> n;
    for(int x = 0; x <= MS; x++)
```

```
            for(int y = 0; y <= MS; y++)
                Max[x][y] = -INF;
        for(int i = 1; i <= n ; i++)
        {
            LL a;
            cin >> a;
            int x = ② , y = a & MS;
            LL v = ③;
            for(int z = 0; z < = MS; z++)
                to_max(v, ④);
            for(int z = 0; z < = MS; z++)
                ⑤;
            to_max(ans , v);
        }
        cout << ans << endl;
        return 0;
}
```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. x >>= 1

B. x ^= x &(x ^ (x + 1))

C. x -= x | -x

D. x ^= x &(x ^ (x - 1))

2.

A. (a & MS) << B

B. a >> B

C. a & (1 << B)

D. a & (MS << B)

3.

A. -INF

B. Max[y][x]

C. 0

D. Max[x][y]

4.

A. Max[x][z] + w(y ^ z)

B. Max[x][z] + w(a ^ z)

C. Max[x][z] + w(x ^ (z << B))

D. Max[x][z] + w(x ^ z)

5.

A. to_max(Max[y][z], v + w(a ^ (z << B)))

B. to_max(Max[z][y], v + w((x ^ z) << B))

C. to_max(Max[z][y], v + w(a ^ (z << B)))

D. to_max(Max[x][z], v + w(y ^ z))

4. 其他

30.填空题 [NOIP 提高组 2005] N 叉树

题目描述:

我们都了解二叉树的先根遍历，中根遍历和后根遍历。当知道先根遍历的结果和中根遍历结果的时候，我们可以唯一的确定二叉树；同样的，如果知道了后根遍历的结果和中根遍历结果，二叉树也是唯一确定的。但是如果只知道先根遍历和后根遍历的结果，二叉树就不是唯一的了。但是我们可以计算满足条件的不同二叉树的一共有多少个。这不是一个很困难的问题，稍微复杂一点，我们把这个问题推广到 N 叉树。

我们用小写英文字母来表示 N 叉树的结点，不同的结点用不同的字母表示。比如，对于 4 叉树，如果先根遍历的结果是 abdefgc，后根遍历的结果是 defgbca，那么我们可以得到 6 个不同的 4 叉树（如下图）。

输入:

输入数据包括 3 行。

第一行是一个正整数 N(1 ⩽ N ⩽ 20)，表示我们要考虑 N 叉树。

第二行和第三行分别是两个字符串序列，分别表示先根遍历和后根遍历的结果。

输出:

输出不同的 N 叉树的数目。题目中给的数据保证得到的结果小于 231 。

输入样例:

4

abdefgc

defgbca

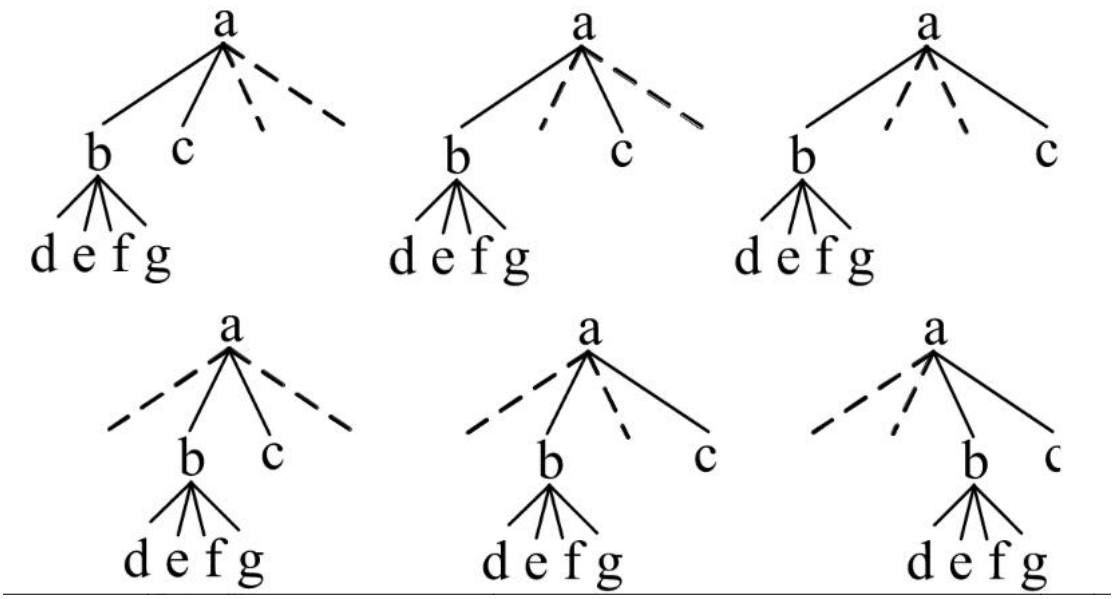
输出样例:

6

程序:

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;
char str1[100], str2[100];
int N;
long com[100][100];
long getcom(int x, int y) {
 if (y == 0 || x == y)     ①   ;
      else if (com[x][y] != 0) return com[x][y];
      else {
           com[x][y] = getcom(x - 1, y)+     ②   ;
           return com[x][y];
      }
}
long count(int a, int b, int c){
      long sum = 1;
      int k = 0;
      int s = a + 1, t = c, p;
      if (a == b) return 1;
      while(s <= b){
       p = t;
       while(str1[s] != str2[t]) t++;
           sum = sum * count(s, s + t - p, p);
           s =     ③   ;
              ④   ;
           k++;
      }
```

```
29        return    ⑤    * getcom(N, k);   }
30  int main(){
31        int len;
32        scanf(""%d"", &N);
33        scanf(""%s%s"", str1, str2);
34        len = strlen(str1);
35        printf(""%ld\n"", count(   ⑥   ));
36        return 0;
37  }
```



31.单选题 [CSP-J2022]

```
1   #include<algorithm>
2   #include<iostream>
3   #include<limits>
4   using namespace std;
5   const int MAXN = 105;
6   const int MAXK = 105;
7   int h[MAXN][MAXK];
8   int f(int n, int m)
9   {
10        if (m == 1) return n;
11        if (n == 0) return 0;
12        int ret = numeric_limits::max();
13        for (int i = 1; i <= n; i++)
14              ret = min(ret, max(f(n - i, m), f(i - 1, m - 1)) + 1);
15        return ret;
16  }
```

```
17 int g(int n, int m)
18 {
19     for (int i = 1; i <= n; i++)
20         h[i][1] = i;
21     for (int j = 1; j <= m; j++)
22         h[0][j] = 0;
23     for (int i = 1; i <= n; i++) {
24         for (int j = 2; j <= m; j++) {
25             h[i][j] = numeric_limits::max();
26             for (int k = 1; k <= i; k++)
27             h[i][j] = min(h[i][j],max(h[i - k][j], h[k - 1][j - 1]) + 1);
28         }
29     }
30     return h[n][m];
31 }
32 int main()
33 {
34     int n, m;
35     cin >> n >> m;
36     cout << f(n, m) << endl << g(n, m) << endl;
37     return 0;
38 }
```

假设输入的 n、m 均是不超过 100 的正整数，完成下面的判断题和单选题：

判断题

当输入为 7 3 时，第 19 行用来取最小值的 min 函数执行了 449 次。

输出的两行整数总是相同的。

当 m 为 1 时，输出的第一行总为 n。

单选题

算法 g(n,m)最为准确的时间复杂度分析结果为（ ）。

当输入为 20 2 时，输出的第一行为（ ）。

当输入 100 100 时，输出的第一行为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. $O(n^{3/2}m)$

B. O(nm)

C. $O(n^2m)$

D. $O(n*m^2)$

5.

A. 4

B. 5

C. 6

D. 20

6.

A. 6

B. 7

C. 8

D. 9

32.单选题 [CSP-S2021] （魔法数字） 小 H 的魔法数字是 4。给定 n， 他希望用若干个 4 进行若干次加法、减法和整除运算得到 n。但由于小 H 计算能力有限，计算过程中只能出现不超过 M=10000
的正整数。求至少可能用到多少个 4。例如，当 n=2 时，有 2=(4+4)/2，用到了 3 个 4，是最优方案。试补全程序。

```
#include <iostream>
#include <cstdlib>
#include <climits>
using namespace std;

```

```
const int M = 10000;
bool Vis[M + 1];
int F[M + 1];

void update(int &x, int y) {
    if (y < x)
        x = y;
}

int main() {
    int n;
    cin >> n;
    for (int i = 0; i <= M; i++)
        F[i] = INT_MAX;
    ①;
    int r = 0;
    while (②) {
        r++;
        int x = 0;
        for (int i = 1; i <= M; i++)
            if (③) {
                x = i;
                Vis[x] = 1;
                for (int i = 1; i <= M; i++)
                    if (④) {
                        int t = F[i] + F[x];
                        if (i + x <= M)
                            update(F[i + x], t);
                        if (i > x)
                            update(F[abs(i - x)], t);
                        if (i % x == 0)
                            update(F[i / x], t);
                        if (x % i == 0)
                            update(F[x / i], t);
                    }
            }
    }
    cout << F[n] << endl;
    return 0;
}
```

①处应填（ ）

A. F[4] = 0

B. F[1] = 4

C. F[1] = 2

D. F[4] = 1

②处应填（ ）

A. !Vis[n]

B. r < n

C. F[M] == INT_MAX

D. F[n] == INT_MAX

③处应填（ ）

A. F[i] == r

B. !Vis[i] && F[i] == r

C. F[i] < F[x]

D. !Vis[i] && F[i] < F[x]

④处应填（ ）

A. F[i] < F[x]

B. F[i]<=r

C. Vis[i]

D. i <= x

1.

A. F[4] = 0

B. F[1] = 4

C. F[1] = 2

D. F[4] = 1

2.

A. !Vis[n]

B. r < n

C. F[M] == INT_MAX

D. F[n] == INT_MAX

3.

A. F[i] == r

B. !Vis[i] && F[i] == r

C. F[i] < F[x]

D. !Vis[i] && F[i] < F[x]

4.

A. F[i] < F[x]

B. F[i]<=r

C. Vis[i]

D. i <= x

答案

1.T

2.A

3.C

4.D

5.A

6.9! （或 362880）

7.C

8.C

9.210

10.T

11.D

12.B

13.4

14.BAAADA

15.ACBBADAD

16.A

17.B

18.C

19.C

20.C

21.ABADBD

22.ABBBDC

23.3

24.4

25.ABABC

26.

1.正确答案： 0

2.正确答案： tmp+a[i]==ans / a[i]+tmp==ans / ans==a[i]+tmp

3.正确答案： <0

4.正确答案： i

5.正确答案： tmp+=a[i] / tmp=tmp+a[i]

27.

1. 正确答案： [1][1]

2. 正确答案： rowsum[i][0]=0;

3. 正确答案： rowsum[i][j-1]+matrix[i][j]

4. 正确答案： area=0

5. 正确答案： rowsum[i][last]-rowsum[i][first-1]

28.

1. 正确答案： rmax[n - 1] = x[n - 1]

2. 正确答案： rmax[i] = x[i]

3. 正确答案： rmax[i] = rmax[i + 1] + x[i]

4. 正确答案： rmax[i] = rmax[i + 1]

5. 正确答案： lmax[i + 1] + rmax[i + 1]

29.DBCAB

30.N 叉树

（1）、return 1 （或者 return 1L，或者 return 1I）

（2）、getcom(x - 1, y - 1)

（3）、s + t - p + 1

（4）、t++ （或者 ++t，或者 t = t + 1，或者 t += 1）

（5）、sum

（6）、0, len - 1, 0

31.BAACCB

32.DADC

GW 信奥 CSP 初赛习题集 6-6

算法——排序算法

1. 排序概述

1.判断题 [GESP202403【4 级】] 如果待排序数据不能都装进内存，需要使用外排序算法。

2.单选题 [NOIP 普及组 2006] 在下列各种排序算法中，不是以“比较”作为主要操作的算法是（ ）。

A. 选择排序

B. 冒泡排序

C. 插入排序

D. 基数排序

3.单选题 [NOIP 普及组 2018] 以下排序算法中，不需要进行关键字比较操作的算法是（ ）。

A. 基数排序

B. 冒泡排序

C. 堆排序

D. 直接插入排序

4.单选题 [GESP202312【6 级】] 内排序有不同的类别，下面哪种排序算法和冒泡排序是同一类？（ ）

A. 希尔排序

B. 快速排序

C. 堆排序

D. 插入排序

5.不定项选择题 [NOIP 提高组 2010] 原地排序是指在排序过程中 （除了存储待排序元素以外的）付诸空间的大小与数据规模无 关的排序算法。一下属于原地排序的有（ ）

A. 冒泡排序

B. 插入排序

C. 基数排序

D. 选择排序

2. 排序稳定性

6.单选题 [GESP202406【4 级】] 关于几种排序算法的说法，下面说法错误的是（ ）。

A. 选择排序不是一个稳定的排序算法

B. 冒泡排序算法不是一种稳定的排序算法

C. 插入排序是一种稳定的排序算法

D. 如果排序前 2 个相等的数在序列中的前后位置顺序和排序后它们 2 个的前后位置顺序相同，则称为一种稳定的排序算法

7.单选题 [GESP 样题【4 级】] 排序算法的稳定性是指（ ）

A. 相同元素在排序后的相对顺序保持不变

B. 排序算法的性能稳定

C. 排序算法对任意输入都有较好的效果

D. 排序算法容易实现

8.单选题 [GESP202306【4 级】] 排序算法是稳定的（Stable Sorting），就是指排序算法可以保证，在待排序数据中有两个相等记录的关键字 R 和 S（R 出现在 S 之前），在排序后的列表中 R 也一定在 S 前。下面关于排序稳定性的描述，正确的是（ ）。

A. 冒泡排序是不稳定的。

B. 插入排序是不稳定的。

C. 选择排序是不稳定的。

D. 以上都不正确。

9.判断题 [GESP 样题【5 级】] 选择排序和快速排序都是不稳定的。

10.判断题 [GESP202406【7 级】/GESP 样题【4 级】] 冒泡排序是稳定的排序算法。

11.单选题 [NOIP 普及组 2009] 排序算法是稳定的意思是关键码相同的记录排序前后相对位置不发生改变，下列哪种排序算法是不稳定的：

A. 冒泡排序

B. 插入排序

C. 归并排序

D. 快速排序

12.不定项选择题 [NOIP 提高组 2009] 排序算法是稳定的意思是关键码相同的记录排序前后相对位置不发生改变，下列哪些排序算法是稳定的：

A. 插入排序

B. 基数排序

C. 归并排序

D. 冒泡排序

13.单选题 [CSP-S2021] 以下排序方法中，（ ）是不稳定的。

A. 插入排序

B. 冒泡排序

C. 堆排序

D. 归并排序

14.单选题 [CSP-S2019] 排序的算法很多，若按排序的稳定性和不稳定性分类，下面算法中（）是不稳定排序。

A. 冒泡排序

B. 直接插入排序

C. 快速排序

D. 归并排序

15.单选题 [CSP-J2022] 以下排序算法的常见实现中，哪个选项的说法是错误的：（ ）。

A. 冒泡排序算法是稳定的

B. 简单选择排序是稳定的

C. 简单插入排序是稳定的

D. 归并排序算法是稳定的

16.不定项选择题 [NOIP 提高组 2017] 下列算法中，( )是稳定的排序算法。

A. 快速排序

B. 堆排序

C. 希尔排序

D. 插入排序

17.判断题 [GESP202406【5 级】] 归并排序和快速排序都采用递归实现，也都是不稳定排序。

3. 排序复杂度

18.判断题 [GESP202403【5 级】] 插入排序的时间复杂度是 O(NlogN)。

19.判断题 [GESP202309【4 级】] 对 N 个元素的数组执行插入排序算法，通常的时间复杂度是 $O(n^2)$。

20.单选题 [GESP202403【4 级】] 插入排序在最好情况下的时间复杂度是（ ）。

A.O(1)

B.O(N/2)

C.O(N)

D.O(N^2)

21.单选题 [GESP202309【4 级】] 对包含 n 个元素的数组进行冒泡排序，平均时间复杂度一般为（ ）。

A.O(n)

B.O(nlogn)

C.O(N^2)

D.以上都不正确

22.单选题 [NOIP 普及组 2010] 基于比较的排序时间复杂度的下限是( ), 其中 n 表示待排序的元素个数。

A. O(n)

B. O(nlogn)

C. O(logn)

D. O(N^2)

23.不定项选择题 [NOIP 提高组 2013/NOIP 普及组 2013] （ ）的平均时间复杂度为 O(nlog n) ，其中 n 是待排序的元素个数。

A. 快速排序

B. 插入排序

C. 冒泡排序

D. 归并排序

24.单选题 [GESP202406【4 级】] 关于直接插入排序，下列说法错误的是( )

A. 插入排序的最好情况是数组已经有序,此时只需要进行 次比较,时间复杂度为 O(n)

B. 最坏情况是数组逆序排序，此时需要进行你（n-1）/2 次比较以及 n-1 次赋值操作（插入）

C. 平均来说插入排序算法的复杂度为 O（N^2）

D. 空间复杂度上，直接插入法是就地排序，空间复杂度为 O(n)

25.判断题 [GESP202406【4 级】] 插入排序算法中，平均时间复杂度是 ，最坏的情况逆序情况下，达到最大时间复杂度。

26.判断题 [GESP202406【5 级】] 插入排序有时比快速排序时间复杂度更低。

27.单选题 [NOIP 普及组 2009] 快速排序最坏情况下的算法时间复杂度为：

A. O(log2n）

B. O(n）

C. O(nlog2n）

D. O(N^2）

28.单选题 [NOIP 提高组 2009] 快速排序平均情况和最坏情况下的算法时间复杂度分别为：

A. 平均情况 O(nlog2n），最坏情况 O(N^2）

B. 平均情况 O(n），最坏情况 O(N^2）

C. 平均情况 O(n），最坏情况 O(nlog2n）

D. 平均情况 O(log2n），最坏情况 O(N^2）

29.单选题 [CSP-S2023] 假设快速排序算法的输入是一个长度为 n 的已排序数组，且该快速排序算法在分治过程总是选择第一个元素作为基准元素。以下哪个选项描述的是在这种情况下的快速排序行为？

A. 快速排序对于此类输入的表现最好，因为数组已经排序。

B. 快速排序对于此类输入的时间复杂度是Θ(nlogn)。

C. 快速排序对于此类输入的时间复杂度是Θ(n^2)

D. 快速排序无法对此类数组进行排序，因为数组已经排序。

30.判断题 [GESP202312【5 级】] 插入排序有时比快速排序时间复杂度更低。

31.单选题 [NOIP 提高组 2011] 应用快速排序的分治思想，可以实现一个求第 K 大数的程序。假定不考虑极端的最坏情况，理论上可以实现的最低的算法时间复杂度为（ ）。

A. $O(N^2)$

B. O(nlogn)

C. O(n)

D. O(1)

32.判断题 [GESP202309【5 级】] 一般说来，冒泡排序算法优于归并排序。

33.判断题 [GESP202403【5 级】] 分治算法的典型应用之一是归并排序，其时间复杂度为 O(NlogN)。

34.判断题 [GESP202312【5 级】] 归并排序的时间复杂度是 O(NlogN)

35.单选题 [NOIP 提高组 2014] 以下时间复杂度不是 O（N^2)的排序方法是（)

A.插入排序

B.归并排序

C.冒泡排序

D.选择排序

36.单选题 [CSP-S2022] 考虑对 n 个数进行排序，以下最坏时间复杂度低于 $O(N^2)$ 的排序方法是（ ）。

A. 插入排序

B. 冒泡排序

C. 归并排序

D. 快速排序

37.不定项选择题 [NOIP 提高组 2017] 以下排序算法在最坏情况下时间复杂度最优的有( )。

A. 冒泡排序

B. 快速排序

C. 归并排序

D. 堆排序

38.单选题 [GESP202403【7 级】] 下列关于排序的说法，正确的是( )。

A. 冒泡排序是最快的排序算法之一。

B. 快速排序通常是不稳定的。

C. 最差情况，n 个元素做归并排序的时间复杂度为 O(n)。

D. 以上均不正确。

39.单选题 [GESP 样题【5 级】] 下面 C++代码用于排序，下列说法中错误的是（ ）。

```
void sort(int *arr, int n) {
    for (int i = 0; i < n - 1; ++i) {
        for (int j = 0; j < n - i - 1; ++j) {
            if (arr[j] > arr[j + 1]) {
                int tmp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = tmp;
            }
        }
    }
}
void sort(int *arr, int start, int end) {
    if (start >= end) {
        return;
    }
    int middle = (start + end) / 2;
    sort(arr, start, middle);
    sort(arr, middle + 1, end);
```

```
19
20      int leftSize = middle - start + 1;
21      int rightSize = end - middle;
22
23      int *left = new int[leftSize];
24      int *right = new int[rightSize];
25
26      for (int i = 0; i < leftSize; i++) {
27          left[i] = arr[start + i];
28      }
29      for (int j = 0; j < rightSize; j++) {
30          right[j] = arr[middle + 1 + j];
31      }
32
33      int i = 0, j = 0, k = start;
34      while (i < leftSize && j < rightSize) {
35          if (left[i] <= right[j]) {
36              arr[k] = left[i];
37              i++;
38          } else {
39              arr[k] = right[j];
40              j++;
41          }
42          k++;
43      }
44
45      while (i < leftSize) {
46          arr[k] = left[i];
47          i++;
48          k++;
49      }
50
51      while (j < rightSize) {
52          arr[k] = right[j];
53          j++;
54          k++;
55      }
56
57      delete[] left;
58      delete[] right;
59  }
```

A. 两种排序算法的时间复杂度不同。

B. 两种排序算法的空间复杂度一致。

C. sortA 的时间复杂度在最好和最坏情况下都是 $O(N^2)$。

D. sortB 的平均时间复杂度、最好情况的时间复杂度都是 O(logN)，最坏情况的时间复杂度是 $O(N^2)$。

4. 排序基本思想

40.单选题 [NOIP 提高组 2011/NOIP 普及组 2011] 体育课的上课铃声响了，同学们都陆续地奔向操场，按老师的要求从高到矮站成一排。每个同学按顺序来到操场时，都从排尾走向排头， 找到第一个比自己高的同学， 并站在 他的后面。这种站队的方法类似于（ ）算法。

A. 快速排序

B. 插入排序

C. 冒泡排序

D. 归并排序

41.单选题 [GESP202312【5 级】] 下面的排序算法都要处理多趟数据，哪种排序算法不能保证在下一趟处理时从待处理数据中选出最大或最小的数据？（ ）

A. 选择排序

B. 快速排序

C. 堆排序

D. 冒泡排序

42.单选题 [GESP202403【5 级】] 在快速排序中，选择的主元素（pivot）会影响算法的（ ）。

A. 不影响

B. 时间复杂度

C. 空间复杂度

D. 时间复杂度和空间复杂度

43.单选题 [NOIP 提高组 2012] 如果不在快速排序中引入随机化，有可能导致的后果是（ ）。

A. 数组访问越界

B. 陷入死循环

C. 排序结果错误

D. 排序时间退化为平方级

44.单选题 [CSP-S2022] 假设在基数排序过程中，受宇宙射线的影响，某项数据异变为一个完全不同的值。请问排 序算法结束后，可能出现的最坏情况是（ ）。

A. 移除受影响的数据后，最终序列是有序序列

B. 移除受影响的数据后，最终序列是前后两个有序的子序列

C. 移除受影响的数据后，最终序列是一个有序的子序列和一个基本无序的子序列

D. 移除受影响的数据后，最终序列基本无序

45.单选题 [GESP202309【5 级】] 归并排序算法的基本思想是（ ）。

A. 将数组分成两个子数组，分别排序后再合并。

B. 随机选择一个元素作为枢轴，将数组划分为两个部分。

C. 从数组的最后一个元素开始，依次与前一个元素比较并交换位置。

D. 比较相邻的两个元素，如果顺序错误就交换位置。

5. 排序操作次数

46.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 将 5 个数的序列排序，不论原先的顺序如何，最少都可以通过（ ）次比较，完成从小到大的排序。

A. 6

B. 7

C. 8

D. 9

47.填空题 [NOIP 普及组 2005/NOIP 提高组 2005] 将数组{32, 74, 25, 53, 28, 43, 86, 47}中的元素按从小到大的顺序排列，每次可以交换任意两个元素，最少需要交换？次。

48.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 将数组{8, 23, 4, 16, 77, -5, 53, 100}中的元素按从大到小的顺序排列，每次可以交换任意两个元素，最少需要交换（ ）次。

A. 4

B. 5

C. 6

D. 7

49.判断题 [GESP202403【4 级】] 对 int a[]={2,0,2,4,3,1,6} ，执行第一趟选择排序处理后 a 中数据变为 {0,2,2,4,3,1,6} 。

50.单选题 [GESP202406【4 级】] 数组{45,66,23,1,10,97,52,88,5,33}进行从小到大冒泡排序过程中，第一遍冒泡过后的序列是（ ）。

A. {45,23,1,10,66,52,88,5,33,97}

B. {45,66,1,23,10,97,52,88,5,33}

C. {45,66,23,1,10,52,88,5,33,97}

D. {45,66,23,1,10,97,52,88,33,5}

51.单选题 [CSP-S2021] 定义一种字符串操作为交换相邻两个字符。将“DACFEB”变为 “ABCDEF”最少需要（ ）次上述操作。

A. 7

B. 8

C. 9

D. 6

52.单选题 [GESP202406【5 级】/CSP-S2019] 设 A 和 B 是两个长度为 的有序数组，现将 A 和 B 合并成一个有序数组，归并排序算法在最坏情况下至少要做（ ）次比较。

A.n^2

B.nlogn

C.2n-1

D.n

53.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 设 A 和 B 是两个长为 n 的有序数组，现在需要将 A 和 B 合并成一个排好序的数组，任何以元素比较作为基本运算的归并算法在最坏情况下至少要做 ( )次比较。

A. N^2

B. nlogn

C. 2n

D. 2n-1

54.单选题 [NOIP 普及组 2012] 使用冒泡排序对序列进行升序排列，每执行一次交换操作系统将会减少 1 个 逆序对，因此序列 ： 5,4,3,2,1 需要执行（ ）次操作，才能完成冒泡排序。

A. 0

B. 5

C. 10

D. 15

55.单选题 [NOIP 普及组 2017] 对于给定的序列{ak}，我们把 (i, j) 称为逆序对当且仅当 i < j 且 ai > aj。那么序列 1, 7, 2, 3, 5, 4 的逆序对数为()个。

A. 4

B. 5

C. 6

D. 7

56.不定项选择题 [NOIP 提高组 2011] 对于序列“ 7,5,1,9,3,6,8,4 ”，在不改变顺序的情况下，去掉（ ）会使逆序 对的个数减少 3。

A. 7

B. 5

C. 3

D. 6

57.单选题 [GESP202309【5 级】] 下面代码用于归并排序，其中 merge() 函数被调用次数为（ ）。

```
#include <iostream>
using namespace std;

void mergeSort(int *listData, int start, int end);
void merge(int *listData, int start, int middle, int end);

void mergeSort(int *listData, int start, int end) {
    if (start >= end) {
        return;
    }
    int middle = (start + end) / 2;
    mergeSort(listData, start, middle);
    mergeSort(listData, middle + 1, end);
    merge(listData, start, middle, end);
}

void merge(int *listData, int start, int middle, int end) {
    int leftSize = middle - start + 1;
    int rightSize = end - middle;

    int *left = new int[leftSize];
    int *right = new int[rightSize];

    for (int i = 0; i < leftSize; i++) {
        left[i] = listData[start + i];
    }
```

```
    for (int j = 0; j < rightSize; j++) {
        right[j] = listData[middle + 1 + j];
    }

    int i = 0, j = 0, k = start;
    while (i < leftSize && j < rightSize) {
        if (left[i] <= right[j]) {
            listData[k] = left[i];
            i++;
        } else {
            listData[k] = right[j];
            j++;
        }
        k++;
    }

    while (i < leftSize) {
        listData[k] = left[i];
        i++;
        k++;
    }
    while (j < rightSize) {
        listData[k] = right[j];
        j++;
        k++;
    }

    delete[] left;
    delete[] right;
}

int main() {
    int listA[] = {1, 3, 2, 7, 11, 5};
    int size = sizeof(listA) / sizeof(listA[0]);

    mergeSort(listA, 0, size - 1);

    for (int i = 0; i < size; i++) {
        cout << listA[i] << "" "";
    }
    cout << endl;

    return 0;
}
```

A. 0

B. 1

C. 6

D. 7

6. 排序的实现与应用

1. 冒泡

58.单选题 [GESP 样题【4 级】/GESP202309【4 级】] 在下列代码的横线处填写（ ），完成对有 n 个 int 类型元素的数组 array 由小到大排序。

```
void BubbleSort(int array[], int n) {
    for (int i = n; i > 1; i--) {
        for (                    ) { // 在此处填充完整
            if (array[j] > array[j + 1]) {
                int t = array[j];
                array[j] = array[j + 1];
                array[j + 1] = t;
            }
        }
    }
}
```

A. int j = i - 2; j >= 0; j--

B. int j = i - 1; j >= 0; j--

C. int j = 0; j < i - 1; j++

D. int j = 0; j < i; j++

59.单选题 [GESP202312【5 级】] 下面的 C++用于对 lstA 排序，使得偶数在前奇数在后，横线处应填入( )。

```
bool isEven(int N)
{
    return N % 2 == 0;
}

```

```
void swap(int &a, int &b)
{
    int t;
    t = a;
    a = b;
    b = t;
    return;
}

void sort(int lstA[], int n)
{
    int i, j, t;
    for (i = n - 1; i > 0; i--)
        for (j = 0; j < i; j++)
            if (_____)
                swap(lstA[j], lstA[j + 1]);

    return;
}
```

A. isEven(lstA[j]) && !isEven(lstA[j+1])

B. !isEven(lstA[j]) && isEven(lstA[j+1])

C. lstA[j] > lstA[j+1]

D. lstA[j] < lstA[j+1]

60.单选题 [GESP 样题【8 级】] 下面的冒泡排序中尝试提前结束比较过程，横线处应该填写的代码是（ ）。

```
void BubbleSort(int R[], int n)
{
    int i, j, lastExchangeIndex;
    i = n;
    while (i > 1)
    {
        lastExchangeIndex = 1;
        for (j = 1; j < i; j++)
        {
            if (R[j] > R[j+1])
            {
                int t;
                t = R[j];
                R[j] = R[j+1];
```

```
                    R[j+1] = t;
                    lastExchangeIndex = j;
                } //if
            }
            _____________________;
        } // while
} // BubbleSort
```

A. i = lastExchangeIndex + 1

B. i = lastExchangeIndex -1

C. i = lastExchangeIndex

D. i = lastExchangeIndex - j

61.填空题 [NOIP 提高组 2012]

```
#include <iostream>
using namespace std;
int n, i, temp, sum, a[100];
int main(){
    cin>>n;
    for (i = 1;i <= n;i++)
        cin>>a[i];
    for (i = 1;i <= n - 1;i++)
        if (a[i] > a[i + 1]) {
            temp = a[i];
            a[i] = a[i + 1];
            a[i + 1] = temp;
        }
    for (i = n;i >= 2;i--)
        if (a[i] < a[i - 1]) {
            temp = a[i];
            a[i] = a[i - 1];
            a[i - 1] = temp;
        }
    sum = 0;
    for (i = 2;i <= n - 1;i++)
        sum += a[i];
    cout<<sum / (n - 2)<<endl;
    return 0;
}
```

输入：

8

40 70 50 70 20 40 10 30

输出：____

62. [NOIP 提高组 2014]

```
#include <iostream>
#include <string>
using namespace std;
const int SIZE = 100;

int main(){
    string dict[SIZE];
    int rank[SIZE];
    int ind[SIZE];
    int i, j, n, tmp;
    cin >> n;
    for (i = 1; i <= n; i++){
        rank[i] = i;
        ind[i] = i;
        cin >> dict[i];
    }
    for(i = 1; i < n; i++){
        for(j = 1; j <= n - i; j++){
            if(dict[ind[j]] > dict[ind[j + 1]]){
                tmp = ind[j];
                ind[j] = ind[j + 1];
                ind[j + 1] = tmp;
            }
        }
    }
    for(i = 1; i <= n; i++)
        rank[ind[i]] = i;
    for(i = 1; i <= n; i++)
        cout << rank[i] << "" "";

    cout << endl;
    return 0;
}
```

输入：

7

AAA 级

阿坝

血脑屏障

AAA 级

AAA 级

CCC 公司

机 管 局

输出：（ ）

63.单选题 [GESP202406【4 级】] 下列程序横线处，应该输入的是 （ ）。

```
#include<iostream>
using namespace std;
int n,a[10001];
void swap(int &a,int &b)
{
    int t=a;
    a=b;
    b=t;
}
int main()
{
    cin>>n;
    for(int i=1;i<=n;i++)
        cin>>a[i];
    for(int i=n;i>1;i--)
        for(int j=1;j<i;j++)
            if(a[j]>a[j+1])
                swap(a[j],a[j+1]);
    for(int i=1;i<=n;i++)
        cout<<a[i]<<" ";
    cout<<endl;
    return 0;
}
```

A. swap(a[j],a[j+1]);

B. swap(a[j-1],a[j]);

C. swap(a[j-1],a[j+1]);

D. swap(&a[j-1],&a[j+1]);

2. 插入

64.单选题 [GESP202406【4 级】] 下面的排序算法程序中，横线处应该填入的是（ ）。

```
int a[8]={ 2,3, 4, 5, 6,2,3,1};
for (int i=1;i<8;i++)
{
    int key = a[i];
    int j=i-1;
    while(a[j]>key && j>=0)
{
        _________;
        j -= 1;
    }
    a[j + 1]= key;
}
```

A. a[j]=a[j-1];

B. a[j]=a[j+1];

C. a[j+1]=a[j-1];

D. a[j+1]=a[j];

3. 选择

65.单选题 [GESP202306【4 级】] 在下列代码的横线处填写（ ），完成对有 n 个 int 类型元素的数组 array 由小到大排序。

```
void SelectionSort(int array[], int n) {
    int i, j, min, temp;
    for (i = 0; i < n - 1; i++) {
        min = i;
        for (j = i + 1; j < n; j++) {
            // 此处缺少代码
            if (_____________) { // 在此处进行比较
                min = j;
            }
        }
        temp = array[min];
        array[min] = array[i];
        array[i] = temp;
    }
```

```
15 }
```

A. array[min] > array[j]

B. array[min] > array[i]

C. min > array[j]

D. min > array[i]

4. 计数

66.单选题 [CSP-J2019] （计数排序）计数排序是一个广泛使用的排序方法。下面的程序使用双关键字计数排序，对 n 对 10000 以内的整数，从小到大排序。

例如有三对整数（3，4）、（2，4）、（3，3），那么排序之后应该是（2，4）、（3，3）、（3，4）。

输入第一行为 n，接下来 n 行，第 i 行有两个数 a[i] 和 b[i]，分别表示第 i 对整数的第一关键字和第二关键字。

数据范围≤n≤10^7,1≤a[i],b[i]≤10^4。

提示：应先对第二关键字排序，再对第一关键字排序。数组 ord[] 存储第二关键字排序的结果，数组 res[] 存储双关键字排序的结果。

试补全程序

```
1  #include <cstdio>
2  #include <cstring>
3  using namespace std;
4  const int maxn = 10000000;
5  const int maxs = 10000;
6
7  int n;
8  unsigned a[maxn], b[maxn],res[maxn], ord[maxn];
9  unsigned cnt[maxs + 1];
10 int main() {
11     scanf(""%d"", &n);
12     for (int i = 0; i < n; ++i)
13         scanf(""%d%d"", &a[i], &b[i]);
14     memset(cnt, 0, sizeof(cnt));
15     for (int i = 0; i < n; ++i)
16         ①; // 利用 cnt 数组统计数量
17     for (int i = 0; i < maxs; ++i)
```

```
            cnt[i + 1] += cnt[i];
        for (int i = 0; i < n; ++i)
            ②; // 记录初步排序结果
        memset(cnt, 0, sizeof(cnt));
        for (int i = 0; i < n; ++i)
            ③; // 利用 cnt 数组统计数量
        for (int i = 0; i < maxs; ++i)
            cnt[i + 1] += cnt[i];
        for (int i = n - 1; i >= 0; --i)
            ④ // 记录最终排序结果
        for (int i = 0; i < n; i++)
            printf(""%d %d"", ⑤);

        return 0;
    }
```

①处应填（ ）

②处应填（ ）

③处应填（ ）

④处应填（ ）

⑤处应填（ ）

1.

A. ++cnt[i]

B. ++cnt[b[i]]

C. ++cnt[a[i] * maxs + b[i]]

D. ++cnt[a[i]]

2.

A. ord[--cnt[a[i]]] = i

B. ord[--cnt[b[i]]] = a[i]

C. ord[--cnt[a[i]]] = b[i]

D. ord[--cnt[b[i]]] = i

3.

A. ++cnt[b[i]]

B. ++cnt[a[i] * maxs + b[i]]

C. ++cnt[a[i]]

D. ++cnt[i]

4.

A. res[--cnt[a[ord[i]]]] = ord[i]

B. res[--cnt[b[ord[i]]]] = ord[i]

C. res[--cnt[b[i]]] = ord[i]

D. res[--cnt[a[i]]] = ord[i]

5.

A. a[i], b[i]

B. a[res[i]], b[res[i]]

C. a[ord[res[i]]],b[ord[res[i]]]

D. a[res[ord[i]]],b[res[ord[i]]]

5. 快速

67.单选题 [GESP 样题【6 级】] 下面 C++代码实现了某种排序算法，其中代码片段 my_sort(arr, begin, i); my_sort(arr, i + 1, end);采用的算法思想是（ ）。

```
void my_sort(int arr[], int begin, int end) {
    if (begin >= end - 1) return;
    int i = begin, j = end - 1, x = arr[begin];
    while (i < j) {
        while (i < j && arr[j] >= x)
            j--;
        if (i < j)
            arr[i++] = arr[j];

        while (i < j && arr[i] < x)
            i++;
        if (i < j)
            arr[j--] = arr[i];
    }
    arr[i] = x;
    my_sort(arr, begin, i);
    my_sort(arr, i + 1, end);
}
```

A．递推

B．贪心

C．分治

D．搜索

68.填空题 [NOIP 提高组 2005]

```
#include <stdio.h>
#include <math.h>
int a[50];
void work(int p, int r) {
    if (p < r) {
        int i = p - 1, j, temp;
        for (j = p; j < r; j++) {
            if (a[j] >= a[r]) {
                i++;
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            }
        }
        temp = a[i + 1];
        a[i + 1] = a[r];
        a[r] = temp;
        work(p, i);
        work(i + 2, r);
    }
}
int main() {
    int n, i, sum = 0;
    scanf("%d", &n);
    for (i = 0; i < n; i++)
        scanf("%d", &(a[i]));
    work(0, n - 1);
    for (i = 0; i < n - 1; i++)
        sum += abs(a[i + 1] - a[i]);
    printf("%d\n", sum);
    return 0;
}
```

输入：10 23 435 12 345 3123 43 456 12 32 -100

输出：

69.单选题 [GESP202312【5 级】] 下面的 C++代码实现对 list 的快速排序，有关说法，错误的是（ ）。

```
vector<int> operator+(const vector<int>& lA, const vector<int>& lB) {
    vector<int> lst;
    for (size_t i = 0; i < lA.size(); ++i) {
        lst.push_back(lA[i]);
    }
    for (size_t i = 0; i < lB.size(); ++i) {
        lst.push_back(lB[i]);
    }
    return lst;
}

// 快速排序函数
vector<int> qSort(const vector<int>& lst) {
    if (lst.size() < 2) {
        return lst;
    }
    int pivot = lst[0];
    vector<int> less, greater;
    for (size_t i = 1; i < lst.size(); ++i) {
        if (lst[i] <= pivot) {
            less.push_back(lst[i]);
        } else {
            greater.push_back(lst[i]);
        }
    }
    return _____________________________________;
}
```

A. qSort(less) + qSort(greater) + (vector<int>)pivot

B. (vector<int>)pivot + (qSort(less) + qSort(greater))

C. (qSort(less) + (vector<int>)pivot + qSort(greater))

D. qSort(less) + pivot + qSort(greater)

70.填空题 [NOIP 普及组 2008/NOIP 提高组 2008] （找第 k 大的数） 给定一个长度为 10^6 的无序正整数序列，以及另一个数 n(1≤n≤10^6)，然后以类似快速排序的方法找到序列中第 n 大的数（关于第 n 大的数：例如序列 {1,2,3,4,5,6} 中第 3 大的数是 4）。

```
#include <iostream>
using namespace std;

int a[1000001],n,ans = -1;
void swap(int &a,int &b)
{
    int c;
    c = a; a = b;    b = c;
}

int FindKth(int left, int right, int n)
{
    int tmp,value,i,j;
    if (left == right) return left;
    tmp = rand()% (right - left) + left;
    swap(a[tmp],a[left]);
    value =[        ①        ];
    i = left;
    j = right;
    while (i < j)
    {
        while (i < j && [        ②        ]) j --;
        if (i < j) {a[i] = a[j]; i ++;} else break;
        while (i < j && [        ③        ]) i ++;
        if (i < j) {a[j] = a[i]; j - -;} else break;
    }
    [        ④        ]
    if (i < n) return   FindKth([        ⑤        ] );
    if (i > n) return [        ⑥        ]
    return i;
}

int main()
{
    int i;
    int m = 1000000;
    for (i = 1;i <= m;i ++)
        cin >> a[i];
    cin >> n;
    ans = FindKth(1,m,n);
    cout << a[ans];
    return 0;
}
```

71.单选题 [CSP-S2020]

```
#include <iostream>
#include <cstdlib>
using namespace std;

int n;
int d[10000];

int find(int L, int R, int k) {
    int x = rand() % (R - L + 1) + L;
    swap(d[L], d[x]);
    int a = L + 1, b = R;
    while (a < b) {
        while (a < b && d[a] < d[L])
            ++a;
        while (a < b && d[b] >= d[L])
            --b;
        swap(d[a], d[b]);
    }
    if (d[a] < d[L])
        ++a;
    if (a - L == k)
        return d[L];
    if (a - L < k)
        return find(a, R, k - (a - L));
    return find(L + 1, a - 1, k);
}

int main() {
    int k;
    cin >> n;
    cin >> k;
    for (int i = 0; i < n; ++i)
        cin >> d[i];
    cout << find(0, n - 1, k);
    return 0;
}
```

假设输入的 n,k 和 d[i]都是不超过 10000 的正整数，且 k 不超过 n，并假设 rand() 函数产生的是均匀的随机数，完成下面的判断题和单选题：

判断题

第 9 行的 x 的数值范围是 L+1 到 R，即 [L+1,R]。（ ）

将第 19 行的 d[a] 改为 d[b]，程序不会发生运行错误。（ ）

单选题

（2.5 分）当输入的 d[i]是严格单调递增序列时，第 17 行的 swap 平均执行次数是( )。【编者注：本题为错题，请选择 A 获取对应的分数】

（2.5 分）当输入的 d[i]是严格单调递减序列时，第 17 行的 swap 平均执行次数是( )。

（2.5 分）若输入的 d[i]为 i，此程序①平均的时间复杂度和②最坏情况下的时间复杂度分别是( )。

（2.5 分）若输入的 d[i]都为同一个数，此程序平均的时间复杂度是( )。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. O(nlogn)

B. O(n)

C. O(logn)

D. $O(N^2)$

4.

A. $O(N^2)$

B. O(n)

C. O(nlogn)

D. O(logn)

5.

A. $O(n),O(N^2)$

B. O(n),O(nlogn)

C. $O(nlogn),O(N^2)$

D. O(nlogn),O(nlogn)

6.

A. O(n)

B. O(logn)

C. O(nlogn)

D. $O(N^2)$

72.填空题 [NOIP 普及组 2008]

```
#include <iostream>
using namespace std;
void func(int ary[], int n )
{
    int i=0, j, x;
    j=n-1;
    while(i<j)
    {
        while (i<j&&ary[i]>0) i++;
        while (i<j&&ary[j]<0) j--;
        if (i<j){
            x=ary[i];
            ary[i++]=ary[j];
            ary[j--]=x;
        }
    }
}
int main()
{

    int a[20], i, m;
    m=10;
    for(i=0; i<m; i++)
    {
        cin>>a[i];
    }
    func(a, m);
    for (i=0; i<m; i++)
        cout<<a[i]<<"" "";
    cout<< endl;
    return 0;
}
```

输入：5 4 -6 -11 6 -59 22 -6 1 10

输出：

73.单选题 [GESP202406【5 级】] 为了正确实现快速排序，下面横线上的代码应为（ ）。

```
vector<int> linear_sieve(int n) { vector<bool> is_prime(n + 1, true); vector<int> primes; is_prime[0] = is_prime[1] = 0; //0 和 1 两个数特殊处理 for (int i = 2; i <= n; ++i) { if (is_prime[i]) {
primes.push_back(i); } _____ { // 在此处填入代码 is_prime[i * primes[j]] = 0; if (i % primes[j] == 0) break; } } return primes; }
1 2 3 4 5 6
7 8 9
10 11 12 13 14 15 16
void qsort(vector<int>& arr, int left, int right) { int i, j, mid; int pivot;
i = left; j = right; mid = (left + right) / 2; // 计算中间元素的索引 pivot = arr[mid]; // 选择中间元素作为基准值
do {
while (arr[i] < pivot) i++; while (arr[j] > pivot) j--; if (i <= j) { swap(arr[i], arr[j]); // 交换两个元素 i++; j--; } } _____; // 在此处填入代码 if (left < j) qsort(arr, left, j); // 对左子数组进行快速排序 if (i < right) qsort(arr, i, right); // 对右子数组进行快速排序 }
1 2 3 4 5 6 7 8 9
10
11 12 13 14 15 16 17 18 19 20
```

A. while (i <= mid)

B. while (i < mid)

C. while (i < j)

D. while (i <= j)

6. 归并

74.单选题 [GESP 样题【5 级】] `sortB`函数，明显体现出的算法思想和编程方法包括（ ）。

```
void sortB(int *arr, int start, int end) {
    if (start >= end) {
        return;
    }
```

```
    int middle = (start + end) / 2;
    sort(arr, start, middle);
    sort(arr, middle + 1, end);

    int leftSize = middle - start + 1;
    int rightSize = end - middle;

    int *left = new int[leftSize];
    int *right = new int[rightSize];

    for (int i = 0; i < leftSize; i++) {
        left[i] = arr[start + i];
    }
    for (int j = 0; j < rightSize; j++) {
        right[j] = arr[middle + 1 + j];
    }

    int i = 0, j = 0, k = start;
    while (i < leftSize && j < rightSize) {
        if (left[i] <= right[j]) {
            arr[k] = left[i];
            i++;
        } else {
            arr[k] = right[j];
            j++;
        }
        k++;
    }

    while (i < leftSize) {
        arr[k] = left[i];
        i++;
        k++;
    }

    while (j < rightSize) {
        arr[k] = right[j];
        j++;
        k++;
    }

    delete[] left;
    delete[] right;
}
```

A. 递归

B. 分治

C. A、B 都正确

D. A、B 都不正确

75.填空题 [NOIP 提高组 2010]

```
#include<iostream>
using namespace std;
int main()
{
    const int SIZE=100;
    int na,nb,a[SIZE],b[SIZE],i,j,k;
    cin>>na;
    for(i=1;i<=na;i++)
        cin>>a[i];
    cin>>nb;
    for(i=1;i<=nb;i++)
        cin>>b[i];
    i=1;
    j=1;
    while( (i<=na)&&(j<=nb) )
    {
        if(a[i]<=b[j])
        {
            cout<<a[i]<<' ';
            i++;
        }
        else
        {
            cout<<b[j]<<' ';
            j++;
        }
    }
    if(i<=na)
        for(k=i;k<=na;k++)
            cout<<a[k]<<' ';
    if(j<=nb)
        for(k=j;k<=nb;k++)
            cout<<b[k]<<' ';
    return 0;
}
```

输入：

5

1 3 5 7 9

4

2 6 10 14

输出：_____

76.单选题 [GESP202312【5 级】] 下面 C++代码以递归方式实现合并排序，并假设 merge (int T[], int R[], int s, int m, int t) 函数将有序（同样排序规则）的 T[s..m]和 T[m+1..t]归并到 R[s..t]中。横线处应填上代码是( )。

```
void mergeSort(int SList[], int TList[], int s, int t, int len)
{
    if (s == t){
        TList[s] = SList[s];
        return;
    }
    int *T2 = new int[len]; // 保存中间结果
    int m = (s + t) / 2;
    ____________________
    merge(T2, SList, s, m, t);
    delete[] T2;
    return ;
}
```

A. mergeSort(SList, T2, s, m,len), mergeSort(SList, T2, m,t,len)

B. mergeSort(SList, T2, s, m-1,len), mergeSort(SList, T2, m+1,t,len)

C. mergeSort(SList, T2, s, m,len), mergeSort(SList, T2, m+1,t,len)

D. mergeSort(SList, T2, s, m-1,len), mergeSort(SList, T2, m-1,t,len)

77.单选题 [GESP202309【5 级】] 在下述的归并排序算法中， mergeSort(listData, start, middle); 和 mergeSort(listData, middle+ 1, end); 涉及到的算法为（ ）。

```
#include <iostream>
using namespace std;

void mergeSort(int *listData, int start, int end);
```

```
5  void merge(int *listData, int start, int middle, int end);
6
7  void mergeSort(int *listData, int start, int end) {
8      if (start >= end) {
9          return;
10     }
11     int middle = (start + end) / 2;
12     mergeSort(listData, start, middle);
13     mergeSort(listData, middle + 1, end);
14     merge(listData, start, middle, end);
15 }
16
17 void merge(int *listData, int start, int middle, int end) {
18     int leftSize = middle - start + 1;
19     int rightSize = end - middle;
20
21     int *left = new int[leftSize];
22     int *right = new int[rightSize];
23
24     for (int i = 0; i < leftSize; i++) {
25         left[i] = listData[start + i];
26     }
27     for (int j = 0; j < rightSize; j++) {
28         right[j] = listData[middle + 1 + j];
29     }
30
31     int i = 0, j = 0, k = start;
32     while (i < leftSize && j < rightSize) {
33         if (left[i] <= right[j]) {
34             listData[k] = left[i];
35             i++;
36         } else {
37             listData[k] = right[j];
38             j++;
39         }
40         k++;
41     }
42
43     while (i < leftSize) {
44         listData[k] = left[i];
45         i++;
46         k++;
47     }
48     while (j < rightSize) {
49         listData[k] = right[j];
```

```
            j++;
            k++;
        }

        delete[] left;
        delete[] right;
}

int main() {
        int listA[] = {1, 3, 2, 7, 11, 5};
        int size = sizeof(listA) / sizeof(listA[0]);

        mergeSort(listA, 0, size - 1);

        for (int i = 0; i < size; i++) {
            cout << listA[i] << "" "";
        }
        cout << endl;

        return 0;
}
```

A. 搜索算法

B. 分治算法

C. 贪心算法

D. 递推算法

78.填空题 [NOIP 提高组 2012]

```
#include <iostream>
using namespace std;
const int SIZE = 20;
int data[SIZE];
int n, i, h, ans;
void merge(){
        data[h-1] = data[h-1] + data[h];
        h--;
        ans++;
}
int main(){
        cin>>n;
        h = 1;
        data[h] = 1;
```

```
    ans = 0;
    for (i = 2;i <= n;i++){
        h++;
        data[h] = 1;
        while (h > 1 && data[h] == data[h-1])
            merge();
    }
    cout << ans << endl;
}
```

1、输入：8

输出：____

2、输入：2012

输出：____

79.单选题 [CSP-S2022] （归并第 k 小） 已知两个长度均为 n 的有序数组 a1 和 a2（均为递增序，但不保证严 格单调递增），并且给定正整数 k（1≤k≤2n），求数组 a1 和 a2 归并排序后的数组里 第 k 小的数值。

试补全程序。

```
#include <bits/stdc++.h>
using namespace std;

int solve(int *a1, int *a2, int n, int k) {
    int left1 = 0, right1 = n - 1;
    int left2 = 0, right2 = n - 1;
    while (left1 <= right1 && left2 <= right2) {
        int m1 = (left1 + right1) >> 1;
        int m2 = (left2 + right2) >> 1;
        int cnt = ①;
        if (②) {
            if (cnt < k) left1 = m1 + 1;
            else right2 = m2 - 1;
        } else {
            if (cnt < k) left2 = m2 + 1;
            else right1 = m1 - 1;
        }
    }
    if (③) {
        if (left1 == 0) {
            return a2[k - 1];
```

```
            } else {
                int x = a1[left1 - 1], ④;
                return std::max(x, y);
            }
        } else {
                if (left2 == 0) {
                    return a1[k - 1];
                } else {
                    int x = a2[left2 - 1], ⑤;
                    return std:: max(x, y);
                }
        }
}
```

①~⑤处应填（ ）

1.

A. (m1 + m2) * 2

B. (m1 - 1) + (m2 - 1)

C. m1 + m2

D. (m1 + 1) + (m2 + 1)

2.

A. a1[m1] == a2[m2]

B. a1[m1] <= a2[m2]

C. a1[m1] >= a2[m2]

D. a1[m1] != a2[m2]

3.

A. left1 == right1

B. left1 < right1

C. left1 > right1

D. left1 != right1

4.

A. y = a1[k - left2 - 1]

B. y = a1[k - left2]

C. y = a2[k - left1 - 1]

D. y = a2[k - left1]

5.

A. y = a1[k - left2 - 1]

B. y = a1[k - left2]

C. y = a2[k - left1 - 1]

D. y = a2[k - left1]

80.填空题 [NOIP 提高组 2017]

```
#include <iostream>
using namespace std;
int n, s, a[100005], t[100005], i;
void mergesort(int l, int r)
{
    if (l == r)
        return;
    int mid = (l + r) / 2;
    int p = l;
    int i = l;
    int j = mid + 1;
    mergesort(l, mid);
    mergesort(mid + 1, r);
    while (i <= mid && j <= r)
    {
        if (a[j] < a[i])
        {
            s += mid - i + 1;
            t[p] = a[j];
            p++;
            j++;
        }
        else
        {
            t[p] = a[i];
            p++;
            i++;
        }
    }
    while (i <= mid)
    {
        t[p] = a[i];
        p++;
```

```
            i++;
        }
        while (j <= r)
        {
            t[p] = a[j];
            p++;
            j++;
        }
        for (i = l; i <= r; i++)
            a[i] = t[i];
}
int main()
{
        cin >> n;
        for (i = 1; i <= n; i++)
            cin >> a[i];
        mergesort(1, n);
        cout << s << endl;
        return 0;
}
```

输入：6

2 6 3 4 5 1

输出：_____

答案

答案

1.T

2.D

3.A

4.B

5.ABD

6.B

7.A

8.C

9.T

10.T

11.D

12.ABCD

13.C

14.A

15.B

16.D

17.F

18.F

19.T

20.C

21.C

22.B

23.AD

24.D

25.T

26.T

27.D

28.A

29.C

30.T

31.C

32.F

33.T

34.T

35.B

36.C

37.CD

38.B

39.D

40.B

41.B

42.B

43.D

44.A

45.A

46.B

47.5

48.B

49.T

50.A

51.A

52.C

53.D

54.C

55.B

56.CD

57.C

58.C

59.A

60.C

61.41

62.2 5 6 3 4 7 1

63.A

64.D

65.A

66.BDCAB

67.C

68.3223

69.C

70.

1.正确答案：a[left]

2.正确答案：a[j]<value / a[j]<=value / value>a[j] / value>=a[j]

3.正确答案：a[i]>value / a[i]>=value

4.正确答案：a[i]=value;

5.正确答案：i+1,right,n

6.正确答案：FindKth(left,i-1,n);

71.BAABAD

72.5 4 10 1 6 22 -59 -6 -11 -6

73.D

74.C

75.1 2 3 5 6 7 9 10 14

76.C

77.B

78.7 2004

79.CBCCA

80.8

GW 信奥 CSP 初赛习题集 6-7

算法——模拟算法

1. 高精度计算

1.单选题 [CSP-J2023] 以下关于高精度运算的说法错误的是()

A.高精度计算主要是用来处理大整数或需要保留多位小数的运算

B.大整数除以小整数的处理的步骤可以是，将被除数和除数对齐，从左到右逐位尝试将除数乘以某个数，通过减法得到新的被除数，并累加商

C.高精度乘法的运算时间只与参与运算的两个整数中长度较长者的位数有关

D.高精度加法运算的关键在于逐位相加并处理进位

2.单选题 [GESP202312【5 级】] 下面的 C++代码使用数组模拟整数加法，可以处理超出大整数范围的加法运算。横线处应填入代码是（ ）。

```
vector<int> operator+(vector<int> a, vector<int> b) {
    vector<int> c;
    int t = 0;

    for (int i = 0; i < a.size() || i < b.size(); i++) {
        if (i < a.size()) t = t + a[i];
        if (i < b.size()) t = t + b[i];
        ________________________________
    }

    if (t) c.push_back(t);
    return c;
}
```

A. c.push_back(t % 10), t = t % 10;

B. c.push_back(t / 10), t = t % 10;

C. c.push_back(t / 10), t = t / 10;

D. c.push_back(t % 10), t = t / 10;

3.单选题 [GESP202403【5 级】] 下面的代码片段用于将两个高精度整数进行相加。请在横线处填入（ ），使其能正确实现相应功能。

```
string add(string num1, string num2) {
    string result;
    int carry = 0;
    int i = num1.size() - 1, j = num2.size() - 1;
    while (i >= 0 || j >= 0 || carry) {
        int x = (i >= 0) ? num1[i--] - '0' : 0;
        int y = (j >= 0) ? num2[j--] - '0' : 0;
        int sum = x + y + carry;
        carry = sum / 10;
        ____________________________________
    }
    return result;
}
```

A. result = to_string(sum % 10) + result;

B. result = to_string(carry % 10) + result;

C. result = to_string(sum / 10) + result;

D. result = to_string(sum % 10 + carry) + result;

4.单选题 [CSP-J2020]

```
#include <iostream>
using namespace std;

long long n, ans;
int k, len;
long long d[1000000];

int main() {
    cin >> n >> k;
    d[0] = 0;
    len = 1;
    ans = 0;
    for (long long i = 0; i < n; ++i) {
        ++d[0];
        for (int j = 0; j + 1 < len; ++j) {
            if (d[j] == k) {
                d[j] = 0;
```

```
                    d[j + 1] += 1;
                    ++ans;
                }
            }
            if (d[len - 1] == k) {
                d[len - 1] = 0;
                d[len] = 1;
                ++len;
                ++ans;
            }
        }
        cout << ans << endl;
        return 0;
```

假设输入的 n 是不超过 2^62 的正整数，k 都是不超过 10000 的正整数。

判断题

1）若 k=1，则输出 ans 时，len = n。（ ）

2）若 k>1，则输出 ans 时，len 一定小于 n。（ ）

3）若 k>1，则输出 ans 时，klen 一定大于 n。（ ）

单选题

若输入的 n 等于 10^15，输入的 k 为 1，则输出等于（ ）。

若输入的 n 等于 205,891,132,094,649(即 3^30)，输入的 k 为 3，则输出等于

若输入的 n 等于 100,010,002,000,090，输入的 k 为 10，则输出等于()。

1.

A. 正确

B. 错误

2.

A. 正确

B.错误

3.

A. 正确

B.错误

4.

A.1

B.(10^30 - 10^15)/2

C.(10^30 + 10^15)/2

D. 10^15

5.

A. 3^30

B.(3^30 - 1)/2

C.3^30-1

D.(3^30+ 1)/2

6.

A.11,112,222,444,543

B.11,122,222,444,453

C.11,122,222,444,543

D.11,112,222,444,453

5.填空题 [NOIP 提高组 2009]

```
#include <iostream>
using namespace std;

int main()
{
    int n,m,i,j,p,k;
    int a[100],b[100];
    cin >> n >> m;
    a[0]=n;
    i=0;
    p=0;
    k=0;
    do
    {
        for (j=0;j<i;j++)
            if (a[i]==a[j])
            {
                p=1;
                k=j;
                break;
            }
```

```
        if (p)
            break;
        b[i]=a[i]/m;
        a[i+1]=a[i]%m*10;
        i++;
    }while (a[i]!=0);

    cout << b[0] << ""."";
    for (j=1; j<k; j++)
        cout << b[j];
    if (p)
        cout << ""("";
    for (j=k;j<i;j++)
        cout << b[j];
    if (p)
        cout << "")"";
    cout << endl;
    return 0;
}
```

输入：5 13

输出：_____

6.填空题 [NOIP 普及组 2011/NOIP 提高组 2011]（ 大整数开方 ）输入一个正整数 n（1≤n≤10^100），试用二分法计算它的平方根的整数部分。

```
#include <iostream>
#include <string>
using namespace std;
const int SIZE = 200;
struct hugeint {
  int len, num[SIZE];
};
//其中 len 表示大整数的位数；num[1]表示个位、num[2]表示十位，以此类推
hugeint times(hugeint a, hugeint b) {
    //计算大整数 a 和 b 的乘积
    int i, j; hugeint ans;
    memset(ans.num, 0, sizeof(ans.num));
    for (i = 1; i <= a.len; i++)
        for (j = 1; j <= b.len; j++)
            ①+= a.num[i] * b.num[j];
    for (i = 1; i <= a.len + b.len; i++) {
        ans.num[i + 1] += ans.num[i] / 10;
```

```
        ②;
    }
    if (ans.num[a.len + b.len] > 0)
        ans.len = a.len + b.len;
    else
        ans.len = a.len + b.len - 1;
    return ans;
}
hugeint add(hugeint a, hugeint b){
    //计算大整数 a 和 b 的和
    int i; hugeint ans;
    memset(ans.num, 0, sizeof(ans.num));
    if (a.len > b.len)
        ans.len = a.len;
    else
        ans.len = b.len;
    for (i = 1; i <= ans.len; i++) {
        ans.num[i] +=③;
        ans.num[i + 1] += ans.num[i] / 10;
        ans.num[i] %= 10;
    }
    if (ans.num[ans.len + 1] > 0) ans.len++;
    return ans;
}
hugeint average(hugeint a, hugeint b){
    //计算大整数 a 和 b 的平均数的整数部分
    int i; hugeint ans;
    ans= add(a, b);
    for(i = ans.len; i >= 2; i--) {
        ans.num[i  - 1] += (④) * 10;
        ans.num[i] /= 2;
    }
    ans.num[1] /= 2;
    if (ans.num[ans.len] == 0) ans.len--;
    return ans;
}
hugeint plustwo(hugeint a) {
    //计算大整数 a 加 2 后的结果
    int i; hugeint ans;
    ans = a;
    ans.num[1] += 2;
    i = 1;
    While ((i <= ans.len) && (ans.num[i] >= 10)) {
        ans.num[i + 1] += ans.num[i] / 10;
        ans.num[i] %= 10;
```

```
            i++;
        }
        if (ans.num[ans.len + 1] > 0)⑤;
        return ans;
}
bool over(hugeint a, hugeint b){
//若大整数 a>b 则返回 true，否则返回 false
        int i;
        if (⑥) return false;
        if (a.len > b.len) return true;
        for (i = a.len; i >= 1; i--) {
            if (a.num[i] < b.num[i])return false;
            if (a.num[i] > b.num[i]) return true;
        }
        return false;
}
int main(){
        string s;
        int i;
        hugeint target, left, middle, right;
        cin>>s;
        memset(target.num, 0, sizeof(target.num));
        target.len = s.length();
        for (i = 1; i <= target.len; i++)
            target.num[i]= s[target.len  - i] - ⑦;
        memset(left.num, 0, sizeof(left.num));
        left.len = 1;
        left.num[1] = 1;
        right = target;
        do {
            middle = average(left, right);
            if (over(⑧))right = middle;
            else left = middle;
        } while (!over(plustwo(left), right));
        for (i = left.len; i >= 1; i--)
            cout<<left.num[i];
        cout<<endl;
        return 0;
}
```

7.填空题 [NOIP 提高组 2017] (大整数除法)给定两个正整数 p 和 q，其中 p 不超过 10^100，q 不超过 100000，求 p 除以 q 的商和余数。(第一空 2 分，其余 3 分)输

入:第一行是 p 的位数 n，第二行是正整数 p，第三行是正整数 q。 输出:两行，分别是 p 除以 q 的商和余数。

```
#include <iostream>
using namespace std;
int p[100];
int n, i, q, rest;
char c;
int main()
{
    cin >> n;
    for (i = 0;
         i < n; i++)
    {
        cin >> c;
        p[i] = c - '0';
    }
    cin >> q;
    rest = (1);
    i = 1;
    while ((2) && i < n)
    {
        rest = rest * 10 + p[i];
        i++;
    }
    if (rest < q)
        cout << 0 << endl;
    else
    {
        cout << (3);
        while (i < n)
        {
            rest = (4);
            i++;
            cout << rest / q;
        }
        cout << endl;
    }
    cout << (5) << endl;
    return 0;
}
```

8.填空题 [NOIP 普及组 2006] 由键盘输入一个奇数 P(P<100,000,000) ，其个位数字不是 5，求一个整数 S，使 P×S=1111...1(在给定的条件下，解 s 必存在) 。要求在屏幕上依次输出以下结果：

(1) S 的全部数字。除最后一行外，每行输出 50 位数字。

(2) 乘积的数字位数。

例 1：输入 P=13，由于 13*8547=111111，则应输出 (1) 8547 ，(2) 6

例 2：输入 P=147，则输出结果应为 (1) 75585789871504157218442932728647014361 3 (2) 42 ，即 等式的右端有 42 个 1。

程序：

```
#include<stdio.h>
int main()
 {
    long p,a,b,c,t,n;
    int bl;
    while(1)
    {
        printf("" 输入 p, 最后一位为 1 或 3 或 7 或 9：\n"");
        scanf(""%ld"",&p);
        if((p%2!=0)&&(p%5!=0)) /* 如果输入的数符合要求，结束循环 */
          ______⑥______;
    }
 a=0; n=0;
    while(a<p);
    {
        a=a*10+1;n++;/*变量 a 存放部分右端项，n 为右端项的位数*/
    }
    t=0;
    do
    {
        b=a/p;
        printf(""%1ld"",b);
        t++;
        if(___________⑦__________)
            printf(""\n"");
         c=_________⑧_________;a=________⑨______;n++;
    }while(c>0);
    printf(""\nn=%ld\n"".__________⑩_______);
}
```

9.单选题 [GESP202406【5 级】] 要实现一个高精度减法函数，则下面代码中加划线应该填写的代码为（ ）。

```
// 假设 a 和 b 均为正数，且 a 表示的数比 b 大
vector<int> minus(vector<int> a, vector<int> b) {
    vector<int> c;
    int len1 = a.size();
    int len2 = b.size();
    int i, t;
    for (i = 0; i < len2; i++) {
        if (a[i] < b[i]) { // 借位
_____________ // 在此处填入代码
            a[i] += 10;
        }
        t = a[i] - b[i];
        c.push_back(t);
    }
    for (; i < len1; i++)
        c.push_back(a[i]);
    len3 = c.size();
    while (c[len3 - 1] == 0) { // 去除前导 0
        c.pop_back();
        len3--;
    }
    return c;
}
```

A. a[i + 1]--;

B. a[i]--;

C. b[i + 1]--;

D. b[i]--;

2. 其他

10.填空题 [NOIP 普及组 2016] （读入整数）请完善下面的程序，使得程序能够读入两个 int 范围内的整数，并将这两个整数分别输出，每行一个。

输入的整数之间和前后只会出现空格或者回车。输入数据保证合法。

例如:

输入:

123　-789

输出:

123

-789

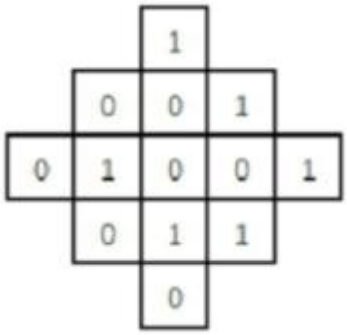
```
#include <iostream>
using namespace std;
int readint() {
    int num = 0; //  存储读取到的整数
    int negative = 0; //  负数标识
    char c;
    c = cin.get(); //  存储当前读取到的字符
    while ((c < '0' || c > '9') && c != '-')
        c = ① ;
    if (c == '-')
        negative = 1;
    else
        ② ;
    c = cin.get();
    while ( ③ ) {
        ④ ;
        c=cin.get();
    }
    if (negative == 1)
        ⑤ ;
    return num;
}
int main() {
    int a, b;
    a = readint();
    b = readint();
    cout << a << endl << b << endl;
    return 0;
}
```

11.填空题 [NOIP 普及组 2010] LZW 编码是一种自适应词典编码。在编码的过程中，开始时只有一部基础构造元素的编 码词典， 如果在编码的过程中遇到一个新的词条，则该词条及一个新的编码会被追加到词典 中，并用于后继信息的编码。 举例说明，考

虑一个待编码的信息串：“xyx yy yy xyx” 。初始词典只有 3 个条目， 第一个为 x，编码为 1；第二个为 y，编码为 2 ；第三个为空格，编码为 3；于是串 “xyx” 的编码为 1-2-1 （其中 - 为编码分隔符） ，加上后面的一个空格就是 1-2-1-3 。但由于有了 一个空格， 我们就知道前面的 “xyx” 是一个单词， 而由于该单词没有在词典中，我们就可以 自适应的把这个词条添加到词典里，编码为 4 ，然后按照新的词典对后继信息进行编码,以 此类推。于是,最后得到编码：1-2-1-3-2-2-3-5-3-4 。现在已知初始词典的 3 个条目如上述，则信息串 “yyxy xx yyxy xyx xx xyx” 的 编码是_____

12.填空题 [NOIP 提高组 2010] LZW 编码是一种自适应词典编码。在编码的过程中，开始时只有一部基础构造元素的编码词典， 如果在编码的过程中
遇到一个新的词条， 则该词条及一个新的编码会被追加到词典 中，并用于后继信息的编码。 举例说明，考虑一个待编码的信息串：“ xyx yy yy xyx ”。初始词典只有 3 个条目， 第一个为 x,编码为 1；第二个为 y，编码为 2；第三个为空格，编码为 3；于是串“ xyx”的 编码为 1-2-1 （其中 -为编码分隔符），加上后面的一个空格就是 1-2-1-3 。但由于有了一个 空格，我们就知道前面的“ xyx”是一个单词，而由于该单词没有在词典中，我们就可以自 适应的把这个词条添加到词典里，编码为 4，然后按照新的词典对后继信息进行编码，以此 类推。于是，最后得到编码：
1-2-1-3-2-2-3-5-3-4 。 我们可以看到，信息被压缩了。压缩好的信息传递到接受方，接收方也只要根据基础 词典就可以完成对该序列的完全恢复。 解码过程是编码过程的逆操作。 现在已知初始词典的 3 个条目如上述，接收端收到的编码信息为
2-2-1-2-3-1-1-3-4-3-1-2-1-3-5-3-6 ，则解码 后的信息串是”
_____ ”。

13.填空题 [NOIP 普及组 2017/NOIP 提高组 2017] 如下图所示，共有 13 个格子。对任何一个格子进行一次操作，会使得它自己以及与它上下左右相邻的格子中的数字改变（由 1 变 0，或由 0 变 1）。现在要使得所有的格子中的数字都变为 0，至少需要_____次操作。



14.单选题 [CSP-S2023] 假设有 n 根柱子，需要按照以下规则依次放置编号为 1,2,3,… 的圆环：每根柱子的底 部固定，顶部可以放入圆环；每次从柱子顶部放入圆环时，需要保证任何两个相邻圆环的编号之和是一个完全平方数。请计算当有 4 根柱子时，最多可以放置（ ）个圆环

A.7

B.9

C.11

D.5

15.单选题 [CSP-J2019]

```
#include<cstdio>
using namespace std;
int n, m;
int a[100], b[100];

int main() {
    scanf(""%d%d"", &n, &m);
    for (int i = 1; i <= n; ++i)
        a[i] = b[i] = 0;
    for (int i = 1; i <= m; ++i) {
        int x, y;
        scanf(""%d%d"", &x, &y);
        if (a[x] < y && b[y] < x) {
            if (a[x] > 0)
                b[a[x]] = 0;
            if (b[y] > 0)
                a[b[y]] = 0;
            a[x] = y;
            b[y] = x;
```

```
            }
        }
        int ans = 0;
        for (int i = 1; i <= n; ++i) {
            if (a[i] == 0)
                ++ans;
            if (b[i] == 0)
                ++ans;
        }
        printf(""%d"", ans);
        return 0;
    }
```

假设输入的 n 和 m 都是正整数，x 和 y 都是在 [1,n]的范围内的整数，完成下面的判断题和单选题：

判断题

当 m>0 时，输出的值一定小于 2n。（ ）

执行完第 27 行的 ++ans 时，ans一定是偶数。（ ）

a[i] 和 b[i] 不可能同时大于 0。（ ）

右程序执行到第 13 行时，x 总是小于 y，那么第 15 行不会被执行。（ ）

选择题

若 m 个 x 两两不同，且 m 个 y 两两不同，则输出的值为（ ）

若 m 个 x 两两不同，且 m 个 y 都相等，则输出的值为（ ）

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 正确

B. 错误

4.

A. 正确

B. 错误

5.

A. 2n−2m

B. 2n+2

C. 2n−2

D. 2n

6.

A. 2n−2

B. 2n

C. 2m

D. 2n−2m

16.填空题 [NOIP 普及组 2017]

```
#include <iostream>
using namespace std;
int main()
{
    int n, m;
    cin >> n >> m;
    int x = 1;
    int y = 1;
    int dx = 1;
    int dy = 1;
    int cnt = 0;
    while(cnt != 2) {
        cnt = 0;
        x = x + dx;
        y = y + dy;
        if(x == 1 || x == n) {
            ++cnt;
            dx = -dx;
        }
        if(y == 1 || y == m) {
            ++cnt;
            dy = -dy;
        }
```

```
    }
    cout << x << "" "" << y << endl;
    return   0 ;
}
```

1）输入：4 3　　输出：

2）输入：2017 1014　　输出：

17.填空题 [NOIP 提高组 2017]

```
#include <iostream>
using namespace std;
int main()
{
    int n, m;
    cin >> n >> m;
    int x = 1;
    int y = 1;
    int dx = 1;
    int dy = 1;
    int cnt = 0;
    while(cnt != 2) {
        cnt = 0;
        x = x + dx;
        y = y + dy;
        if(x == 1 || x == n) {
            ++cnt;
            dx = -dx;
        }
        if(y == 1 || y == m) {
            ++cnt;
            dy = -dy;
        }
    }
    cout << x << "" "" << y << endl;
    return   0 ;
}
```

1）输入：4 3　输出：

2）输入：2017 1014 输出：

3）输入 3：987 321 输出：

18.填空题 [NOIP 提高组 2017]

```
#include <iostream>
using namespace std;
int main() {
    int n, i, j, x, y, nx, ny;
    int a[40][40];
    for (i = 0; i < 40; i++)
        for (j = 0; j < 40; j++)
            a[i][j] = 0;
    cin >> n;
    y = 0;
    x = n - 1;
    n = 2 * n - 1;
    for (i = 1; i <= n * n; i++) {
        a[y][x] = i;
        ny = (y - 1 + n) % n;
        nx = (x + 1) % n;
        if ((y == 0 && x == n - 1) || a[ny][nx] != 0)
            y = y + 1;
        else {
            y = ny; x = nx;
        }
    }
    for (j = 0; j < n; j++)
        cout << a[0][j] << "" "";
    cout << endl;
    return 0;
}
```

输入：3

输出：_____

答案

1.C

2.D

3.A

4.BBADBD

5.0.(384615)

6.

1.正确答案： ans.num[i+j-1]

2.正确答案： ans.num[i]%=10

3.正确答案： a.num[i]+b.num[i]

4.正确答案： ans.num[i] % 2

5.正确答案： ans.len++

6.正确答案： a.len<b.len

7.正确答案： '0' / 48

8.正确答案： times(middle,middle),target

7.

1.正确答案： p[0]

2.正确答案： rest<q / q>rest

3.正确答案： rest/q

4.正确答案： rest%q*10+p[i]

5.正确答案： rest%q"

8.

1. 正确答案： break

2. 正确答案： t%50==0

3. 正确答案： a-p*b

4. 正确答案： c*10+1

5. 正确答案： n-1

9.A

10.

1. 正确答案： cin.get()

2. 正确答案： num = c - '0'

3. 正确答案： c > '0' && c < '9'

4. 正确答案： num = num * 10 + c - '0'

5. 正确答案： num = -num

11.2-2-1-2-3-1-1-3-4-3-1-2-1-3-5-3-6

12.yyxy xx yyxy xyx xx xyx

13.3

14.C

15.ABBBAA

16.

输出 1：1 3

输出 2：2017 1

17.

输出 1：1 3

输出 2：2017 1

输出 3：1 321

18.17 24 1 8 15

GW 信奥 CSP 初赛习题集 7-1

数据结构——线性表

1. 链表概述

1. 一般链表

1.判断题 [GESP202309【6 级】/GESP 样题【7 级】] 有些算法或数据结构在 C/C++ 语言中使用指针实现，一个典型的例子就是链表。因此，链表这一数据结构在 C/C++ 语言中只能使用指针来实现。

2.单选题 [NOIP 普及组 2015/NOIP 提高组 2015] 线性表若采用链表存储结构，要求内存中可用存储单元地址（ ）

A. 必须连续

B. 部分地址必须连续

C. 一定不连续

D. 连续不连续均可

3.单选题 [CSP-J2019/CSP-J2020/NOIP 普及组 2014/NOIP 普及组 2015] 链表不具有的特点是（）

A. 插入删除不需要移动元素

B. 不必事先估计存储空间

C. 所需空间与线性表长度成正比

D. 可随机访问任一元素

4.单选题 [CSP-J2022] 链表和数组的区别包括（ ）。

A. 数组不能排序，链表可以

B. 链表比数组能存储更多的信息

C. 数组大小固定，链表大小可动态调整

D. 以上均正确

5.判断题 [GESP202406【5 级】] 链表的存储空间物理上可以连续，也可以不连续。

6.判断题 [GESP202406【5 级】] 数组和链表都是线性表，链表的优点是插入删除不需要移动元素，并且能随机查找。

7.判断题 [GESP 样题【5 级】] 在任何场景下，链表都是比数组更优秀的数据结构。

2. 循环链表与双向链表

8.判断题 [GESP202406【5 级】] 如果将双向链表的最后一个结点的下一项指针指向第一个结点，第一个结点的前一项指针指向最后一个结点，则该双向链表构成循环链表。

9.判断题 [GESP202406【5 级】] 如果将双向链表的最后一个结点的下一项指针指向第一个结点，第一个结点的前一项指针指向最后一个结，则该双向链表构成循环链表。

10.判断题 [GESP202309【5 级】] 在 C++中，通过恰当的实现，可以将链表首尾相接，形成循环链表。

11.单选题 [GESP 样题【5 级】] 下列关于链表说法，正确的是（ ）。

A. 不能用数组来实现链表。

B. 在链表头部插入元素的时间复杂度是 O(1)。

C. 循环链表使得任意一个结点都可以很方便地访问其前驱与后继。

D. 从双向链表的任意一个节点出发，并不一定能够访问到所有其他节点。

2. 链表的操作

1. 链表查找

12.判断题 [GESP 样题【7 级】/GESP202309【5 级】] 在 C++中，可以使用二分法查找链表中的元素。

13.判断题 [GESP 样题【6 级】] 将 N 个数据按照从小到大顺序存放在一个单向链表中。如果采用二分查找，那么查找的平均时间复杂度是 O(logN)。

14.单选题 [NOIP 提高组 2014] 对长度位 n 的有序单链表，若检索每个元素的概率相等，则顺序检索到表中任一元素的平均检索长度为（).

A.n/2

B.(n+1)/2

C.(n-1)/2

D.n/4

15.判断题 [GESP202406【8 级】] 使用单链表和使用双向链表，查找元素的时间复杂度相同。

16.判断题 [GESP202406【6 级】] n 个节点的双向循环链表，在其中查找某个节点的平均时间复杂度是 O(logn)。

17.单选题 [GESP202309【6 级】] N 个节点的双向循环链，在其中查找某个节点的平均时间复杂度是（ ）。

A.O(1)

B.O(N)

C.0(logN)

D.O(N^2)

18.单选题 [NOIP 普及组 2011] 在含有 n 个元素的双向链表中查询是否存在关键字为 k 的元素，最坏情况下运行的时间复杂度是( )。

A. O(1)

B. O(log n)

C. O(n)

D. O(n log n)

19.单选题 [GESP202406【5 级】] 小杨采用如下双链表结构保存他喜欢的歌曲列表：

```
struct dl_node {
    string song;
    dl_node* next;
    dl_node* prev;
};
```

小杨想在头指针为 head 的双链表中查找他喜欢的某首歌曲，采用如下查询函数，该操作的时间复杂度为（ ）。

```
dl_node* search(dl_node* head, string my_song) {
    dl_node* temp = head;
    while (temp != nullptr) {
        if (temp->song == my_song)
            return temp;
        temp = temp->next;
    }
    return nullptr;
}
```

A.O(1)

B.O(n)

C.O(logn)

D.O(n^2)

2. 链表增删

20.判断题 [GESP202403【5 级】] 单链表和双链表都可以在常数时间内实现在链表头部插入或删除节点的操作。

21.判断题 [GESP202403【6 级】] 删除单向链表中的节点，只需知道待删除节点的地址即可，无需访问前一个节点。

22.单选题 [CSP-J2023] 假设有一个链表的节点定义如下 struct Node { int data; Node* next;}
现在有一个指向链表头部的指针：Node* head。如果想要在链表中插入一个新节点，其成员 data 的值为 42，并使新节点成为链表的第一个节点，下面哪个操作是正确的？
A. Node* newNode = new Node; newNode->data = 42; newNode->next = head; head = newNode;
B. Node* newNode = new Node; head->data = 42; newNode->next = head; head = newNode;
C. Node* newNode = new Node; newNode->data = 42; head->next = newNode;
D. Node* newNode = new Node; newNode->data = 42; newNode->next = head;

23.单选题 [NOIP 提高组 2014] 有以下结构体说明和变量定义，如图所示，指针 p、q、r 分别指向一个链表中的三个续 结点。

```
struct node {
    int data;
    node *next;
}*p,*q,*r;
```

现要将 q 和 r 所指的结点先后位置交换，同时要保持链表的连续，以下程序段中错误的是（ ）。
A.q->next = r->next; p-> next = r; r->next = q;
B.p->next = r; q->next = r->next; r->next = q;

C.q->next = r->next; r->next = q; p->next = r;

D.r->next = q; q->next = r->next; p->next = r;

24.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 双向链表中有两个指针域 llink 和 rlink ，分别指向该结点的前驱及后继。 设 p 指向 链表中的一个结点，它的左右结点均非空。现要求删除结点 p ，则下面语句序列中错误的是 （ ）。

A. p->rlink->llink = p->rlink; p->llink->rlink = p->llink; delete p;

B. p->llink->rlink = p->rlink; p->rlink->llink = p->llink; delete p;

C. p->rlink->llink = p->llink; p->rlink->llink->rlink = p->rlink; delete p;

D. p->llink->rlink = p->rlink; p->llink->rlink->llink = p->llink; delete p;

25.单选题 [NOIP 提高组 2015] 双向链表中有两个指针域，llink 和 rlink，分别指回前驱及后继，设 p 指向链表中的 一个结点，q 指向一待插入结点，现要求在 p 前插入 q，则正确的插入为（ ）。

A. p->llink = q; q->rlink = p; p->llink->rlink = q;q->llink = p->llink;

B. q->llink = p->llink; p->llink->rlink = q; q->rlink = p;p->llink = q->rlink;

C. q->rlink = p; p->rlink = q;p->llink->rlink = q; q->rlink = p;

D. p->llink->rlink = q; q->rlink = p;q->llink = p->llink; p->llink = q;

26.不定项选择题 [NOIP 提高组 2009] 在带尾指针（链表指针 clist 指向尾结点）的非空循环单链表中每个结点都以 next 字段的指针指向下一个节点。假定其中已经有 2 个以上的结点。下面哪些说法是正确的：

A. 如果 p 指向一个待插入的新结点，在头部插入一个元素的语句序列为：p->next = clist->next; clist->next = p;

B. 如果 p 指向一个待插入的新结点，在尾部插入一个元素的语句序列为：p->next = clist； clist->next = p;

C. 在头部删除一个结点的语句序列为： p = clist->next; clist->next = clist->next->next; deletep;

D. 在尾部删除一个结点的语句序列为。p = clist; clist = clist ->next; delete p;

27.单选题 [CSP-J2022] 以下哪组操作能完成在双向循环链表结点 p 之后插入结点 s 的效果（其中，next 域为结点的直接后继，prev 域为结点的直接前驱）：（ ）。

A. p->next->prev=s; s->prev=p; p->next=s; s->next=p->next;

B. p->next->prev=s; p->next=s; s->prev=p; s->next=p->next;

C. s->prev=p; s->next=p->next; p->next=s; p->next->prev=s;

D. s->next=p->next; p->next->prev=s; s->prev=p; p->next=s;

28.单选题 [GESP202406【5 级】] 小杨采用如下双链表结构保存他喜欢的歌曲列表：

struct dl_node {

string song;

dl_node* next;

dl_node* prev;

};

小杨想在如上题所述的双向链表中加入一首新歌曲。为了能快速找到该歌曲，他将其作为链表的第一首歌曲，则下面横线上应填写的代码为（ ）。

```
void insert(dl_node* head, string my_song) {
    p = new dl_node;
    p->song = my_song;
    p->prev = nullptr;
    p->next = head;
    if (head != nullptr) {
_____________________________________ // 在此处填入代码
    }
    head = p;
}
```

A. head->next->prev = p;

B. head->next = p;

C. head->prev = p;

D. 触发异常，不能对空指针进行操作。

3. 其他

29.判断题 [GESP202312【5 级】] 同样的整数序列分别保存在单链表和双向链中，这两种链表上的简单冒泡排序的复杂度相同。

30.判断题 [GESP202312【6 级】] 同样的整数序列分别保存在单链表和双向链中，这两种链表上的简单冒泡排序的复杂度相同。

答案

1.F

2.D

3.D

4.C

5.T

6.F

7.F

8.T

9.T

10.T

11.B

12.F

13.F

14.B

15.T

16.F

17.B

18.C

19.B

20.T

21.F

22.A

23.D

24.A

25.D

26.AC

27.D

28.C

29.T

30.T

GW 信奥 CSP 初赛习题集 7-2

数据结构——带限制的线性表

1. 栈

1. 栈的基本概念

1.单选题 [GESP202406【6 级】] 在栈数据结构中，元素的添加和删除是按照什么原则进行的？

A. 先进先出

B. 先进后出

C. 最小值先出

D. 随机顺序

2.判断题 [GESP202403【6 级】] 栈的基本操作包括入栈（push）和出栈（pop）。

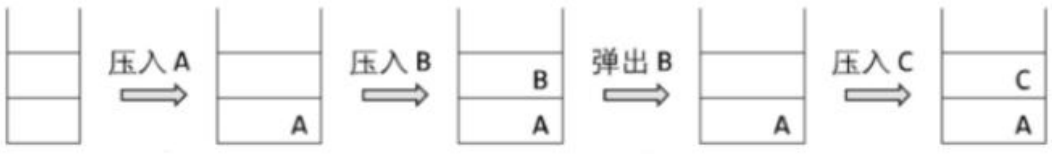
3.单选题 [CSP-J2020/NOIP 普及组 2018/NOIP 普及组 2013] 下图中所使用的数据结构是（ ）。

A. 栈

B. 队列

C. 二叉树

D. 哈希表



2. 出入栈问题

4.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 某个车站呈狭长形，宽度只能容下一台车，并且只有一个出入口。已知某时刻该车站状态为空，从这一时刻开始的出入记录为：“进，出，进，进，出，进，进，进，出，出，进，出”。假设车辆入站的顺序为 1，2，3，……，则车辆出站的顺序为（ ）。

A. 1, 2, 3, 4, 5

B. 1, 2, 4, 5, 7

C. 1, 3, 5, 4, 6

D. 1, 3, 5, 6, 7

E. 1, 3, 6, 5, 7

5.不定项选择题 [NOIP 提高组 2005/NOIP 普及组 2005] 设栈 S 的初始状态为空，元素 a, b, c, d, e, f, g 依次入栈，以下出栈序列不可能出现的有（ ）。

A. a, b, c, e, d, f, g

B. b, c, a, f, e, g, d

C. a, e, c, b, d, f, g

D. d, c, f, e, b, a, g

E. g, e, f, d, c, b, a

6.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 某个车站呈狭长形，宽度只能容下一台车，并且只有一个出入口。已知某时刻该车站状态为空，从这一时刻开始的出入记录为：“进，出，进，进，进，出，出，进，进，进，出,出”。假设车辆入站的顺序为 1，2，3，……，则车辆出站的顺序为（ ）。

A. 1, 2, 3, 4, 5

B. 1, 2, 4, 5, 7

C. 1, 4, 3, 7, 6

D. 1, 4, 3, 7, 2

7.单选题 [NOIP 普及组 2006/NOIP 提高组 2006] 设栈 S 的初始状态为空，元素 a, b, c, d, e 依次入栈，以下出栈序列不可能出现的有（ ）。

A. a, b, c, e, d

B. b, c, a, e, d

C. a, e, c, b, d

D. d, c, e, b, a

8.单选题 [NOIP 普及组 2007/NOIP 提高组 2007] 地面上有标号为 A、B、C 的 3 根细柱，在 A 柱上放有 10 个直径相同中间有孔的圆盘，从上到下依次编号为 1，2，3，……，将 A 柱上的部分盘子经过 B 柱移入 C 柱，也可以在 B 柱上暂存。如果 B 柱上的操作记录为：“进，进，出，进，进，出，出，进，进，出，进，出，出”。那么，在 C 柱上，从下到上的盘子的编号为（ ）。

A. 2 4 3 6 5 7

B. 2 4 1 2 5 7

C. 2 4 3 1 7 6

D. 2 4 3 6 7 5

9.单选题 [NOIP 普及组 2009] 有六个元素 FEDCBA 从左至右依次顺序进栈，在进栈过程中会有元素被弹出栈。问下列哪一个不可能是合法的出栈序列？

A. EDCFAB

B. DECABF

C. CDFEBA

D. BCDAEF

10.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 对于入栈顺序为 a, b, c, d, e, f, g 的序列，下列()不可能是合法的出栈序列。

A. a,b,c,d,e,f,g

B. a,d,c,b,e,g,f

C. a,d,b,c,g,f,e

D. g,f,e,d,c,b,a

11.单选题 [CSP-J2021] 对于入栈顺序为 a,b,c,d,e 的序列，下列（ ）不是合法的出栈序列。

A.a,b,c,d,e

B.e,d,c,b,a

C.b,a,c,d,e

D.c,d,a,e,b

12.单选题 [CSP-J2022] 有 6 个元素，按照 6,5,4,3,2,1 的顺序进入栈 S，请问下列哪个出栈序列是非法的（ ）。

A.5,4,3,6,1,2

B.4,5,3,1,2,6

C.3,4,6,5,2,1

D.2,3,4,1,5,6

13.单选题 [CSP-S2022] 若元素 a、b、c、d、e、f 依次进栈，允许进栈、退栈操作交替进行，但不允许连续三次退栈操作，则不可能得到的出栈序列是（ ）。

A. dcebfa

B. cbdaef

C. bcaefd

D. afedcb

14.单选题 [NOIP 普及组 2015/NOIP 提高组 2015/CSP-S2020] 今有一空栈 S，对下列待进栈的数据元素序列 a,b,c,d,e,f 依次进行进栈，进栈，出栈，进栈，进 栈， 出栈的操作，则此操作完成后，栈 S 的栈顶元素为（ ）。

A. f

B. c

C. a

D. b

15.单选题 [GESP202403【6 级】] 给定一个空栈，执行以下操作序列：

操作序列： push(1), push(2), push(3), pop(), pop(), push(4), push(5), pop()

最终栈中的元素是（ ）。

A. 1, 2

B. 1, 4, 5

C. 1, 2, 5

D. 1, 4

16.不定项选择题 [NOIP 提高组 2010/NOIP 普及组 2010] 元素 R1、R2 、R3 、R4、R5 入栈的顺序为 R1 、R2、R3 、R4 、R5。如果第一个出栈的 是 R3，那么第五个出栈的可能是（ ）。

A. R1

B. R2

C. R4

D. R5

17.单选题 [NOIP 普及组 2008] 设栈 S 的初始状态为空，元素 a，b，c，d，e，f 依次入栈 S，出栈的序列为 b，d，f，e，c，a，则栈 S 的容量至少应该是（ ）。

A. 6

B. 5

C. 4

D. 3

18.单选题 [NOIP 提高组 2008] 设栈 S 的初始状态为空，元素 a，b，c，d，e，f 依次入栈 S，出栈的序列为 b，d，c，f，e，a，则栈 S 的容量至少应该是（ ）。

A. 6

B. 5

C. 4

D. 3

E. 2

19.不定项选择题 [NOIP 提高组 2012NOIP 普及组 2012] 如果一个栈初始时为空，且当前栈中的元素从栈顶到栈底依次为 a，b，c（如右图所示），另有元素 d 已经出栈，则可能的入栈顺序是（ ）。A. a, b, c, d

B. b, a, c, d

C. a, c, b, d

D. d, a, b, c

20.单选题 [NOIP 普及组 2017/GESP202406【6 级】] 向一个栈顶指针为 hs 的链式栈中插入一个指针 s 指向的结点时，应执行（ ）。

A. hs->next=s;

B. s->next=hs;hs=s;

C. s->next=hs->next;hs->next=s;

D. s->next=hs;hs=hs->next;

21.单选题 [GESP202403【6 级】] 在队列中，元素的添加和删除是按照（ ）原则进行的。

A. 先进先出

B. 先进后出

C. 最小值先出

D. 随机进出

二、队列

22.单选题 [NOIP 普及组 2012]（ ）是一种先进先出的线性表。

A. 栈

B. 队列

C. 哈希表（散列表）

D. 二叉树

23.单选题 [CSP-J2022] 对假设栈 S 和队列 Q 的初始状态为空。存在 e1~e6 六个互不相同的数据，每个数据按照进栈 S、出栈 S、进队列 Q、出队列 Q 的顺序操作，不同数据间的操作可能会交错。已知栈 S 中依次有数据 e1、e2、e3、e4、e5 和 e6 进栈，队列 Q 依次有数据 e2、e4、e3、e6、e5 和 e1 出队列。则栈 S 的容量至少是（ ）个数据。

A.2

B.3

C.4

D.6

答案

1.B

2.T

3.A

4.E

5.CE

6.C

7.C

8.D

9.C

10.C

11.D

12.C

13.D

14.B

15.D

16.ACD

17.C

18.D

19.AD

20.B

21.A

22.B

23.B

GW 信奥 CSP 初赛习题集 7-3

数据结构——树

1. 树的存储

1.判断题 [GESP202406【6 级】] 完全二叉树可以用数组存储数据。

2.单选题 [GESP202312【6 级】] 下面有关树的存储，错误的是（ ）.

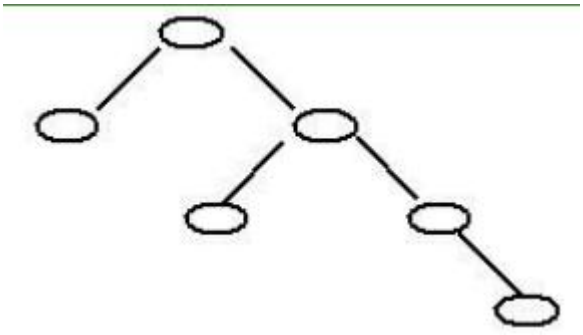
A. 完全二叉树可以用 list 存储

B. 一般二叉树都可以用 list 存储，空子树位置可以用 None 表示

C. 满二叉树可以用 list 存储

D. 树数据结构，都可以用 list 存储

3.单选题 [NOIP 提高组 2016] 一棵二叉树如右图所示 ，若采用二叉树链表存储该二叉 树（各个结点包括结点的数据、左孩子指针、右孩子指 针）。如果没有左孩子或者右孩子 ，则对应的为空指针。 那么该链表中空指针的数目为 （ ）。



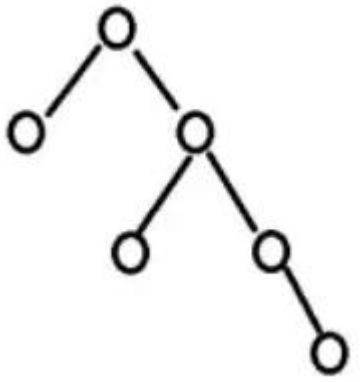
A. 6

B. 7

C. 12

D. 14

4.单选题 [CSP-J2019] 一棵二叉树如右图所示，若采用顺序存储结构，即用一维数组元素存储该二叉树中的结点（根结点的下标为 1，若某结点的下标为 i，则其左孩子位于下标 2i 处、右孩子位于下标 2i+1 处），则该数组的最大下标至少为（ ）。



5.单选题 [GESP202312【6 级】] 构造二叉树 [1,2,3,null,4] （ ）。

A. 1(2()(4))(3)

B. 1(2(3)())(4)

C. (1,2(3),(4))

D. (1,(2)(3),(4))

6.单选题 [GESP202312【6 级】] 有关下面 C++代码的说法，错误的是( )。

```
struct BiNode {
    char data;
    BiNode* lchild, *rchild;
};

class BiTree {
private:
    BiNode* Creat();
    void Release(BiNode* t);
    BiNode* root;

public:
    BiTree() {
        root = Creat();
    }

    ~BiTree() {
```

```
18              Release(root);
19          }
20  };
```

A. 上列 C++代码适用于构造各种二叉树

B. 代码 struct BiNode 用于构造二叉树的节点

C. 代码 BiTree(){root=Creat();} 用于构造二叉树

D. 析构函数不可以省略

2. 树的性质

7.单选题 [NOIP 普及组 2008] 设 T 是一棵有 n 个顶点的树，下列说法不正确的是（ ）。

A. T 有 n 条边

B. T 是连通的

C. T 是无环的

D. T 有 n-1 条边

8.不定项选择题 [NOIP 提高组 2008] 设 T 是一棵有 n 个顶点的树，下列说法正确的是（ ）。

A. T 是连通的、无环的

B. T 是连通的，有 n-1 条边

C. T 是无环的，有 n-1 条边

D. 以上都不对

9.不定项选择题 [NOIP 提高组 2018] 下列说法中，是树的性质的有（ ）。

A. 无环

B. 任意两个结点之间有且只有一条简单路径

C. 有且只有一个简单环

D. 边的数目恰是顶点数目减 1

10.判断题 [GESP202406【7 级】] 一颗 N 层的二叉树，至少有 2^{N-1} 个节点。

11.判断题 [GESP 样题【6 级】] 任意二叉树都至少有一个结点的度是 2。

12.判断题 [GESP202403【6 级】] 完全二叉树的任意一层都可以不满。

13.判断题 [GESP202312【7 级】] 假设一棵完全二叉树共有 N 个节点，则树的深度为 logN+1。

14.单选题 [NOIP 普及组 2006] 高度为 n 的均衡的二叉树是指：如果去掉叶结点及相应的树枝，它应该是高度为 n-1 的满二叉树。在这里，树高等于叶结点的最大深度，根结点的深度为 0，如果某个均衡的二叉树共有 2381 个结点，则该树的树高为（ ）。

A. 10

B. 11

C. 12

D. 13

15.单选题 [NOIP 提高组 2006] 高度为 n 的均衡的二叉树是指：如果去掉叶结点及相应的树枝，它应该是高度为 n-1 的满二叉树。在这里，树高等于叶结点的最大深度，根结点的深度为 0，如果某个均衡的二叉树共有 2381 个结点，则该树的树高为（ ）。

A. 10

B. 11

C. 12

D. 13

E. $2^{10} - 1$

16.不定项选择题 [NOIP 提高组 2011/NOIP 普及组 2011] 如果根节点的深度记为 1，则一棵恰有 2011 个叶结点的二叉树的深度可能是 （ ）.

A. 10

B. 11

C. 12

D. 2011

17.单选题 [NOIP 普及组 2015/NOIP 提高组 2015/CSP-J2020] 如果根的高度为 1，具有 61 个结点的完全二叉树的高度为（ ）。

A. 5

B. 6

C. 7

D. 8

18.单选题 [CSP-S2021] 令根结点的高度为 1，则一棵含有 2021 个结点的二叉树的高度至少为（ ）。

A. 10

B. 11

C. 12

D. 2021

19.判断题 [GESP202403【7 级】] 一棵有 N 个节点的完全二叉树，则树的深度为 $\lfloor \log(N) \rfloor + 1$

20.单选题 [CSP-J2023/GESP202312【7 级】] 根节点的高度为 1，一棵拥有 2023 个节点的三叉树高度至少为（）。

A. 6

B. 7

C. 8

D. 9

21.单选题 [GESP 样题【6 级】] 在具有 2N 个结点的完全二叉树中，叶子结点个数为（ ）。

A．N/2

B．N − 1

C．N

D．N+ 1

22.单选题 [NOIP 普及组 2005] 完全二叉树的结点个数为 11，则它的叶结点个数为（ ）。

A. 4

B. 3

C. 5

D. 2

E. 6

23.单选题 [NOIP 提高组 2005] 完全二叉树的结点个数为 4 * N + 3，则它的叶结点个数为（ ）。

A. 2 * N

B. 2 * N - 1

C. 2 * N + 1

D. 2 * N - 2

E. 2 * N + 2

24.单选题 [NOIP 普及组 2008/NOIP 提高组 2008] 完全二叉树共有 2*N-1 个结点，则它的叶节点数是（ ）。

A. N-1

B. N

C. 2*N

D. $2^{N}-1$

25.不定项选择题 [NOIP 提高组 2012] 一棵二叉树一共有 19 个节点，其叶子节点可能有（　　）个。

A. 1

B. 9

C. 10

D. 15

26.填空题 [NOIP 普及组 2015] 一棵结点数为 2015 的二叉树最多有（ ）个叶子结点。

27.填空题 [NOIP 普及组 2016] 约定二叉树的根节点高度为 1。一棵结点数为 2016 的二叉树最少有(①)个叶子结点;一棵结点数为 2016 的二叉树最小的高度值是(②)。

28.单选题 [NOIP 普及组 2009] 一个包含 n 个分支结点（非叶结点)的非空二叉树，它的叶结点数目最多为:

A. 2n + 1

B. 2n-1

C. n-1

D. n+1

29.单选题 [NOIP 提高组 2009] 一个包含 n 个分支结点（非叶结点)的非空满 k 叉树，k>=1，它的叶结点数目为:

A. nk + 1

B. nk-1

C. (k+1)n-1

D. (k-1)n+1

30.单选题 [GESP202406【8 级】/NOIP 普及组 2013] 已知一棵二叉树有 10 个节点，则其中最多有（ ）个节点有 2 个子节点。

A. 4

B. 5

C. 6

D. 3

31.单选题 [NOIP 提高组 2013] 已知一棵二叉树有 2013 个节点，则其中至多有（ ）个节点有 2 个子节点。

A. 1006

B. 1007

C. 1023

D. 1024

32.不定项选择题 [NOIP 提高组 2018] 2-3 树是一种特殊的树，它满足两个条件：

（1）每个内部结点有两个或三个子结点；

（2）所有的叶结点到根的路径长度相同。

如果一棵 2-3 树有 10 个叶结点，那么它可能有（ ）个非叶结点。

A. 5

B. 6

C. 7

D. 8

33.单选题 [NOIP 普及组 2004/NOIP 提高组 2004] 满二叉树的叶结点个数为 N，则它的结点总数为（ ）。

A. N

B. 2 * N

C. 2 * N − 1

D. 2 * N + 1

E. 2^N − 1

34.单选题 [GESP202403【6 级】] 一个有 124 个叶子节点的完全二叉树，最多有（ ）个结点。

A. 247

B. 248

C. 249

D. 250

35.单选题 [GESP202406【6 级】/NOIP 普及组 2014] 一棵 5 层的满二叉树中节点数为（ ）。

A. 31

B. 32

C. 33

D. 16

36.单选题 [GESP 样题【7 级】] 深度为 4 的完全二叉树，结点总数最少有多少个？（ ）

A. 5

B. 6

C. 7

D. 8

37.单选题 [NOIP 普及组 2010] 如果树根算第 1 层，那么一棵 n 层的二叉树最多有（ ）个结点

A. 2^n-1

B. 2^n

C. 2n +1

D. 2^{n+1}

38.单选题 [NOIP 普及组 2018/NOIP 提高组 2018] 根节点深度为 0，一棵深度为 h 的满 k（k>1）叉树，即除最后一层无任何子节点外，每一层上的所有结点都有 k 个子结点的树，共有（ ）个结点。

A. $(k^{h+1} - 1) / (k - 1)$

B. k^{h-1}

C. k^{h}

D. $k^{h-1} / (k - 1)$

39.单选题 [CSP-J2022] 一棵有 n 个结点的完全二叉树用数组进行存储与表示，已知根结点存储在数组的第 1 个位置。若存储在数组第 9 个位置的结点存在兄弟结点和两个子结点，则它的兄弟结点和右子结点的位置分别是（ ）。

A. 8、18

B.10、18

C.8、19

D.10、19

40.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 完全二叉树的顺序存储方案，是指将完全二叉树的结点从上至下、从左至右依次存放 到一个顺序结构的数组中。假定根结点存放在数组的 1 号位置，则第 k 号结点的父结点如 果存在的话，应当存放在数组的（ ）号位置。

A. 2k

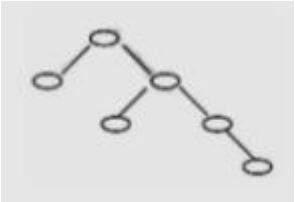
B. 2k+1

C. $\lfloor k/2 \rfloor$

D. $\lfloor (k+1)/2 \rfloor$

41.单选题 [NOIP 普及组 2016] 一棵二叉树如右图所示，若采用顺序存储结构，即用一 维数组元素存储该二 叉树中的结点 （根结点的下标为 1，若某结点的下标为 i，

则其左孩子位于下标 2i 处、 右孩子位于下标 (2i+1)处），则图中所有结点的最大下标为（ ）。



A. 6

B. 10

C. 12

D. 15

42.单选题 [CSP-J2021] 如果一棵二叉树只有根结点，那么这棵二叉树高度为 1。请问高度为 5 的完全二叉树有 （ ）种不同的形态？

A. 16

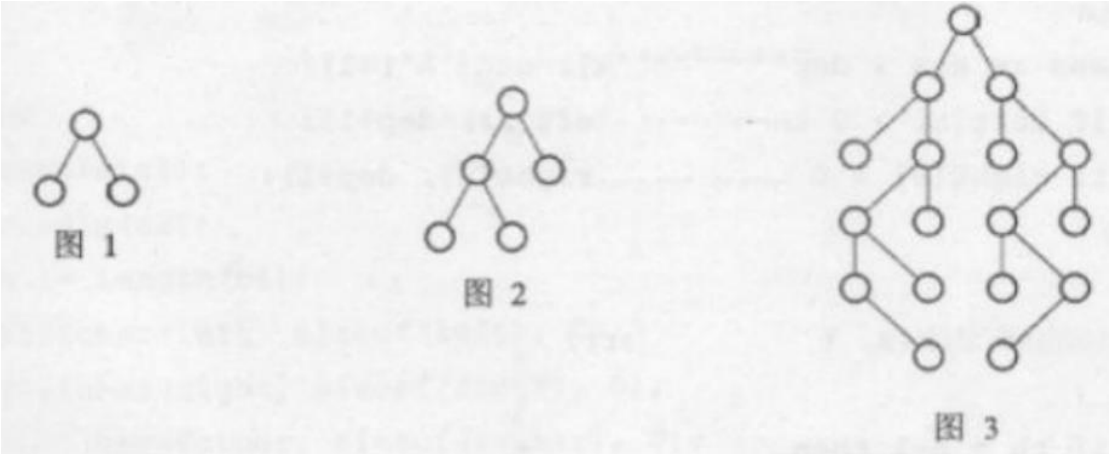
B. 15

C. 17

D. 32

43.填空题 [NOIP 提高组 2015] 结点数为 5 的不同形态的二叉树一共有_____种。（结点数为 2 的二叉树一共有 2 种：一种是根结点和左儿子，另一种是根结点和右儿子。）

44.填空题 [NOIP 提高组 2012] 对于一棵二叉树，独立集是指两两互不相邻的节点构成的集合。例如，图 1 有 5 个不同的独立集（1 个双点集合、3 个单点集合、1 个空集），图 2 有 14 个不同的独立集。那么，图 3 有____个不同的独立集。



3. 树的遍历

45.单选题 [GESP202406【6 级】/NOIP 普及组 2013] 二叉树的（ ）第一个访问的节点是根节点。

A. 先序遍历

B. 中序遍历

C. 后序遍历

D. 以上都是

46.单选题 [NOIP 提高组 2013] 二叉查找树具有如下性质：每个节点的值都大于其左子树上所有节点的值、小于其右子树 上所有节点的值。那么，二叉查找树的（ ）是一个有序序列。

A. 先序遍历

B. 先序遍历

C. 后序遍历

D. 宽度优先遍历

47.单选题 [GESP202312【8 级】] 对有 n 个元素的二叉排序树进行中序遍历，其时间复杂度是（ ）。

A.O(1)

B.O(log(n))

C.O(n)

D.O(n^2)

48.单选题 [GESP202406【7 级】] 关于图的深度优先搜索和广度优先搜索，下列说法错误的是（ ）。

A. 二叉树也是一种图。

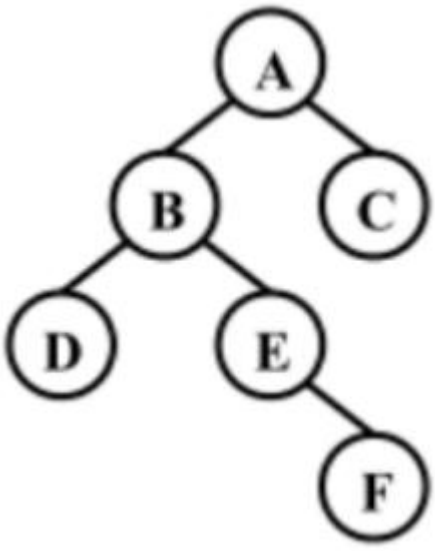
B. 二叉树的前序遍历和后序遍历都是深度优先搜索的一种。

C. 深度优先搜索可以从任意根节点开始。

D. 二叉树的后序遍历也是广度优先搜索的一种。

49.判断题 [GESP 样题【6 级】] 深度优先遍历算法的时间复杂度为 O(NlogN)，其中 N 为树的节点数。

50.单选题 [NOIP 提高组 2011] 下图是一棵二叉树，它的先序遍历是（ ）。



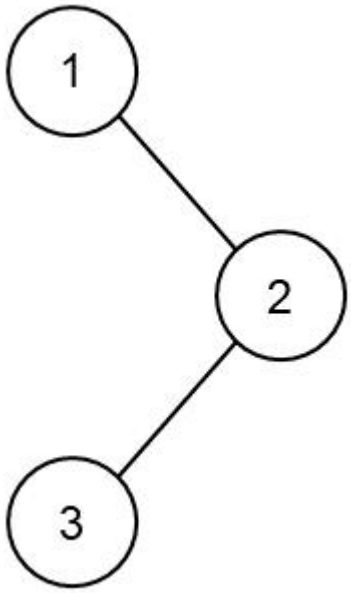
A. ABDEFC

B. DBEFAC

C. DFEBCA

D. ABCDEF

51.单选题 [GESP202406【7 级】] 对于如下图的二叉树，说法正确的是（ ）。



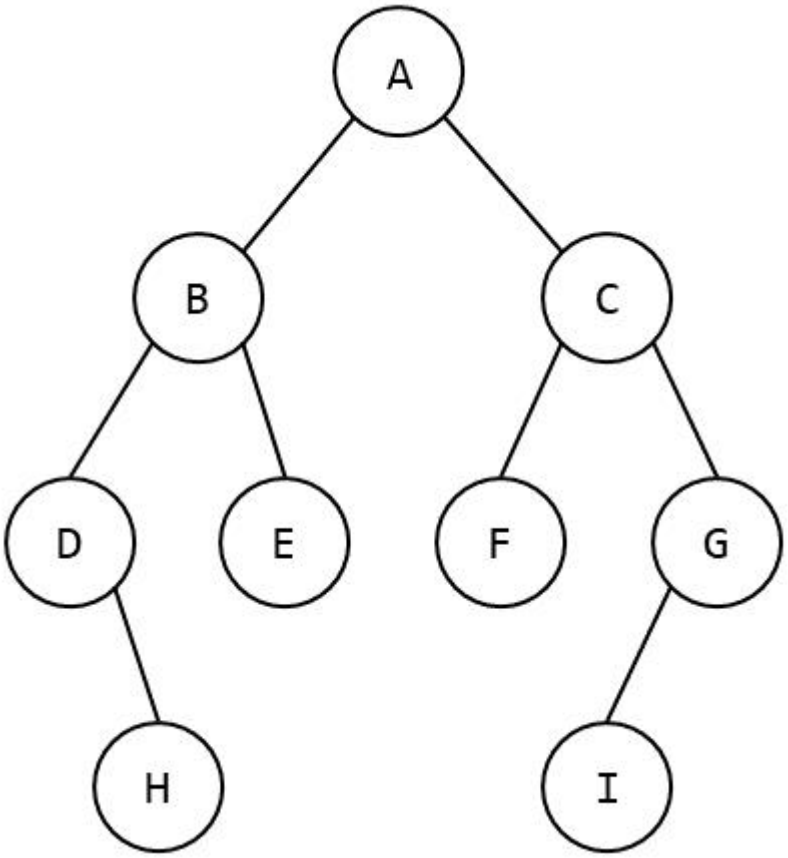
A. 先序遍历是 132 。

B. 中序遍历是 123 。

C. 后序遍历是 312 。

D. 先序遍历和后序遍历正好是相反的。

52.单选题 [GESP202406【7 级】] 对于如下二叉树，下面访问顺序说法错误的是（ ）。



A. HDEBFIGCA 不是它的后序遍历序列

B. ABCDEFGHI 是它的广度优先遍历序列

C. ABDHECFGI 是它的深度优先遍历序列

D. ABDHECFGI 是它的先序遍历序列

53.单选题 [NOIP 普及组 2015] 前序遍历序列与中序遍历序列相同的二叉树为（ ）。

A. 根结点无左子树的二叉树

B. 根结点无右子树的二叉树

C. 只有根结点的二叉树或非叶子结点只有左子树的二叉树

D. 只有根结点的二叉树或非叶子结点只有右子树的二叉树

54.单选题 [NOIP 提高组 2015] 前序遍历序列与后序遍历序列相同的二叉树为( )。

A. 非叶子结点只有左子树的二叉树

B. 只有根结点的二叉树

C. 根结点无右子树的二叉树

D. 非叶子结点只有右子树的二叉树

55.单选题 [CSP-S2021] 前序遍历和中序遍历相同的二叉树为且仅为（ ）。

A. 只有 1 个点的二叉树

B. 根结点没有左子树的二叉树

C. 非叶子结点只有左子树的二叉树

D. 非叶子结点只有右子树的二叉树

56.单选题 [NOIP 提高组 2004/NOIP 普及组 2004] 二叉树 T，已知其前序遍历序列为 1 2 4 3 5 7 6，中序遍历序列为 4 2 1 5 7 3 6，则其后序遍历序列为（ ）。

A. 4 2 5 7 6 3 1

B. 4 2 7 5 6 3 1

C. 4 2 7 5 3 6 1

D. 4 7 2 3 5 6 1

E. 4 5 2 6 3 7 1

57.单选题 [NOIP 普及组 2007] 已知 7 个结点的二叉树的先根遍历是 1 2 4 5 6 3 7（数字为结点的编号，以下同），中根遍历是 4 2 6 5 1 7 3，则该二叉树的后根遍历是（ ）

A. 4 6 5 2 7 3 1

B. 4 6 5 2 1 3 7

C. 4 2 3 1 5 4 7

D. 4 6 5 3 1 7 2

58.单选题 [NOIP 普及组 2008] 二叉树 T，已知其先根遍历是 1 2 4 3 5 7 6（数字为结点的编号，以下同），中根遍历是 2 4 1 5 7 3 6，则该二叉树的后根遍历是（ ）。

A. 4 2 5 7 6 3 1

B. 4 2 7 5 6 3 1

C. 7 4 2 5 6 3 1

D. 4 2 7 6 5 3 1

59.单选题 [CSP-J2023] 给定一棵二叉树，其前序遍历结果为：ABDECFG,中序遍历结果为：DEBACFG。请问这棵树的正确后序遍历结果是什么？

A. EDBGFCA

B. EDGBFCA

C. DEBGFCA

D. DBEGFCA

60.单选题 [GESP202403【6 级】] 若一棵二叉树的先序遍历为：A, B, D, E, C, F、中序遍历为：D, B, E, A, F, C，它的后序遍历为（ ）。

A. D, E, B, F, C, A

B. E, D, B, F, C, A

C. D, E, B, C, F, A

D. E, D, B, C, F, A

61.不定项选择题 [NOIP 提高组 2006/NOIP 普及组 2006] 已知 6 个结点的二叉树的先根遍历是 1 2 3 4 5 6（数字为结点的编号，以下同），后根遍历是 3 2 5 6 4 1，则该二叉树的可能的中根遍历是（ ）

A. 3 2 1 4 6 5

B. 3 2 1 5 4 6

C. 2 3 1 5 4 6

D. 2 3 1 4 6 5

62.不定项选择题 [NOIP 提高组 2007] 已知 7 个结点的二叉树的先根遍历是 1 2 4 5 6 3 7（数字为结点的编号，以下同），后根遍历是 4 6 5 2 7 3 1，则该二叉树的可能的中根遍历是（ ）

A. 4 2 6 5 1 7 3

B. 4 2 5 6 1 3 7

C. 4 2 3 1 5 4 7

D. 4 2 5 6 1 7 3

63.不定项选择题 [NOIP 提高组 2008] 二叉树 T，已知其先根遍历是 1 2 4 3 5 7 6（数字为结点的编号，以下同），后根遍历是 4 2 7 5 6 3 1，则该二叉树的可能的中根遍历是（ ）。

A. 4 2 1 7 5 3 6

B. 2 4 1 7 5 3 6

C. 4 2 1 7 5 6 3

D. 2 4 1 5 7 3 6

64.单选题 [CSP-J2019] 假设一棵二叉树的后序遍历序列为 DGJHEBIFCA，中序遍历序列为 DBGEHJACIF，则其前序遍历序列为（ ）。

A.ABCDEFGHIJ

B.ABDEGHJCFI

C.ABDEGJHCFI

D.ABDEGHJFIC

65.单选题 [NOIP 普及组 2012] 如果一棵二叉树的中序遍历是 BAC，那么它的先序遍历不可能是( )。

A. ABC

B. CBA

C. ACB

D. BAC

66.单选题 [GESP202312【7 级】] 某二叉树 T 的先序遍历序列为：{A B D F C E G H}，中序遍历序列为：{B F D A G E H C}，则下列说法中正确的是( )。

A. T 的度为 1

B. T 的高为 4

C. T 有 4 个叶节点

D. 以上说法都不对

67.单选题 [GESP202403【7 级】] 已知一颗二叉树的中序遍历序列为：{C F B A E D G}，后序遍历序列为：{F C B E G D A}，则下列说法中正确的是( )。

A. 该树是平衡二叉树。

B. 该树的高为 4。

C. 该树有 4 个叶节点。

D. 以上说法都不对。

68.单选题 [CSP-S2022] 一个深度为 5（根结点深度为 1）的完全 3 叉树，按前序遍历的顺序给结点从 1 开始编号，则第 100 号结点的父结点是第（ ）号。

A. 95

B. 96

C. 97

D. 98

69.单选题 [NOIP 普及组 2005] 二叉树 T 的宽度优先遍历序列为 A B C D E F G H I，已知 A 是 C 的父结点，D 是 G 的父结点，F 是 I 的父结点，树中所有结点的最大深度为 3（根结点深度设为 0），可知 F 的父结点是（ ）。

A. 无法确定

B. B

C. C

D. D

E. E

70.不定项选择题 [NOIP 提高组 2005] 二叉树 T 的宽度优先遍历序列为 A B C D E F G H I，已知 A 是 C 的父结点，D 是 G 的父结点，F 是 I 的父结点，树中所有结点的最大深度为 3（根结点深度设为 0），可知 E 的父结点可能是（ ）。

A. A

B. B

C. C

D. D

E. F

71.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 一棵二叉树的前序遍历序列是 ABCDEFG，后序遍历序列是 CBFEGDA，则根结点的左子树的结点个数可能是（ ）。

A. 2

B. 3

C. 4

D. 5

72.单选题 [GESP202403【6 级】] 阅读以下广度优先搜索的代码：

```
void bfs(TreeNode* root) {
    if (root == NULL) {
        return;
    }
    queue<TreeNode*> q;
    q.push(root);
    while (!q.empty()) {
        TreeNode* current = q.front();
        q.pop();
        cout << current->val << "" "";
        if (current->left) {
            q.push(current->left);
        }
        if (current->right) {
            q.push(current->right);
        }
    }
}
```

使用以上算法遍历以下这棵树，可能的输出是（ ）。

```
    1
   / \
```

A. 1 2 8 9 4 5 3 6 7 10 11

B. 1 2 3 4 5 6 7 8 9 10 11

C. 1 2 3 8 9 6 4 5 7 10 11

D. 1 2 3 8 9 4 5 6 7 10 11

73.单选题 [GESP202312【6 级】] 有关下面 C++代码不正确的说法是（ ）。

```
int Depth(BiTree T)
{
    if (T == NULL)
    {
        return 0;
    }
    else
    {
        int m = Depth(T->lchild);
        int n = Depth(T->rchild);
        if (m > n)
        {
            return m + 1;
        }
        else
        {
            return n + 1;
        }
    }
}
```

A. 该代码可用于求解二叉树的深度

B. 代码中函数 Depth( ) 的参数 T 表示根节点，非根节点不可以作为参数

C. 代码中函数 Depth( ) 采用了递归方法

D. 代码中函数 Depth( ) 可用于求解各种形式的二叉树深度，要求该二叉树节点至少有 left 和 right 属性

74.单选题 [GESP202312【6 级】] 基于第 4 题的定义，有关下面 C++代码的说法正确的是（ ）。

```
struct BiNode {
    char data;
    BiNode* lchild, *rchild;
};

class BiTree {
private:
    BiNode* Creat();
    void Release(BiNode* bt);
    BiNode* root;

public:
    BiTree() {
        root = Creat();
    }
    ~BiTree() {
        Release(root);
    }

    void Order(BiNode* bt) {
        if (bt == nullptr)
            return;
        else {
            cout << bt->data;
            Order(bt->lchild);
            Order(bt->rchild);
        }
    }
};
```

A. 代码中 Order( ) 函数是中序遍历二叉树的方法

B. 代码中 Order( ) 先访问根节点，然后对左子树进行前序遍历，再对右子树前序遍历

C. 代码中 Order( ) 先访问中序遍历左子树，然后访问根节点，最后则是中序遍历右子树

D. 代码中 Order( ) 先后序遍历左子树，然后后序遍历右子树，最后访问根节点

75.填空题 [NOIP 提高组 2012]

```
#include <iostream>
#include <string>
using namespace std;
int lefts[20], rights[20], father[20];
string s1, s2, s3;
int n, ans;
void calc(int x, int dep){
    ans = ans + dep*(s1[x] - 'A' + 1);
    if (lefts[x] >= 0) calc(lefts[x], dep+1);
    if (rights[x] >= 0) calc(rights[x], dep+1);
}
void check(int x){
    if (lefts[x] >= 0) check(lefts[x]);
    s3 = s3 + s1[x];
    if (rights[x] >= 0) check(rights[x]);
}
void dfs(int x, int th){
    if (th == n){
        s3 = """;
        check(0);
        if (s3 == s2){
            ans = 0;
            calc(0, 1);
            cout<<ans<<endl;
        }
        return;
    }
    if (lefts[x] == -1 && rights[x] == -1){
        lefts[x] = th;
        father[th] = x;
        dfs(th, th+1);
        father[th] = -1;
        lefts[x] = -1;
    }
    if (rights[x] == -1){
        rights[x] = th;
        father[th] = x;
        dfs(th, th+1);
        father[th] = -1;
        rights[x] = -1;
    }
```

```
    if (father[x] >= 0)dfs(father[x], th);
}
int main(){
    cin>>s1;
    cin>>s2;
    n = s1.size();
    memset(lefts, -1, sizeof(lefts));
    memset(rights, -1, sizeof(rights));
    memset(father, -1, sizeof(father));
    dfs(0, 1);
}
```

输入：

ABCDEF

BCAEDF

输出：____

76.填空题 [NOIP 普及组 2008]

```
#include <iostream>
#include <cstring>
using namespace std;
#define MAX 100

void solve(char first[], int spos_f, int epos_f, char mid[], int spos_m, int epos_m)
{
    int i, root_m;
    if (spos_f > epos_f)
        return;
    for (i = spos_m; i <= epos_m; i++)
    {
        if (first[spos_f] == mid[i])
        {
            root_m = i;
            break;
        }
    }
    solve(first, spos_f + 1, spos_f + (root_m - spos_m), mid, spos_m, root_m - 1);
    solve(first, spos_f + (root_m - spos_m) + 1, epos_f, mid, root_m + 1, epos_m);
    cout << first[spos_f];
}

int main()
```

```
{
    char first[MAX], mid[MAX];
    int len;
    cin >> len;
    cin >> first >> mid;
    solve(first, 0, len - 1, mid, 0, len - 1);
    cout << endl;
    return 0;
}
```

输入： 7

ABDCEGF

BDAGECF

输出：

77.单选题 [CSP-J2019]

```
#include <iostream>
using namespace std;
const int maxn = 10000;
int n;
int a[maxn];
int b[maxn];
int f(int l, int r, int depth) {
    if (l > r)
        return 0;
    int min = maxn, mink;
    for (int i = l; i <= r; ++i) {
        if (min > a[i]) {
            min = a[i];
            mink = i;
        }
    }
    int lres = f(l, mink - 1, depth + 1);
    int rres = f(mink + 1, r, depth + 1);
    return lres + rres + depth * b[mink];
}
int main() {
    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> a[i];
    for (int i = 0; i < n; ++i)
        cin >> b[i];
```

```
27      cout << f(0, n - 1, 1) << endl;
28      return 0;
29  }
```

判断题

如果 a 数组有重复的数字，则程序运行时会发生错误。( )

如果 b 数组全为 0，则输出为 0。( )

选择题

当 n=100 时，最坏情况下，与第 12 行的比较运算执行的次数最接近的是：（ ）。

当 n=100 时，最好情况下，与第 12 行的比较运算执行的次数最接近的是：（ ）。

当 n=10 时，若 b 数组满足，对任意 0≤i<n，都有 b[i] = i + 1，那么输出最大为 ( )。

当 n=100 时，若 b 数组满足，对任意 0≤i<n，都有 b[i]=1，那么输出最小为（ ）。

1.

A. 正确

B. 错误

2.

A. 正确

B. 错误

3.

A. 5000

B. 600

C. 6

D. 100

4.

A. 100

B. 6

C. 5000

D. 600

5.

A. 386

B. 383

C. 384

D. 385

6.

A. 582

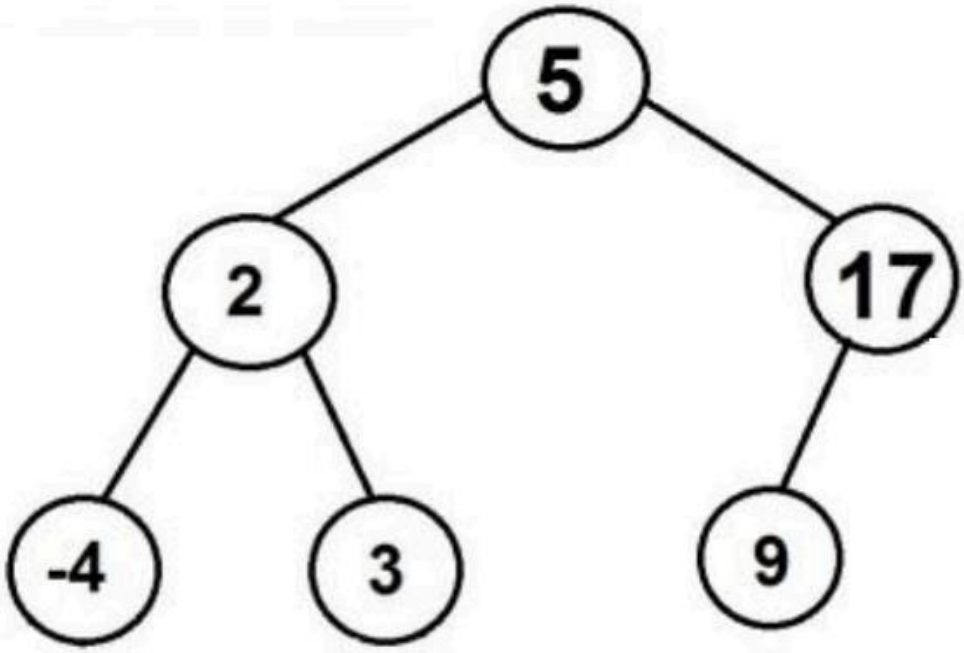
B. 580

C. 579

D. 581

78.单选题 [GESP202406【6 级】] 阅读以下二叉树的广度优先搜索代码:

```
#include <iostream>
#include <queue>
using namespace std;
// 二叉树节点的定义
struct TreeNode {
    int val;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
};
// 宽度优先搜索（BFS）迭代实现
TreeNode* bfs(TreeNode* root, int a) {
    if (root == nullptr) return nullptr;
    queue<TreeNode*> q;
    q.push(root);
    while (!q.empty()) {
        TreeNode* node = q.front();
        q.pop();
        if (node->val == a)
            return node;
        cout << node->val << " "; // 先访问当前节点
        if (node->left) q.push(node->left); // 将左子节点入队
        if (node->right) q.push(node->right); // 将右子节点入队
    }
    return nullptr;
}
```



使用以上算法，在以下这棵树搜索数值 20 时，可能的输出是( )。

A. 5 2 -4 3 17 9

B. -4 2 3 5 9 17

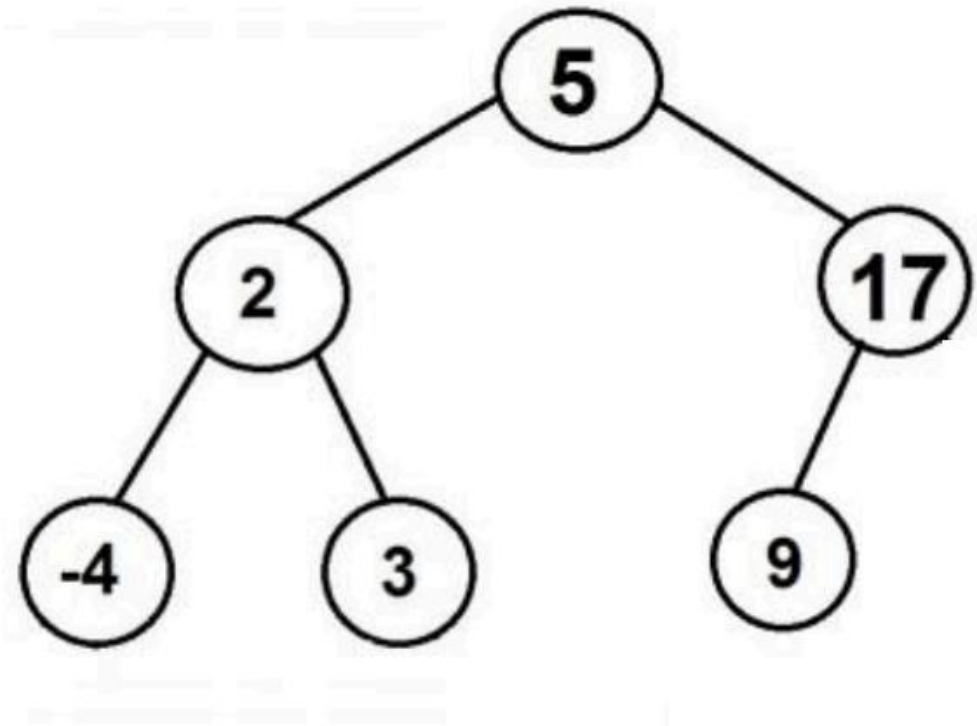
C. 5 2 17 -4 3 9

D. 以上都不对

79.单选题 [GESP202406【6 级】] 阅读以下二叉树的广度优先搜索代码，在如下的树中搜索数值 时，在二叉树搜索数值 20 时，可能的输出是( )

```
#include <iostream>
#include <stack>
using namespace std;
// 非递归深度优先搜索（DFS）
TreeNode* dfs(TreeNode* root, int a) {
    if (root == nullptr) return nullptr;
    stack<TreeNode*> stk;
    stk.push(root);
    while (!stk.empty()) {
        TreeNode* node = stk.top();
        stk.pop();
        if (node->val == a)
            return node;
```

```
            cout << node->val << " "; // 访问当前节点
            if (node->right) stk.push(node->right); // 先压入右子节点
            if (node->left) stk.push(node->left); // 再压入左子节点
        }
        return nullptr;
}
```



A. 5 2 -4 3 17 9

B. -4 2 3 5 9 17

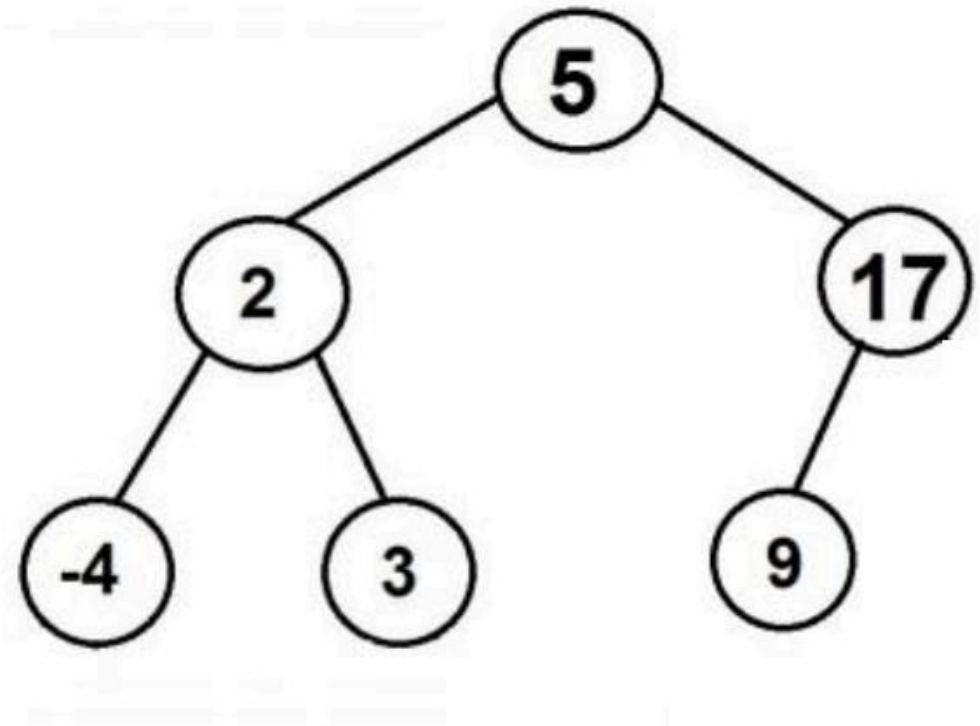
C. 5 2 17 -4 3 9

D. 以上都不对

80.单选题 [GESP202406【6 级】] 阅读以下二叉树的广度优先搜索代码，在如下的树中搜索数值 3 时，采用深度优先搜索一共比较的节点数为（ ）

```
#include <iostream>
#include <stack>
using namespace std;
// 非递归深度优先搜索（DFS）
TreeNode* dfs(TreeNode* root, int a) {
    if (root == nullptr) return nullptr;
    stack<TreeNode*> stk;
    stk.push(root);
```

```
    while (!stk.empty()) {
        TreeNode* node = stk.top();
        stk.pop();
        if (node->val == a)
            return node;
        cout << node->val << " "; // 访问当前节点
        if (node->right) stk.push(node->right); // 先压入右子节点
        if (node->left) stk.push(node->left); // 再压入左子节点
    }
    return nullptr;
}
```



A. 2

B. 3

C. 4

D. 5

4. 哈夫曼树

81.判断题 [GESP202403【6 级】] 哈夫曼树是一种二叉树。

82.判断题 [GESP202406【6 级】] 哈夫曼编码本质上是一种贪心策略。

83.判断题 [GESP202403【6 级】/GESP202309【6 级】] 哈夫曼编码的主要应用领域是有损数据压缩。

84.判断题 [GESP202312【6 级】] 哈夫曼编码（Huffman Coding）具有唯一性，因此有确定的压缩率。

85.判断题 [GESP 样题【6 级】] 哈夫曼编码树中，两个频率相同的字符一定具有相同的哈夫曼编码。

86.判断题 [GESP202403【6 级】] 使用哈夫曼编码对一些字符进行编码，如果两个字符的频率差异最大，则它们的编码可能出现相同的前缀。

87.单选题 [NOIP 提高组 2015/CSP-J2021] 在数据压缩编码的应用中，哈夫曼（Huffman）算法是一种采用了（　　）思想的算法

A. 贪心

B. 分治

C. 递推

D. 回溯

88.单选题 [GESP202403【6 级】] 在构建哈夫曼树时，每次应该选择（ ）合并。

A. 最小权值的节点

B. 最大权值的节点

C. 随机节点

D. 深度最深的节点

89.不定项选择题 [NOIP 提高组 2015] 下列有关树的叙述中，叙述正确的有（　　）。

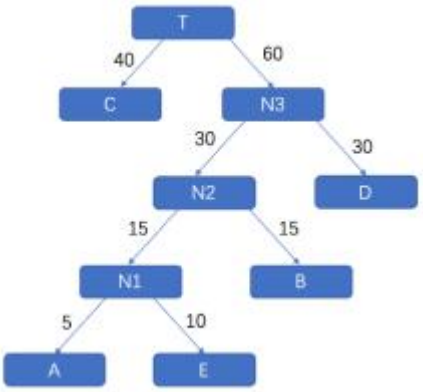
A. 在含有 n 个结点的树中，边数只能是(n-1)条

B. 在哈夫曼树中，叶结点的个数比非叶结点个数多 1

C. 完全二叉树一定是满二叉树

D. 在二叉树的前序序列中，若结点 u 在结点 v 之前，则 u 一定是 v 的祖先

90.单选题 [GESP 样题【6 级】] 如下图所示的哈夫曼树，按照哈夫曼编码规则，假设图中字符 C 的编码为 0，则 E 的编码为（ ）。



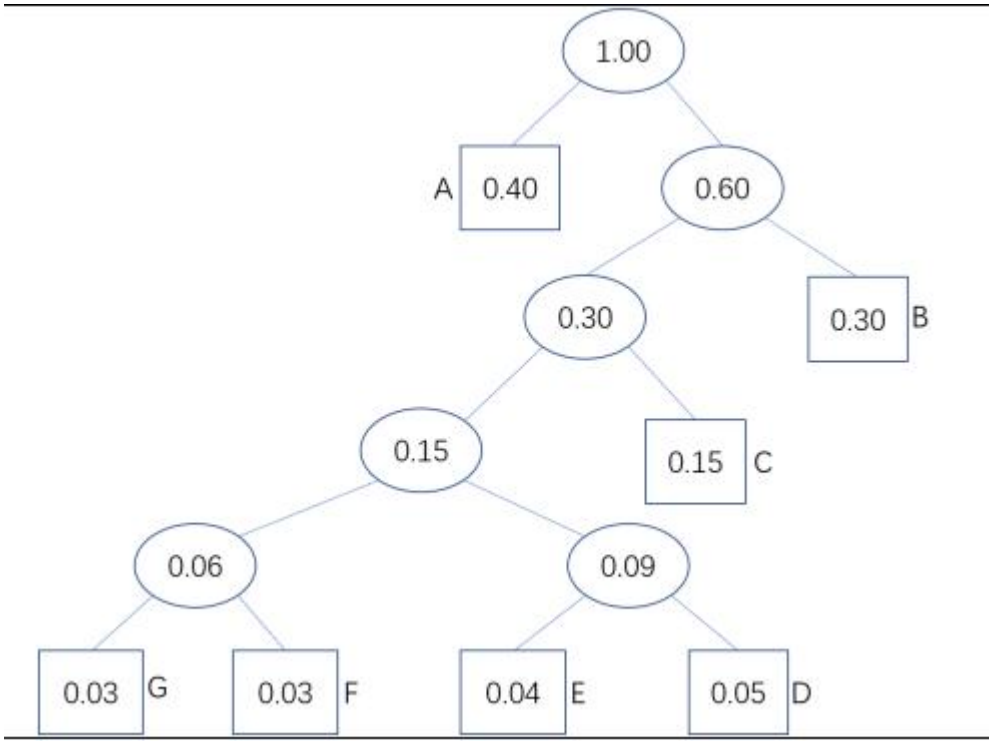
A．1111

B．1010

C．1101

D．1001

91.单选题 [GESP202309【6 级】] 某内容仅会出现 ABCDEFG ，其对应的出现概率为 0.40、0.30、0.15、0.05、0.04、0.03、0.03，如下图所示。按照哈夫曼编码规则，假设 B 的编码为 11 ，则 D 的编码为（ ）。



A. 10010

B. 10011

C. 10111

D. 10001

92.单选题 [GESP202406【6 级】] 对“classmycls”使用哈夫曼（Huffman）编码，最少需要（ ）比特。

A. 10

B. 20

C. 25

D. 30

93.单选题 [CSP-J2023] 假设有一组字符 {a,b,c,d,e,f}，对应的频率分别为 5%,9%,12%,13%,16%,45%。请问以下哪个选项是字符 abcdef 分别对应的一组哈夫曼编码？

A. 1111,1110,101,100,110,0

B. 1010,1001,1000,011,010,00

C. 000,001,010,011,10,11

D. 1010,1011,110,111,00,01

94.单选题 [NOIP 普及组 2011/NOIP 提高组 2011] 现有一段文言文，要通过二进制哈夫曼编码进行压缩。简单起见，假设这段文言文只由 4 个汉 字“之”、“呼”、“者”、“也”组成，它们出现的次数分别为 700、600、300、200。那么，“也”字的编码长度是（ ）。

A. 1

B. 2

C. 3

D. 4

95.单选题 [CSP-J2022] 假设字母表{a,b,c,d,e} 在字符串出现的频率分别为 10%，15%，30%，16%，29%。若使用哈夫曼编码方式对字母进行不定长的二进制编码，字母 d 的编码长度（ ）位。

A.1

B.2

C.2 或 3

D.3

96.单选题 [GESP202312【6 级】] 对 hello world 使用霍夫曼编码（Huffman Coding），最少 bit（比特）为（ ）。

A. 4

B. 32

C. 64

D. 88

97.单选题 [NOIP 提高组 2009] 最优前缀编码，也称 Huffman 编码。这种编码组合的特点是对于较频繁使用的元素给与较短的唯一编码，以提高通讯的效率。下面编码组合哪一组不是合法的前缀编码。

A.（00，01，10，11）

B.（0，1，00，11）

C.（0，10，110，111）

D.（1，01，000，001）

5. 表达式树

98.单选题 [NOIP 普及组 2009] 表达式 a*(b+c)-d 的后缀表达式是：

A. abcd*+-

B. abc+*d-

C. abc*+d-

D. -+*abcd

99.单选题 [NOIP 提高组 2009] 表达式 a*(b+c)-d 的后缀表达式是：

A. abcd*+-

B. abc+*d-

C. abc*+d-

D. -+*abcd

100.单选题 [NOIP 提高组 2016] 表达式 a*(b+c)-d 的后缀表达形式为（ ）。

A. abcd*+-

B. abc+*d-

C. abc*+d-

D. -+*abcd

101.单选题 [NOIP 普及组 2017/NOIP 提高组 2017] 表达式 a * (b + c) * d 的后缀形式是()。

A. abcd*+*

B. abc+*d*

C. a*bc+*d

D. b+c*a*d

102.单选题 [CSP-S2020] 表达式 a*(b+c)-d 的后缀表达形式为（ ）。

A. abc*+d−

B. −+*abcd

C. abcd*+−

D. abc+*d−

103.单选题 [CSP-J2021] 表达式 a*(b+c)*d 的后缀表达式为( ),其中*和+是运算符。

A.**a+bcd

B.abc+*d*

C.abc+d**

D.*a*+bcd

104.单选题 [CSP-J2023] 后缀表达式 6 2 3 + - 3 8 2 / + * 2 ^ 3 + 对应的中缀表达式是

A. ((6-(2+3))*(3+8/2))^2+3

B. 6-2+3*3+8/2^2+3

C. (6-(2+3))*((3+8/2)^2)+3

D. 6-((2+3)*(3+8/2))^2+3

105.单选题 [CSP-J2022] 对表达式 a+(b-c)*d 的前缀表达式为（ ），其中 +、-、* 是运算符。

A. *+a-bcd

B. +a*-bcd

C. abc-d*+

D. abc-+d

106.单选题 [NOIP 提高组 2018] 表达式 a * d - b * c 的前缀形式是（ ）。

A. a d * b c * -

B. - * a d * b c

C. a * d - b * c

D. - * * a d b c

107.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 前缀表达式“ + 3 * 2 + 5 12 ”的值是（ ）

A. 23

B. 25

C. 37

D. 65

答案

1.T

2.D

3.B

4.C

5.A

6.D

7.A

8.ABC

9.ABD

10.F

11.F

12.F

13.F

14.B

15.B

16.CD

17.B

18.B

19.T

20.C

21.C

22.E

23.E

24.B

25.ABC

26.1008

27.

1

11

28.D

29.D

30.A

31.A

32.CD

33.C

34.B

35.A

36.D

37.A

38.A

39.C

40.C

41.D

42.A

43.42

44.5536

45.A

46.B

47.C

48.D

49.F

50.A

51.D

52.A

53.D

54.B

55.D

56.B

57.A

58.B

59.A

60.A

61.BC

62.ABD

63.ABD

64.B

65.C

66.B

67.B

68.C

69.C

70.BC

71.A

72.C

73.B

74.B

75.55

76.DBGEFCA

77.BAADDB

78.C

79.A

80.C

81.T

82.T

83.F

84.F

85.F

86.F

87.A

88.A

89.ABCD

90.D

91.B

92.C

93.A

94.C

95.B

96.B

97.B

98.B

99.B

100.B

101.B

102.D

103.B

104.A

105.B

106.B

107.C

GW 信奥 CSP 初赛习题集 7-4

数据结构——图

1. 图的存储

1.不定项选择题 [NOIP 提高组 2014/CSP-S2019] 以下哪些结构可以用来存储图（）.

A.邻接矩阵

B.栈

C.邻接表

D.二叉树

2.单选题 [GESP202312【8 级】] 使用邻接矩阵表达 n 个顶点的有向图，则该矩阵的大小为（ ）。

A.n*(n+1)

B.n*n

C.n*(n-1)

D.n*(n-1)/2

3.单选题 [GESP202403【8 级】] 使用邻接表表达一个无向简单图，图中包含 v 个顶点、 e 条边，则该表中边节点的个数为（ ）。

A.v*(v-1)

B.v*v

C.2*e

D.e

4.单选题 [CSP-J2022] 考虑由 N 个顶点构成的有向连通图，采用邻接矩阵的数据结构表示时，该矩阵中至少存在（ ）个非零元素。

A.N−1

B.N

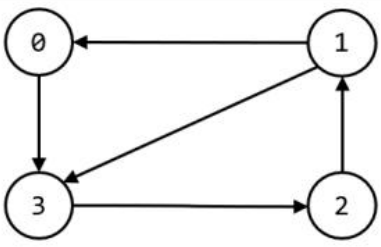
C.N+1

D.N^2

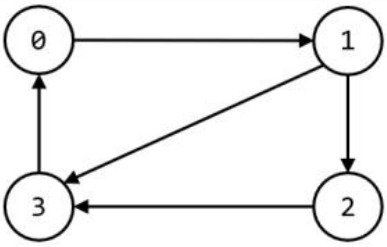
5.单选题 [GESP202312【8 级】] 下面的程序使用出边的邻接表表达有向图，则下列选项中哪个是它表达的图？（ ）。

```
#include <iostream>

struct Edge {
    int e;
    Edge * next;
};

struct Node {
    Edge * first;
};

int main() {
    Edge e[5] = {{1, nullptr}, {2, &e[2]}, {3, nullptr}, {3, nullptr}, {0, nullptr}};
    Node n[4] = {&e[0], &e[1], &e[3], &e[4]};
    // 先输出数组
    return 0;
}
```

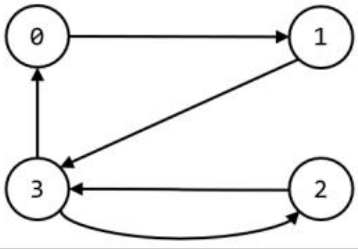
A.



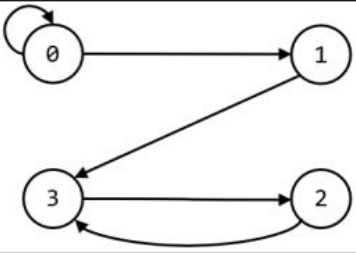
B.



C.



D.



A.A

B.B

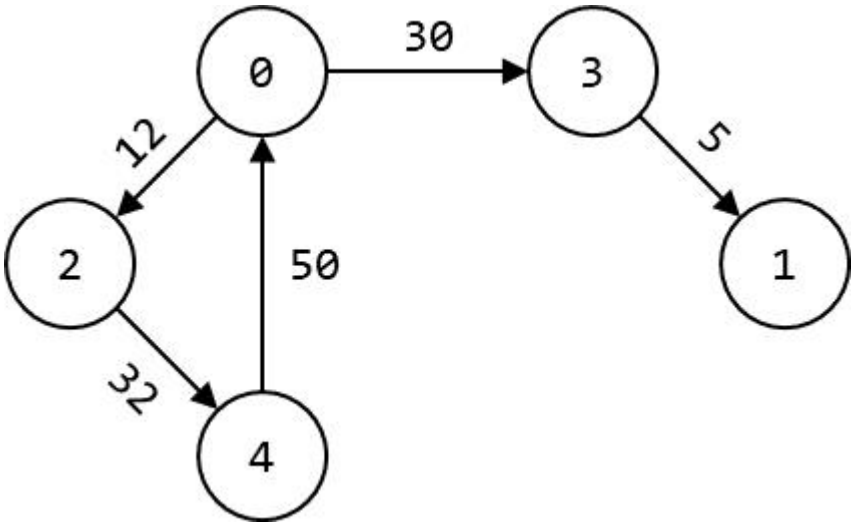
C.C

D.D

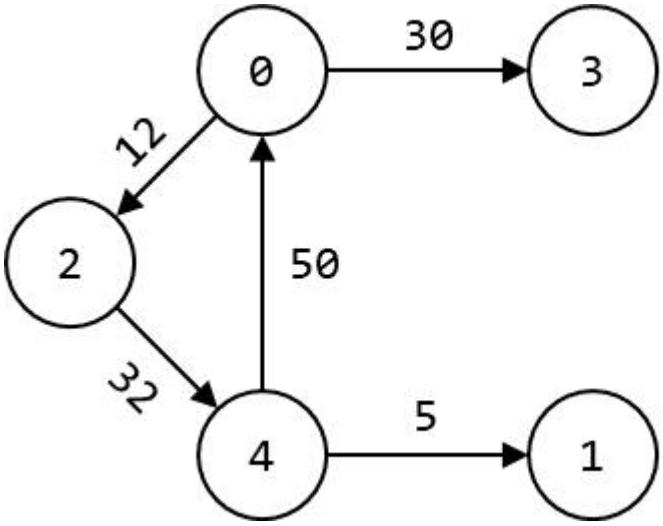
6.单选题 [GESP202406【7 级】] 如下图所示的邻接矩阵（inf 表示无穷大），表示的是下列哪个选项中的图

	0	1	2	3	4
0	inf	inf	12	30	inf
1	inf	inf	inf	inf	inf
2	inf	inf	inf	inf	32
3	inf	5	inf	inf	inf
4	50	inf	inf	inf	inf

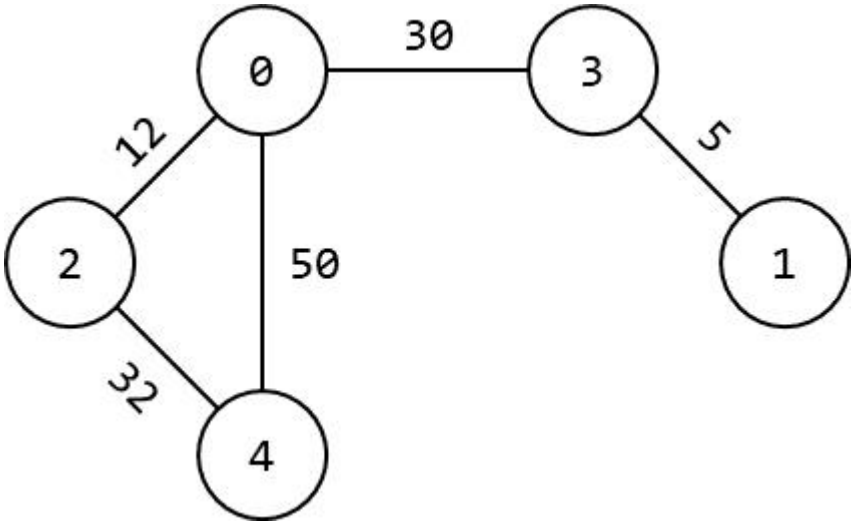
A.



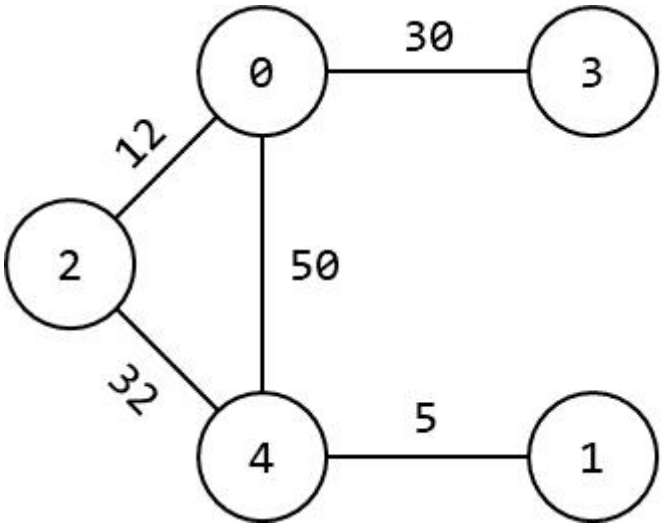
B.



C.



D.



2. 图的性质

1. 有向图与无向图

7.单选题 [NOIP 普及组 2014] 有向图中每个顶点的度等于该顶点的（）。

A. 入度

B. 出度

C. 入度和出度之和

D. 入度和出度之差

8.单选题 [NOIP 提高组 2014] 在无向图中，所有顶点的度数之和是边数的（)倍 .

A.0.5

B.1

C.2

D.4

9.单选题 [NOIP 普及组 2016] 设简单无向图 G 有 16 条边且每个顶点的度数都是 2，则图 G 有（)个顶点。

A. 10

B. 12

C. 8

D. 16

10.单选题 [GESP 样题【7 级】] 4 个结点的简单有向图，最多可以有多少条边（ ）。

A. 4

B. 6

C. 8

D. 12

2. 完全图

11.判断题 [GESP202312【8 级】] N 个顶点的有向完全图（不带自环）有 N*(N-1)/2 条边。

12.判断题 [GESP202403【8 级】] N 个顶点的无向完全图有 N*(N-1)条边。

13.单选题 [GESP202403【7 级】] 一个简单有向图有 10 个结点、30 条边。再增加多少条边可以成为完全图。（ ）

A. 60

B. 70

C. 15

D. 20

14.单选题 [GESP 样题【7 级】] 一个简单有向图有 20 个结点，假设图中已经存在 300 条边，请问增加多少条边可以成为完全图。（ ）

A. 77

B. 78

C. 79

D. 80

15.单选题 [NOIP 普及组 2011] 无向完全图是图中每对顶点之间都恰有一条边的简单图。已知无向完全图 G 有 7 个顶点，则它共有（ ）条边。

A. 7

B. 21

C. 42

D. 49

3. 其他

16.填空题 [NOIP 提高组 2011] 平面图是可以画在平面上、且它的边仅在顶点上才能相交的简单无向图。 4 个顶点的平面图至多有 6 条边，如右图所示。那么， 5 个顶点的平面 图至多有 _____ 条边。

3. 图的遍历

17.判断题 [GESP202312【6 级】] 深度优先搜索（DFS，Depth First Search 的简写）属于图算法，其过程是对每一个可能的分支路径深入到不能再深入为止，而且每个节点只能访问一次。

18.判断题 [GESP 样题【7 级】] 简单有向图的深搜结果和广搜结果一样。

19.判断题 [GESP202406【7 级】] 非连通图不能使用广度优先搜索算法进行遍历。

20.单选题 [GESP202312【7 级】] 图的广度优先搜索中既要维护一个标志数组标志已访问的图的结点，还需哪种结构存放结点以实现遍历？( )

A. 双向栈

B. 队列

C. 哈希表

D. 堆

21.单选题 [CSP-J2022] 以下对数据结构的表述不恰当的一项为：（ ）。

A. 图的深度优先遍历算法常使用的数据结构为栈。

B. 栈的访问原则后进先出，队列的访问原则是先进先出。

C. 队列常常被用于广度优先搜索算法。

D. 栈与队列存在本质不同，无法用栈实现队列。

22.单选题 [GESP202312【7 级】] 学生在读期间所上的某些课程中需要先上其他的课程，所有课程和课程间的先修关系构成一个有向图 G ，有向边 <U, V> 表示课程 U 是课程 V 的先修课，则要找到某门课程 C 的全部先修课下面哪种方法不可行？( )

A. BFS 搜索

B. DFS 搜索

C. DFS+BFS

D. 动态规划

23.单选题 [GESP202406【7 级】] 图的存储和遍历算法，下面说法错误的是（ ）。

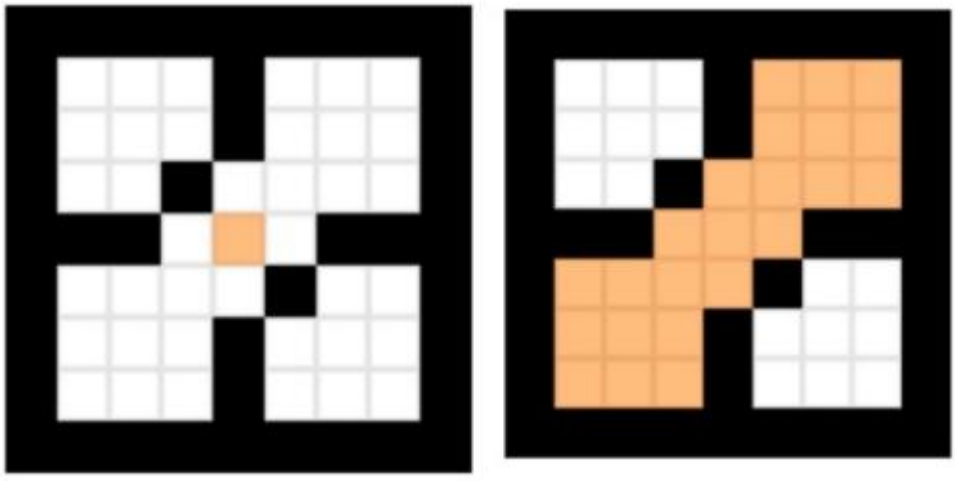
A. 图的深度优先搜索和广度优先搜索对有向图和无向图都适用。

B. 图的深度优先搜索和二叉树的先序遍历道理是不一样的。

C. 图的深度优先搜索需要借助栈来完成。

D. 邻接表中，顶点 v 对应的链表中的边结点数目正好是顶点 v 的度。

24.判断题 [GESP202312【7 级】] 小杨在开发画笔刷小程序（applet），操作之一是选中黄颜色，然后在下面的左图的中间区域双击后，就变成了右图。这个操作可以用图的泛洪算法来实现。



25.判断题 [GESP202309【6 级】] 如果节点数为 N，广度搜索算法的最差时间复杂度为 O(N)。

26.单选题 [NOIP 提高组 2015/CSP-S2020/GESP202406【8 级】] 具有 n 个顶点，e 条边的图采用邻接表存储结构，进行深度优先遍历和广度优先遍历运 算的时间复杂度均为（ ）。

A. $O(n^2)$

B. $O(e^2)$

C. O(ne)

D. O(n+e)

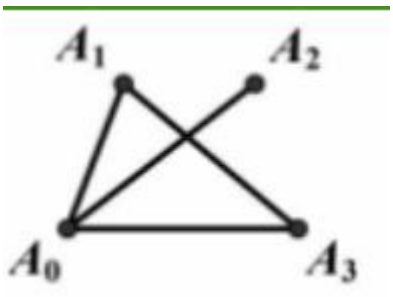
27.单选题 [NOIP 普及组 2013] 以 A 0 作为起点，对下面的无向图进行 深度 优先遍历时，遍历顺序 不可能 是（ ）。

A. A 0, A 1, A 2, A 3

B. A 0, A 1, A 3 , A 2

C. A 0, A 2, A 1, A 3

D. A 0, A 3, A 1 , A 2



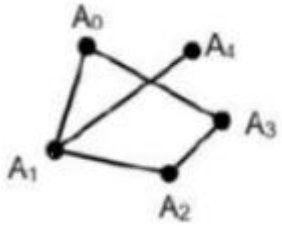
28.不定项选择题 [NOIP 提高组 2013] 以 A0 作为起点，对下面的无向图进行深度优先遍历时（遍历的顺 序 与 顶 点 字 母 的 下 标 无 关 ），最 后 一 个 遍 历 到 的 顶 点 可 能 是 （ ）。

A. A1

B. A2

C. A3

D. A4



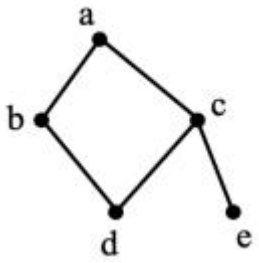
29.单选题 [CSP-J2021] 以 a 为起点，对下边的无向图进行深度优先遍历，则 b,c,d,e 四个点中有可能作为最后一个遍历到的点的个数为（ ）。

A. 1

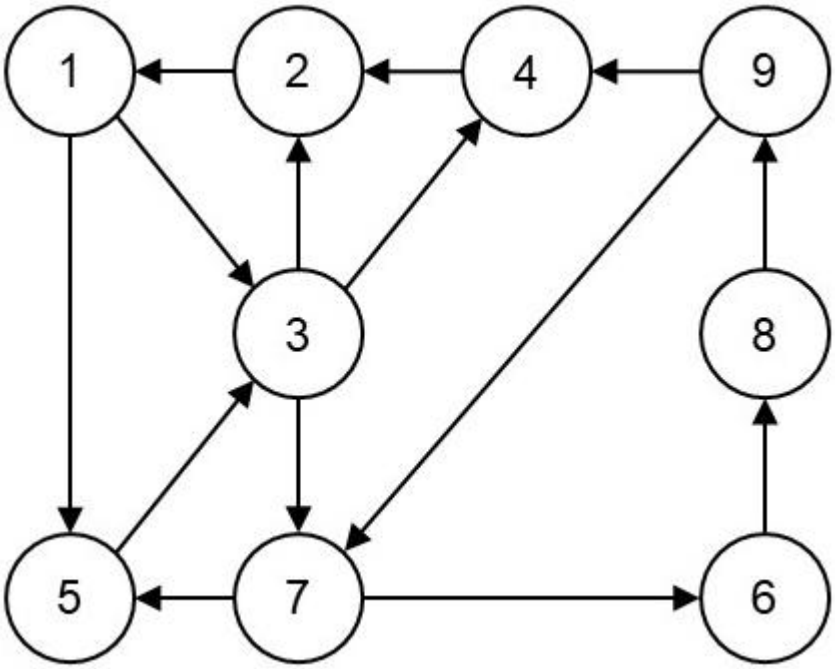
B. 2

C. 3

D. 4



30.单选题 [GESP202406【7 级】] 下列选项中，哪个可能是下图的深度优先遍历序列（ ）。



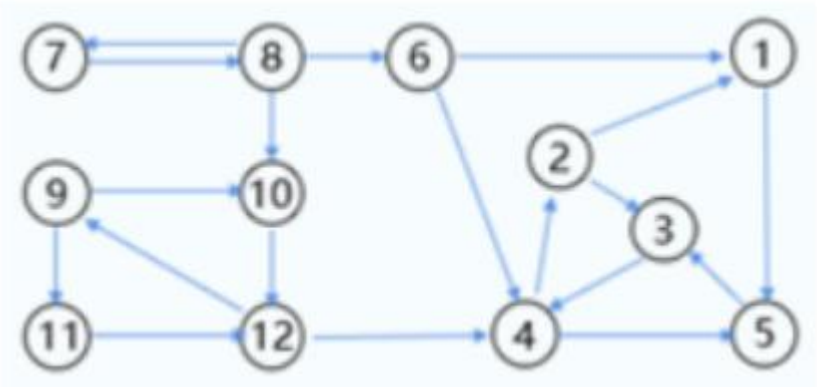
A.A. 1, 3, 7, 5, 4, 2, 6, 8, 9

B. 9, 4, 2, 1, 3, 5, 7, 6, 8

C. 1, 3, 4, 2, 7, 6, 8, 9, 5

D. 9, 7, 6, 8, 4, 2, 1, 5, 3

31.单选题 [GESP202403【7 级】] 下列选项中，哪个可能是下图的深度优先遍历序列（ ）。



A. 8, 6, 1, 5, 3, 4, 2, 10, 7, 12, 11, 9

B. 7, 8, 6, 4, 2, 1, 5, 3, 12, 9, 11, 10。

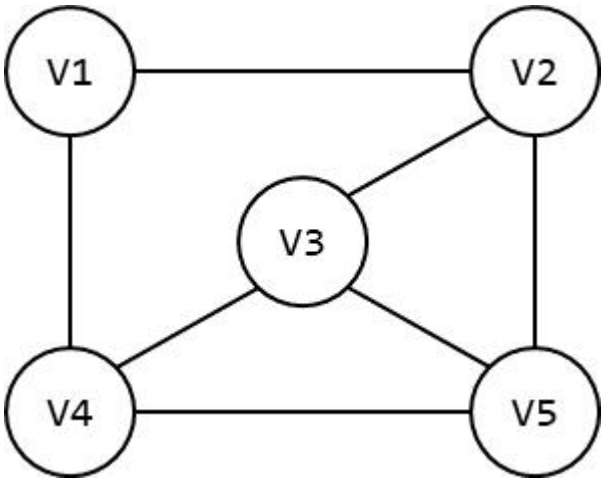
C. 8, 10, 12, 9, 11, 4, 5, 3, 2, 1, 6, 7

D. 7, 8, 10, 9, 11, 12, 4, 5, 1, 2, 3, 6。

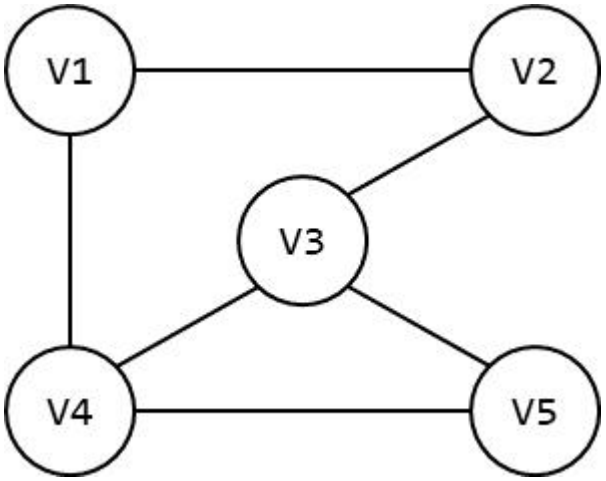
32.单选题 [GESP202406【7 级】] 下列选项中，哪个可能是下图的深度优先遍历序列（ ）。

0	V1	3	1	^	
1	V2	4	2	0	^
2	V3	4	3	1	^
3	V4	2	0	^	
4	V5	2	1	^	

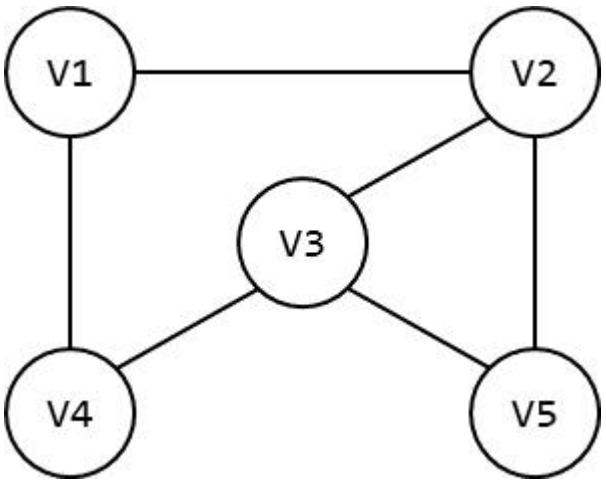
A.



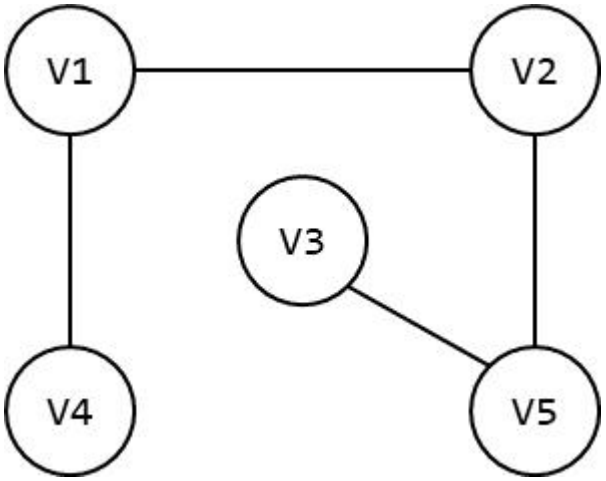
B.



C.



D.



33.单选题 [GESP 样题【7 级】] 阅读以下代码，visited 起到的作用是（ ）。

```
#include <iostream>
#include <string>
#include <cmath>
using namespace std;

int visited[100], k=0;

void dfs(int graph[][100], int start, int vexnum)
{
    visited[start]++;
    cout << start << "" "";
    for (int i=0; i<vexnum; i++) {
        if (graph[start][i] && !visited[i]) {
            dfs(graph, i, vexnum);
        }
    }
```

```
}
```

A. 实现遍历过程

B. 以广度优先的方式记录图中的顶点

C. 存储深搜时节点的访问顺序

D. 能够记录最短路径

34.判断题 [GESP 样题【8 级】] 下面是图的深度遍历的代码，则横线处可以填入：if(vis[x]) return。

```
void dfs(int x) { // 深度优先搜索
    ________________________;
    vis[x] = 1; // 标记节点 x 已经被访问
    cout << x << endl; // 访问当前节点
    for(int i = vFirst[x]; i != -1; i = e[i].next) { // 遍历每一个相邻的节点
            dfs(e[i].v);
    }
}
```

35.填空题 [NOIP 普及组 2018]

```
#include <cstdio>
int n, d[100];
bool v[100];
int main() {
    scanf("%d", &n);
    for (int i = 0; i < n; ++i) {
        scanf("%d", d + i);
        v[i] = false;
    }
    int cnt = 0;
    for (int i = 0; i < n; ++i) {
        if (!v[i]) {
            for (int j = i; !v[j]; j = d[j]) {
                v[j] = true;
            }
            ++cnt;
        }
    }
    printf("%d", cnt);
    return 0;
```

```
21 }
```

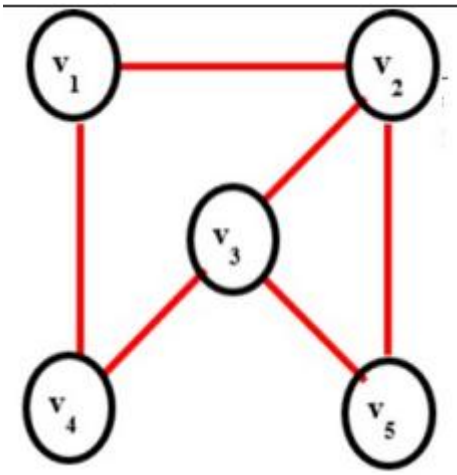
输入：

10 7 1 4 3 2 5 9 8 0 6

输出：（ ）

36.单选题 [GESP202312【7 级】] 从顶点 v1 开始遍历下图 G 得到顶点访问序列，在下面所给的 4 个序列中符合广度优先的序列有几个？( )

{v1 v2 v3 v4 v5} ， {v1 v2 v4 v3 v5} ， {v1 v4 v2 v3 v5} ， {v1 v2 v4 v5 v3}



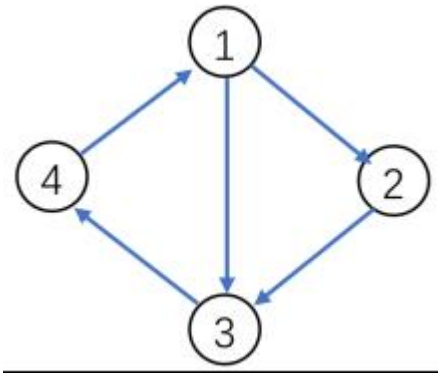
A. 4

B. 3

C. 2

D. 1

37.单选题 [GESP 样题【7 级】] 下面有向图中的数字表示顶点序号，则从 1 号顶点出发的 BFS 遍历的输出顶点序列可能是（ ）。



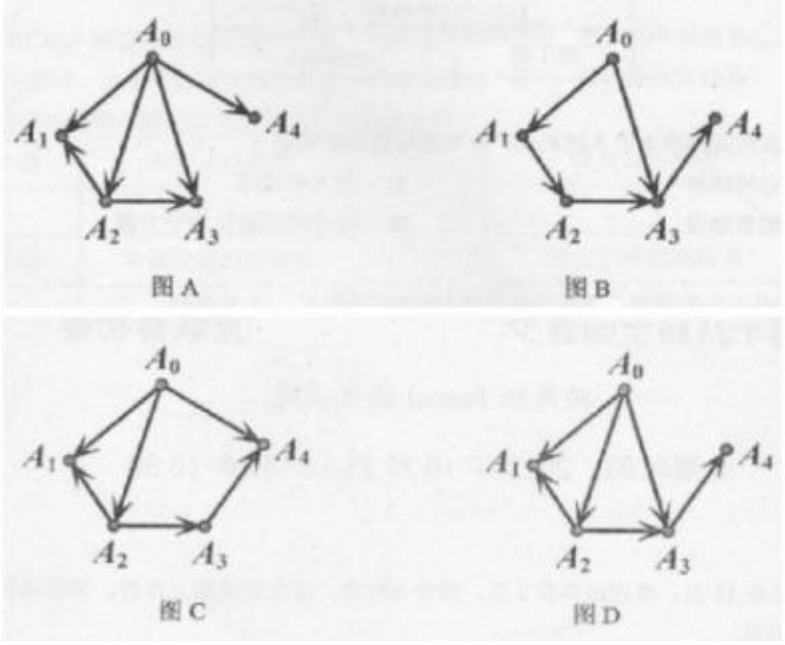
A. 1 4 3 2

B. 1 4 2 3

C. 1 3 2 4

D. 1 2 4 3

38.不定项选择题 [NOIP 提高组 2012] 从顶点 A0 出发，对有向图（　　）进行广度优先搜索（BFS）时，一种可能的遍历顺序是 A0，A1,A2，A3，A4 。



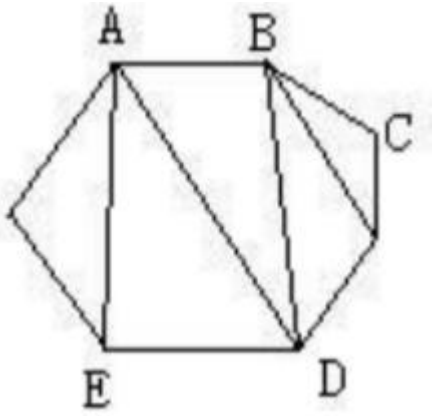
A. 图 A

B. 图 B

C. 图 C

D. 图 D

39.单选题 [NOIP 普及组 2004] 在下图中，从顶点（ ）出发存在一条路径可以遍历图中的每条边一次，而且仅遍历一次



A. A 点

B. B 点

C. C 点

D. D 点

E. E 点

4. 图的连通性

40.判断题 [GESP 样题【7 级】] 判断图是否连通可以用深搜实现。

41.判断题 [GESP202403【8 级】] 可以使用深度优先搜索算法判断图的连通性。

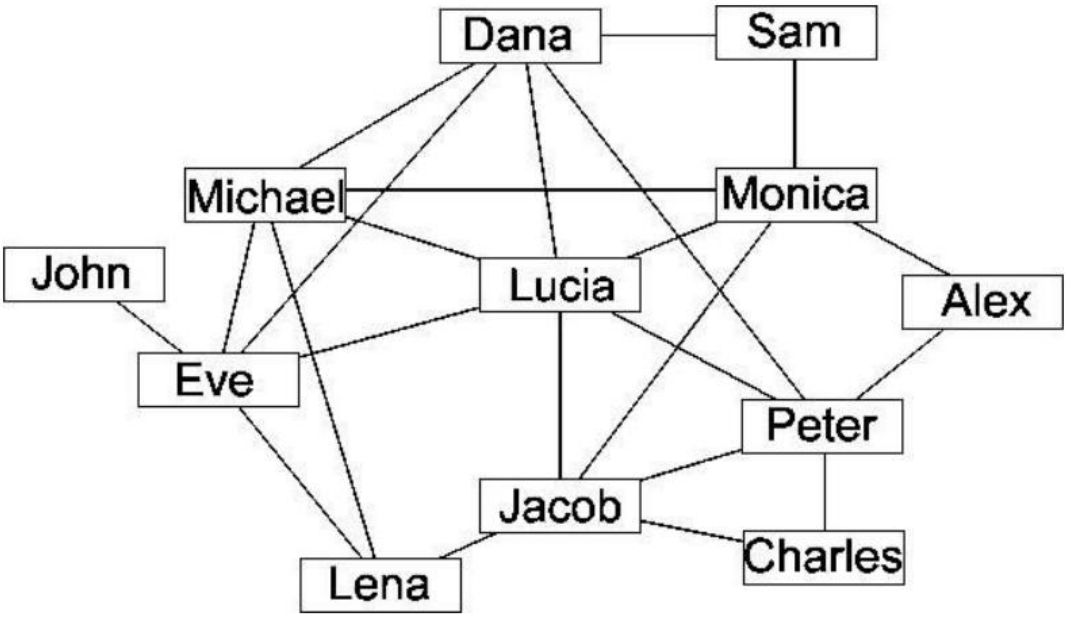
42.判断题 [GESP202312【7 级】] 广度优先搜索（BFS）能够判断图是否连通。

43.判断题 [GESP202312【8 级】] 判断图是否连通只能用广度优先搜索算法实现。

44.判断题 [GESP202406【8 级】] 要判断无向图的连通性，在深度优先搜索和广度优先搜索中选择，深度优先的平均时间复杂度更低。

45.单选题 [NOIP 普及组 2016/NOIP 提高组 2016] Lucia 和她的朋友以及朋友的朋友都在某社交网站上注册了账号。下图是他们 之间的关系图 ，两个人之间有边相连代表这两个人是朋友 ，没有边相连代表 不是朋友。这个社交网站的规则是 ：如果某人 A

向他 （她）的朋友 B 分享了 某张照片 ，那么 B 就可以对该照片进行评论 ；如果 B 评论了该照片 ，那么他 （她） 的所有朋友都可以看见这个评论以及被评论的照片 ，但是不能对该照片进行评论 （除非 A 也向他 （她）分享了该照片 ）。现在 Lucia 已经上传了 一张照片，但是她不想让 Jacob 看见这张照片，那么她可以向以下朋友（ ）分 享该照片。



A. Dana, Michael, Eve

B. Dana, Eve, Monica

C. Michael, Eve, Jacob

D. Micheal, Peter, Monica

46.单选题 [CSP-S2021] G 是一个非连通简单无向图（没有自环和重边），共有 36 条边，则该图至少有（ ）个点。

A. 8

B. 9

C. 10

D. 11

47.单选题 [GESP202403【7 级】] 一个连通的简单无向图，共有 28 条边，则该图至少有( )个顶点。

A. 6

B. 7

C. 8

D. 9

48.单选题 [CSP-S2019] G 是一个非连通无向图（没有重边和自环），共有 28 条边，则该图至少有 ( )个顶点。

A. 10

B. 9

C. 11

D. 8

49.单选题 [NOIP 提高组 2016/GESP202312【7 级】] G 是一个非连通简单无向图 ，共有 28 条边，则该图至少有 （ ）个顶点。

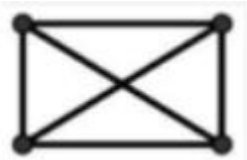
A. 10

B. 9

C. 8

D. 7

50.单选题 [NOIP 普及组 2013] 在一个无向图中,如果任意两点之间都存在路径相连，则称其为连通图。下图是一个有 4 个 顶点、 6 条边的连通图。若要使它不再是连通图，至少要删去其中的（ ）条边。



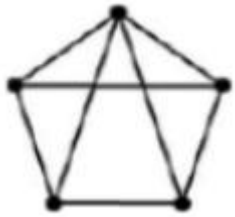
A. 1

B. 2

C. 3

D. 4

51.单选题 [NOIP 提高组 2013] 在一个无向图中，如果任意两点之间都存在路径相连，则称其为连通图。下图是一个有 5 个顶点、8 条边的连通图。若要使它不再是连通图，至少要删去其中的（ ）条边。



A. 2

B. 3

C. 4

D. 5

52.单选题 [CSP-J2020] 有 10 个顶点的无向图至少应该有（ ）条边才能确保是一个连通图。

A. 9

B. 10

C. 11

D. 12

53.单选题 [NOIP 提高组 2017] 由四个不同的点构成的简单无向连通图的个数是( )。

A. 32

B. 35

C. 38

D. 41

54.单选题 [NOIP 普及组 2018] 由四个没有区别的点构成的简单无向连通图的个数是（ ）。

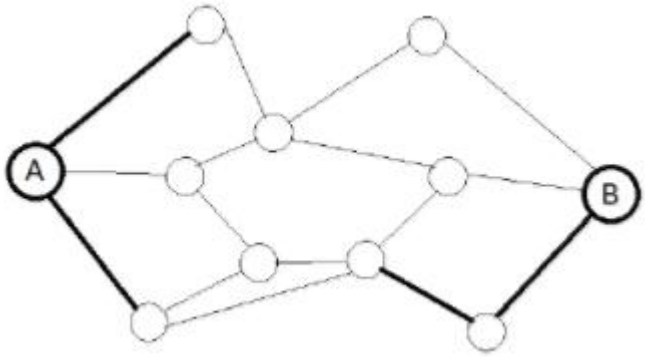
A. 6

B. 7

C. 8

D. 9

55.单选题 [NOIP 提高组 2017] 如下图所示，A 到 B 是连通的。假设删除一条细的边的代价是 1，删除一条粗的边的代价是 2，要让 A,B 不连通，最小代价是（_____）（2 分），最小代价的不同方案数是（_____）（3 分）。（只要有一条删除的边不同，就是不同的方案）



56.单选题 [GESP202406【7 级】] 下面关于图的说法正确的是（ ）。

A. 在无向图中，环是指至少包含三个不同顶点，并且第一个顶点和最后一个顶点是相同的路径。

B. 在有向图中，环是指一个顶点经过至少另一个顶点到自身的路径。

C. 在有向图中，如果任意两个顶点之间都存在一条边，则这个图一定是强连通图。

D. 在有向图中，所有顶点的入度和出度的总和就是图的边数的两倍。

57.单选题 [NOIP 普及组 2009] 已知 n 个顶点的有向图，若该图是强连通的（从所有顶点都存在路径到达其他顶点），则该图中最少有多少条有向边？

A. n

B. n+1

C. n-1

D. n*(n-1)

58.不定项选择题 [NOIP 提高组 2009] 若 3 个顶点的无权图 G 的邻接矩阵用数组存储为{{0，1，1}，{1，0，1}，{0，1，0}}，假定在具体存储中顶点依次为: v1，v2，v3。关于该图，下面的说法哪些是正确的:

A. 该图是有向图。

B. 该图是强连通的。

C. 该图所有顶点的入度之和减所有顶点的出度之和等于 1。

D. 从 v1 开始的深度优先遍历所经过的顶点序列与广度优先的顶点序列是相同的。

59.单选题 [CSP-S2022] 强连通图的性质不包括（ ）:

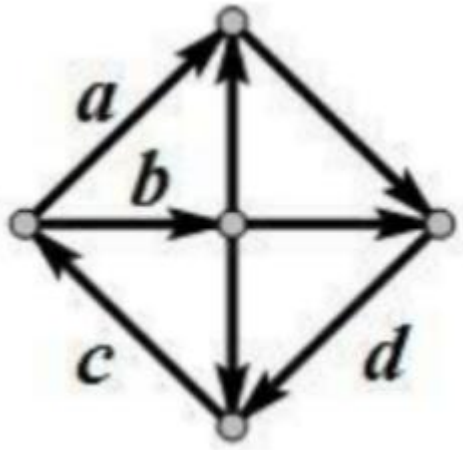
A. 每个顶点的度数至少为 1

B. 任意两个顶点之间都有边相连

C. 任意两个顶点之间都有路径相连

D. 每个顶点至少都连有一条边

60.单选题 [NOIP 普及组 2011] 对一个有向图而言，如果每个节点都存在到达其他任何节点的路径，那么就称它是强连通的。例如，右图就是一个强连通图。事实上，在删掉边( )后，它依然是强连通的。



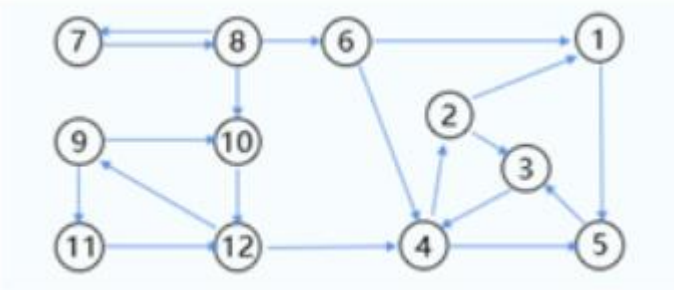
A. a

B. b

C. c

D. d

61.单选题 [GESP 样题【7 级】] 在下面的有向图中，强连通分量有多少个？（ ）



A. 3

B. 4

C. 5

D. 6

5. 图的拓扑排序

62.单选题 [NOIP 普及组 2010/NOIP 提高组 2010] 关于拓扑排序，下面说法正确的是（ ）。

A. 所有连通的有向图都可以实现拓扑排序

B. 对同一个图而言，拓扑排序的结果是唯一的

C. 拓扑排序中入度为 0 的结点总会排在入度大于 0 的结点的前面

D. 拓扑排序结果序列中的第一个结点一定是入度为 0 的点

63.单选题 [CSP-J2023] 考虑一个有向无环图，该图包含 4 条有向边：(1,2),(1,3),(2,4) 和(3,4)。以下哪个选项是这个有向无环图的一个有效的拓扑排序?

A. 4,2,3,1

B. 1,2,3,4

C. 1,2,4,3

D. 2,1,3,4

64.不定项选择题 [NOIP 提高组 2004/NOIP 普及组 2004] 某大学计算机专业的必修课及其先修课程如下表所示：

课程代号	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
课程名称	高等数学	程序设计语言	离散数学	数据结构	编译技术	操作系统	普通物理	计算机原理
先修课程			C_0, C_1	C_1, C_2	C_3	C_3, C_7	C_0	C_6

请你判断下列课程安排方案哪个是不合理的（ ）。

A. C0, C1, C2, C3, C4, C5, C6, C7

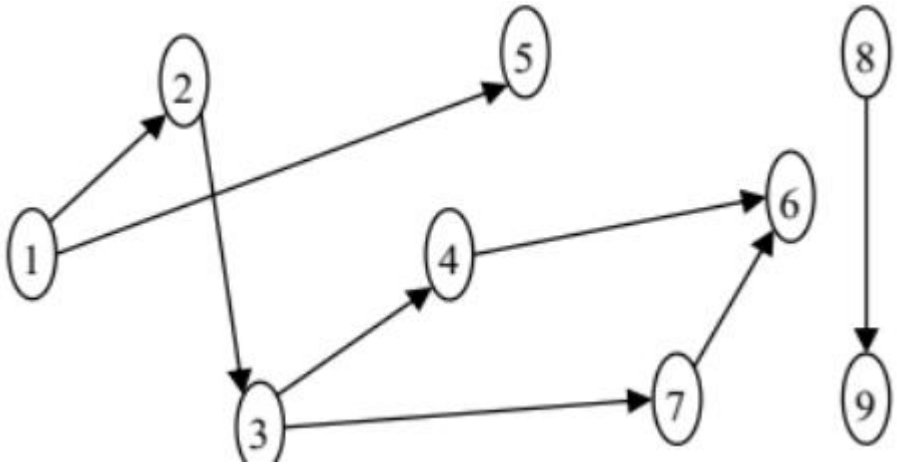
B. C0, C1, C2, C3, C4, C6, C7, C5

C. C0, C1, C6, C7, C2, C3, C4, C5

D. C0, C1, C6, C7, C5, C2, C3, C4

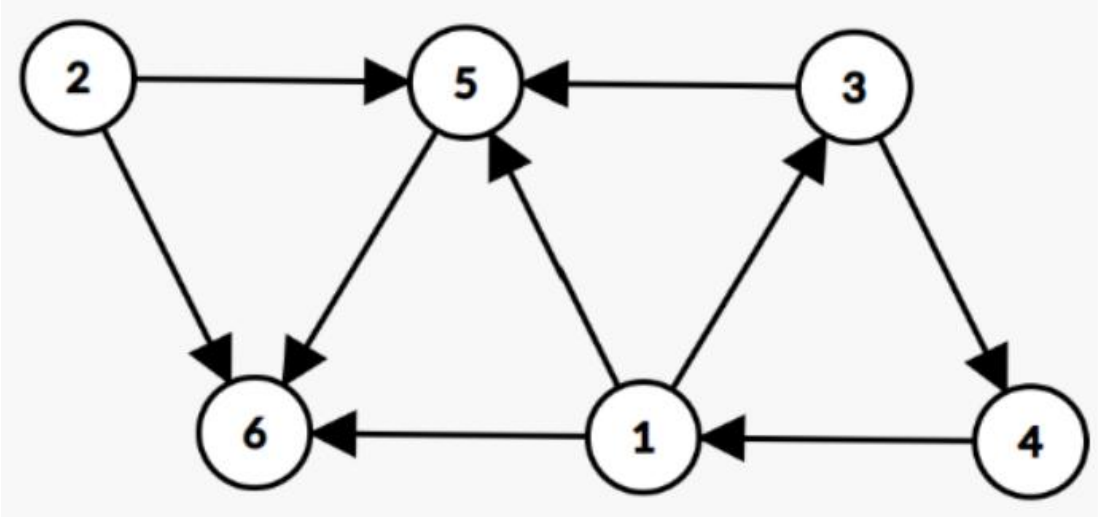
E. C0, C1, C2, C3, C6, C7, C5, C4

65.填空题 [NOIP 提高组 2009] 拓扑排序是指将有向无环图 G 中的所有顶点排成一个线性序列，使得图中任意一对顶点 u 和 v，若<u，v> ∈E(G.，则 u 在线性序列中出现在 v 之前，这样的线性序列成为拓扑序列。如下的有向无环图，对其顶点做拓扑排序，则所有可能的拓扑序列的个数为 。



66.单选题 [CSP-S2023] 如图是一张包含 6 个顶点的有向图，但顶点间不存在拓扑序。如果要删除其中一条边，使这 6 个顶

点能进行拓扑排序，请问总共有多少条边可以作为候选的被删除边？（ ）



A.1

B.2

C.3

D.4

6. 图的生成树

67.单选题 [GESP202403【8 级】] 关于生成树的说法，错误的是（ ）。

A. 一个无向连通图可以有多个生成树。

B. 一个无向图，只要连通，就一定有生成树。

C. n 个顶点的无向完全图，有 n^{n-2} 棵生成树。

D. n 个顶点的无向图，生成树包含 n-1 条边。

68.判断题 [GESP 样题【8 级】] 连通图 G 有 n 个顶点 m 条边，须删除 m-n+1 条边后才能变成一棵生成树。

69.单选题 [NOIP 提高组 2014] 设 G 是有 6 个结点的完全图，要得到一颗生成树，需要从 G 中删去（ ）条边。

A.6

B.9

C.10

D.15

70.单选题 [NOIP 普及组 2017/NOIP 提高组 2017/CSP-J2021] 设 G 是有 n 个结点、m 条边(n ≤m)的连通图，必须删去 G 的( )条边，才能使得 G 变成一棵树。

A. m－n+1

B. m-n

C. m+n+1

D. n－m+1

71.判断题 [GESP 样题【8 级】] 最小生成树的权值是指生成树所有边的权值之和最小。

72.单选题 [NOIP 普及组 2005] 平面上有五个点 A(5, 3), B(3, 5), C(2, 1), D(3, 3), E(5, 1)。以这五点作为完全图 G 的顶点，每两点之间的直线距离是图 G 中对应边的权值。以下哪条边不是图 G 的最小生成树中的边（ ）。

A. AD

B. BD

C. CD

D. DE

E. EA

73.单选题 [NOIP 提高组 2005] 平面上有五个点 A(5, 3), B(3, 5), C(2, 1), D(3, 3), E(5, 1)。以这五点作为完全图 G 的顶点，每两点之间的直线距离是图 G 中对应边的权值。图 G 的最小生成树中的所有边的权值综合为（ ）。

A. 8

B. 7+√5

C. 9

D. 6+√5

E. 4+2√2 +√5

74.单选题 [GESP202312【8 级】] 一个无向图包含 n 个顶点，则其最小生成树包含多少条边？（ ）。

A.n−1

B.n

C.n+1

D. 最小生成树可能不存在。

75.单选题 [NOIP 普及组 2015/NOIP 提高组 2015] 6 个顶点的连通图的最小生成树，其边数为（ ）。

A. 6

B. 5

C. 7

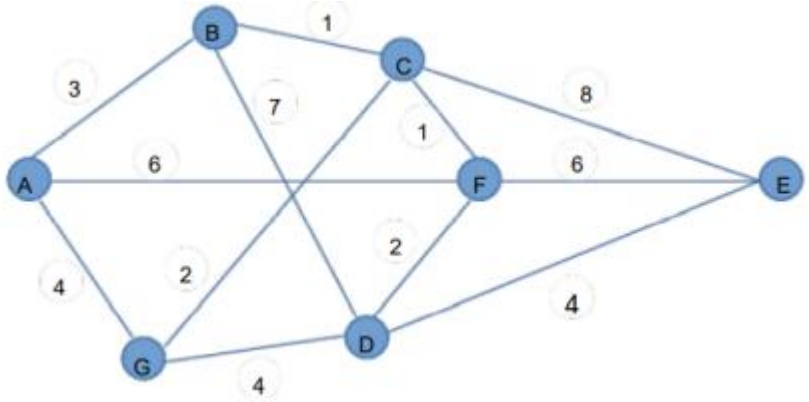
D. 4

7. 图的最短路

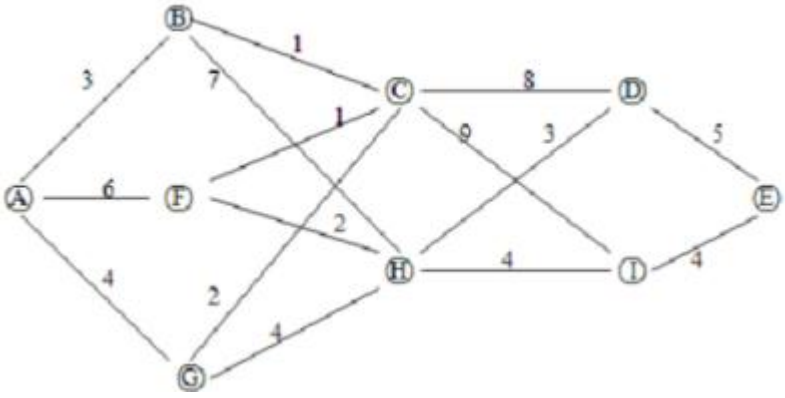
76.填空题 [NOIP 普及组 2008/NOIP 提高组 2008] 有 6 个城市，任何两个城市之间都有一条道路连接，6 个城市两两之间的距离如下表所示，则城市 1 到城市 6 的最短距离为_____。

	城市 1	城市 2	城市 3	城市 4	城市 5	城市 6
城市 1	0	2	3	1	12	15
城市 2	2	0	2	5	3	12
城市 3	3	2	0	3	6	5
城市 4	1	5	3	0	7	9
城市 5	12	3	6	7	0	2
城市 6	15	12	5	9	2	0

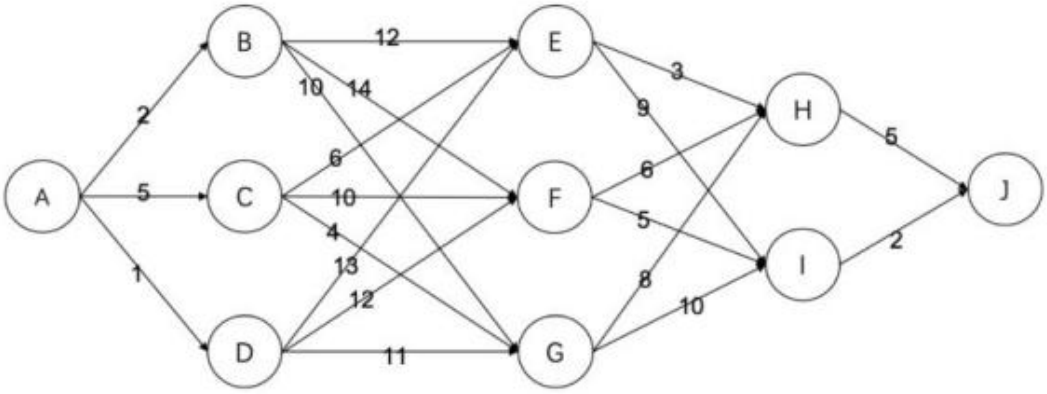
77.填空题 [NOIP 普及组 2014] 如图所示，图中每条边上的数字表示该边的长度，则从 A 到 E 的最短距离是_____。



78.填空题 [NOIP 提高组 2014] 如图所示，图中每条边上的数字表示该边的长度，则从 A 到 E 的最短距离是 _____.



79.单选题 [CSP-S2021] 有如下的有向图，节点为 A,B,⋯,J，其中每条边的长度都标在图中。则节点 A 到节点 J 的最短路径长度为（ ）。



A. 16

B. 19

C. 20

D. 22

80.单选题 [GESP202403【8 级】] 下面的程序使用邻接矩阵表达的带权无向图，则从顶点 0 到顶点 3 的最短距离为（ ）

```
int weight[4][4] = {
        {0, 1, 7, 100},
        {1, 0, 5, 15},
        {7, 5, 0, 6},
        {100, 15, 6, 0}
};
```

A. 100

B. 16

C. 12

D. 13

81.单选题 [GESP202403【7 级】] 下面的程序使用邻接矩阵表达的带权无向图，则从顶点 0 到顶点 3 的最短距离为（ ）

```
int weight[4][4] = {
        {0, 2, 5, 8},
        {2, 0, 1, 7},
        {5, 1, 0, 4},
        {8, 7, 4, 0}
```

```
6 };
```

A. 6

B. 7

C. 8

D. 9

答案

1.AC

2.B

3.C

4.B

5.B

6.A

7.C

8.C

9.D

10.D

11.F

12.F

13.A

14.D

15.B

16.9

17.T

18.F

19.F

20.B

21.D

22.D

23.B

24.T

25.T

26.D

27.A

28.CD

29.B

30.C

31.C

32.C

33.C

34.T

35.6

36.B

37.C

38.AD

39.E

40.T

41.T

42.T

43.F

44.F

45.A

46.C

47.C

48.A

49.B

50.C

51.B

52.A

53.C

54.A

55.4 9

56.D

57.A

58.ABD

59.B

60.A

61.B

62.D

63.B

64.BCE

65.432

66.C

67.D

68.T

69.A

70.A

71.T

72.D

73.D

74.D

75.B

76.7

77.11

78.15

79.B

80.C

81.B

GW 信奥 CSP 初赛习题集 8

其他 CSP-J 冷门考点

1.单选题 [NOIP 普及组 2011] 生物特征识别， 是利用人体本身的生物特征进行身份论证的一种技术。 目前，指纹识别、 虹膜识别、 人脸识别等技术已广泛应用于政府、银行、安全防卫等领域。以下不属于生物 特征识别技术及其应用的是（ ）。

A. 指静脉验证

B. 步态验证

C. ATM 机密码验证

D. 声音验证

2.不定项选择题 [NOIP 提高组 2011] 生物特征识别， 是利用人体本身的生物特征进行身份论证的一种技术。 目前，指纹识别、 虹膜识别、 人脸识别等技术已广泛应用于政府、银行、安全防卫等领域。以下属于生物 特征识别技术及其应用的是（ ）。

A. 指静脉验证

B. 步态验证

C. ATM 机密码验证

D. 声音验证

3.单选题 [NOIP 普及组 2012] 仿生学的问世开辟了独特的科学技术发展道路。人们研究生物体的结构、功能和工作原理，并将这些原理移植于新兴的工程技术之中。以下关于仿生学的叙述，错误的是（ ）。

A. 由研究蝙蝠，发明雷达

B. 由研究蜘蛛网，发明因特网

C. 由研究海豚，发明声纳

D. 由研究电鱼，发明伏特电池

4.单选题 [NOIP 提高组 2012] 仿生学的问世开辟了独特的科学技术发展道路。人们研究生物体的结构、功能和工作原理，并将这些原理移植于新兴的工程技术中。以下关于仿生学的叙述，错误的是（ ）

A. 由研究蝙蝠，发明雷达

B. 由研究蜘蛛网，发明因特网

C. 由研究海豚，发明声纳

D. 由研究电鱼，发明伏特电池

5.单选题 [NOIP 普及组 2010] 在下列 HTML 语句中，可以正确产生一个指向 NOI 官方网站的超链接的是（ ）

A. 欢迎访问 NOI 网站

B. 欢迎访问 NOI 网站

C. <a> http://www.noi.cn

D. 欢迎访问 NOI 网站

6.不定项选择题 [NOIP 提高组 2010] 在下列 HTML 语句中，可以正确产生一个指向 NOI 官方网站的超链接的是（ ）

A. 欢迎访问 NOI 网站

B. 欢迎访问 NOI 网站

C. <a> http://www.noi.cn

D. 欢迎访问 NOI 网站

7.单选题 [NOIP 普及组 2010] Linux 下可执行文件的默认扩展名为（ ）

A. exe

B. com

C. dll

D. 以上都不是

8.单选题 [NOIP 提高组 2013] 把 64 位非零浮点数强制转换成 32 位浮点数后，不可能（ ）。

A. 大于原数

B. 小于原数

C. 等于原数

D. 与原数符号相反

9.单选题 [NOIP 普及组 2013] 把 64 位非零浮点数强制转换成 32 位浮点数后，不可能（ ）。

A. 大于原数

B. 小于原数

C. 等于原数

D. 与原数符号相反

10.单选题 [NOIP 普及组 2008] 在 32*32 点阵的“字库”中，汉字“北”与“京”的字模占用字节数之和是（ ）。

A. 512

B. 256

C. 384

D. 128

11.单选题 [NOIP 普及组 2006] 在编程时（使用任一种高级语言，不一定是 C++），如果需要从磁盘文件中输入一个很大的二维数组（例如 1000*1000 的 double 型数组），按行读（即外层循环是关于行的）与按列读（即外层循环是关于列的）相比，在输入效率上（ ）。

A. 没有区别

B. 按行读的方式要高一些

C. 按列读的方式要高一些

D. 取决于数组的存储方式。

12.单选题 [NOIP 提高组 2006] 在编程时（使用任一种高级语言，不一定是 C++），如果需要从磁盘文件中输入一个很大的二维数组（例如 1000*1000 的 double 型数组），按行读（即外层循环是关于行的）与按列读（即外层循环是关于列的）相比，在输入效率上（ ）。

A. 没有区别

B. 有一些区别，但机器处理速度很快，可忽略不计

C. 按行读的方式要高一些

D. 按列读的方式要高一些

E. 取决于数组的存储方式。

13.单选题 [CSP-J2022] 以下哪种功能没有涉及 C++ 语言的面向对象特性支持：（ ）。

A. C++ 中调用 printf 函数

B. C++ 中调用用户定义的类成员函数

C. C++ 中构造一个 class 或 struct

D. C++ 中构造来源于同一基类的多个派生类

14.填空题 [NOIP 普及组 2010]

```
#include <iostream>
using namespace std;
void swap(int & a, int & b)
{
    int t;
    t = a;
    a = b;
    b = t;
}
int main()
{
    int a1, a2, a3, x;

    cin>>a1>>a2>>a3;
    if (a1 > a2)
```

```
        swap(a1, a2);
    if (a2 > a3)
        swap(a2, a3);
    if (a1 > a2)
        swap(a1, a2);
    cin>>x;
    if (x < a2)
        if (x < a1)
            cout<<x<<' '<<a1<<' '<<a2<<' '<<a3<<endl;
        else
            cout<<a1<<' '<<x<<' '<<a2<<' '<<a3<<endl;
    else
        if (x < a3)
            cout<<a1<<' '<<a2<<' '<<x<<' '<<a3<<endl;
        else
            cout<<a1<<' '<<a2<<' '<<a3<<' '<<x<<endl;
    return 0;
}
```

输入：

91 2 20

77

输出：

15.填空题 [NOIP 普及组 2004] 75 名儿童到游乐场去玩。他们可以骑旋转木马，坐滑行铁道，乘宇宙飞船。已知其中 20 人这三种东西都玩过，55 人至少玩过其中的两种。若每样乘坐一次的费用是 5 元，游乐场总共收入 700，可知有？名儿童没有玩过其中任何一种。

16.填空题 [NOIP 普及组 2005] 有 3 个课外小组：物理组，化学组和生物组。今有张、王、李、赵、陈 5 名同学，已知张、王为物理组成员，张、李、赵为化学组成员，李、赵、陈为生物组成员。如果要在 3 个小组中分别选出 3 位组长，一位同学最多只能担任一个小组的组长，共有？种选择方案。

17.填空题 [NOIP 普及组 2017] （快速幂）请完善下面的程序，该程序使用分治法求 x^p mod m 的值。

输入：三个不超过 10000 的正整数 x, p, m。

输出：x^p mod m 的值。

提示：若 p 为偶数，x^p = (x^2)^(p/2)；若 p 为奇数，x^p = x (x^2)^((p-1)/2)。

```
#include <iostream>
using namespace std;
int x, p, m, i, result;
int main()
{
    cin >> x >> p >> m;
    result = ①;
    while(②)
    {
        if(p % 2 == 1)
            result = ③;
        p /= 2;
        x = ④;
    }
    cout << ⑤ << endl;
    return 0 ;
}
```

18.填空题 [NOIP 提高组 2004] 已知 a, b, c, d, e, f, g 七个人中，a 会讲英语；b 会讲英语和汉语；c 会讲英语、意大利语和俄语；d 会讲汉语和日语；e 会讲意大利语和德语；f 会讲俄语、日语和法语；g 会讲德语和法语。能否将他们的座位安排在圆桌旁，使得每个人都能与他身边的人交谈？如果可以，请以
“a b”开头写出你的安排方案：？

19.填空题 [NOIP 普及组 2012] 在 NOI 期间，主办单位为了欢迎来自各国的选手，举行了盛大的晚宴。在第 十八桌，有 5 名大陆选手和 5 名港澳选手共同进膳。 为了增进交流， 他们决定相 隔就坐，即每个大陆选手左右旁都是港澳选手， 每个港澳选手左右旁都是大陆选 手。那么，这一桌一共有 _____种不同的就坐方案。注：如果在两个方案 中，每个选手左右相邻的选手相同 ，则视为同一种方案。

20.单选题 [NOIP 普及组 2013] 7 个同学围坐一圈，要选 2 个不相邻的作为代表，有____种不同的选法。

21.单选题 [CSP-J2019] 一副纸牌除掉大小王有 52 张牌，四种花色，每种花色 13 张。假设从这 52 张牌中随机抽取 13 张纸牌，则至少（ ）张牌的花色一致。

A. 4

B. 2

C. 3

D. 5

22.单选题 [CSP-J2020] （最小区间覆盖）给出 n 个区间，第 i 个区间的左右端点是[ai,bi]。现在要在这些区间中选出若干个，使得区间 [0,m]被所选区间的并覆盖（即每一个 0≤i≤m 都在某个所选的区间中）。保证答案存在，求所选区间个数的最小值。

输入第一行包含两个整数 n 和 m（1≤n≤5000, 1≤m≤109）。

接下来 n 行，每行两个证书 ai，bi（0≤ai，bi≤m）。

提示：使用贪心法解决这个问题。先用 Θ(n^2) 的时间复杂度排序，然后贪心选择这些区间。

试补全程序。

```
#include <iostream>

using namespace std;

const int MAXN = 5000;
int n, m;
struct segment { int a, b; } A[MAXN];

void sort() // 排序
{
    for (int i = 0; i < n; i++)
        for (int j = 1; j < n; j++)
            if (①)
            {
                segment t = A[j];
                ②
            }
```

```
}

int main()
{
    cin >> n >> m;
    for (int i = 0; i < n; i++)
        cin >> A[i].a >> A[i].b;
    sort();
    int p = 1;
    for (int i = 1; i < n; i++)
        if (③)
            A[p++] = A[i];
    n = p;
    int ans = 0, r = 0;
    int q = 0;
    while (r < m)
    {
        while (④)
            q++;
        ⑤;
        ans++;
    }
    cout << ans << endl;
    return 0;
}
```

1)①处应填（ ）

2)②处应填（ ）

3)③处应填（ ）

4)④处应填（ ）

5)⑤处应填（ ）

1.

A. A[j].b>A[j-1].b

B. A[j].a<A[j-1].a

C. A[j].a>A[j-1].a

D. A[j].b<A[j-1].b

2.

A. A[j+1]=A[j];A[j]=t;

B. A[j-1]=A[j];A[j]=t;

C. A[j]=A[j+1];A[j+1]=t;

D. A[j]=A[j-1];A[j-1]=t;

3.

A. A[i].b>A[p-1].b

B. A[i].b<A[i-1].b

C. A[i].b>A[i-1].b

D. A[i].b<A[p-1].b

4.

A. q+1<n&&A[q+1].a<=r

B. q+1<n&&A[q+1].b<=r

C. q<n&&A[q].a<=r

D. q<n&&A[q].b<=r

5.

A. r=max(r,A[q+1].b)

B. r=max(r,A[q].b)

C. r=max(r,A[q+1].a)

D. q++

23.填空题 [NOIP 普及组 2018] 对于一个 1 到 n 的排列 P（即 1 到 n 中每一个数在 P 中出现了恰好一次），令 qi 为第 i 个位置之后第一个比 Pi 值更大的位置，如果不存在这样的位置，则 qi=n+1。举例来说，如果 n=5 且 P 为 15423，则 q 为 2, 6, 6, 5, 6

下列程序读入了排列 P，使用双向链表求解了答案。试补全程序。

数据范围 1≤n≤10^5。

```
#include <iostream>
using namespace std;
const int N = 100010;
int n;
int L[N], R[N], a[N];
```

```
int main() {
    cin >> n;
    for (int i = 1; i <= n; ++i) {
        int x;
        cin >> x;
        ① ;
    }
    for (int i = 1; i <= n; ++i) {
        R[i] = ② ;
        L[i] = i - 1;
    }
    for (int i = 1; i <= n; ++i) {
        L[ ③ ] = L[a[i]];
        R[L[a[i]]] = R[ ④ ];
    }
    for (int i = 1; i <= n; ++i) {
        cout << ⑤ << "" "";
    }
    cout << endl;
    return 0;
}
```

24.填空题 [NOIP 普及组 2013] （二叉查找树）二叉查找树具有如下性质：每个节点的值都大于其左子树上所有节点的值、小于 其右子树上所有节点的值。试判断一棵树是否为二叉查找树。

输入的第一行包含一个整数 n，表示这棵树有 n 个顶点，编号分别为 1, 2, …, n ，其中编号为 1 的为根结点。之后的第 i 行有三个数 value, left_child, right_child ，分别表示该节点关 键字的值、左子节点的编号、右子节点的编号；如果不存在左子节点或右子节点，则用 0 代替。输 出 1 表示这棵树是二叉查找树，输出 0 则表示不是。

```
#include <iostream>
using namespace std;
const int SIZE = 100;
const int INFINITE = 1000000;
struct node {
    int left_child, right_child, value;
};
node a[SIZE];
int is_bst(int root, int lower_bound, int upper_bound){
```

```
    int cur;
    if (root == 0)return 1;
    cur = a[root].value;
    if ((cur > lower_bound) && ( ① ) &&
    (is_bst(a[root].left_child, lower_bound, cur) == 1) && (is_bst(②,③,④) == 1))
        return 1;
    return 0;
}
int main(){
    int i, n;
    cin>>n;
    for (i = 1; i <= n; i++)
        cin>>a[i].value>>a[i].left_child>>a[i].right_child;
    cout<<is_bst( ⑤ , -INFINITE, INFINITE)<<endl;
    return 0;
}
```

25.单选题 [NOIP 普及组 2013] 将（ 2, 6, 10, 17 ）分别存储到某个地址区间为 0~10 的哈希表中，如果哈希函数 h(x) = （ ），将不会产生冲突，其中 a mod b 表示 a 除以 b 的余数。

A. x mod 11

B. x^2 mod 11

C. 2x mod 11

D. [x] mod 11 ，其中 [x]表示 x 下取整

答案

1.C

2.ABD

3.B

4.B

5.B

6.B

7.D

8.D

9.D

10.B

11.D

12.E

13.A

14.2 20 77 91

15.10

16.11

17.

1. 正确答案： 1
2. 正确答案： p > 0
3. 正确答案： result * x % m
4. 正确答案： x * x % m
5. 正确答案： result

18.abdfgec

19.2880

20.14

21.A

22.BDAAB

23.

1. 正确答案： a[x] = i
2. 正确答案： i + 1
3. 正确答案： R[a[i]]
4. 正确答案： a[i]
5. 正确答案： R[i]

24.

1. 正确答案： cur < upper_bound

2. 正确答案： a[root].right_child
3. 正确答案： cur
4. 正确答案： upper_bound
5. 正确答案： 1

25.D