普及组 CSP-J 2025 初赛模拟卷 1

扫码获取答案解析

一、单项选择题（共 15 题，每题 2 分，共计 30 分；每题有且仅有一个正确选项）

1. 在标准 ASCII 码表中，已知英文字母 c 的 ASCII 码十进制表示是 99，那么英文字母 x 的 ASCII 码十六进制表示是（　　）。

   A. 77　　B. 78　　C. 79　　D. 7A

2. 以下关于 CSP 与 GESP 的描述正确的是（　　）。

   A. CSP-J/CSP-S 属于非专业级别软件能力认证，只有中小学生才能参加

   B. CSP-J/CSP-S 是中国通信学会举办的程序设计竞赛

   C. GESP 是中国电子学会举办的程序设计竞赛

   D. GESP C++ 七级成绩 80 分及以上或者八级成绩 60 分及以上，可以申请免 CSP-J 初赛

3. 以下可以用作 C++程序中的变量名的是（　　）。

   A. `_x1`　　B. `new`　　C. `class`　　D. `public`

4. 以下不属于桌面或者手机操作系统的是（　　）。

   A. Linux　　B. Android　　C. MATLAB　　D. Windows 11

5. C++中使用输入和输出函数 `cin` 和 `cout` 会用到（　　）头文件。

   A. iostream　　B. cmath　　C. cstdio　　D. algorithm

6. 寻找最短路径的广度优先搜索算法经常用到的数据结构是（　　）。

   A. 栈　　B. 链表　　C. 向量　　D. 队列

7. 以下哪个域名后缀不属于中华人民共和国管辖？（　　）

   A. cn　　B. uk　　C. hk　　D. mo

8. 下列排序算法中，平均情况下（　　）算法的时间复杂度最小。

   A. 插入排序　　B. 选择排序　　C. 归并排序　　D. 冒泡排序

9. 关于计算机网络，下面的说法中正确的是（　　）。

A. TCP 是网络层协议

B. 计算机病毒只能通过 U 盘等介质传播，不能通过计算机网络传播

C. 计算机网络可以实现资源共享

D. 公司内部的几台计算机组成的网络规模太小，不能称为计算机网络

10. 序列(7, 5, 1, 12, 3, 6, 9, 4)的逆序对有（　　）个。

A. 15　　B. 12　　C. 13　　D. 14

11. 下列属于图像文件格式的是（　　）。

A. MPEG　　B. DOCX　　C. JPEG　　D. WMV

12. 不管 P、Q 如何取值，以下逻辑表达式中取值恒为假的是（　　）。

A. $(\neg Q \wedge P) \vee (Q \wedge \neg P)$　　B. $((\neg P \vee Q) \vee (P \vee \neg Q)) \wedge P \wedge \neg Q$

C. $\neg P \wedge ((\neg Q \vee P) \vee (Q \vee \neg P)) \wedge P$　　D. $((\neg P \vee Q) \vee (Q \vee \neg P)) \wedge Q \wedge \neg P$

13. 树的根结点的高度为 1，某完全二叉树有 2025 个结点，其高度是（　　）。

A. 10　　B. 11　　C. 12　　D. 13

14. 现有 9 个苹果，要放入 5 个不同的盘子，允许有的盘子中放 0 个苹果，则不同的放法共有（　　）种。

A. 720　　B. 715　　C. 126　　D. 252

15. G 是一个非连通无向图（没有重边和自环），共有 36 条边，则该图至少有（　　）个顶点。

A. 6　　B. 9　　C. 10　　D. 8

二、阅读程序（程序输入不超过数组或字符串定义的范围；判断题正确填√，错误填×；除特殊说明外，判断题每题 1.5 分，选择题每题 3 分，共计 40 分）

（1）

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 using i64 = long long;
```

```
05
06 int popcount(i64 x)
07 {
08     int res = 0;
09     while(x)
10     {
11         if(x & 1 == 1)
12             res++;
13         x >>= 1;
14     }
15     return res;
16 }
17
18 int calc(i64 x)
19 {
20     int sum = 0;
21     for(i64 i = 1; i <= x; i++)
22         sum += popcount(i);
23     return sum;
24 }
25
26 int sum(i64 l, i64 r)
27 {
28     return calc(r) - calc(l);
29 }
30
31 int main()
32 {
33     i64 l, r;
34     cin >> l >> r;
35     cout << calc(l) << ' ' << sum(l, r) << endl;
36     return 0;
37 }
```

■ **判断题**

16. 若程序输入为 5 8，则程序输出 7 6。 ( )
17. 若将第 11 行中的&符号改为^符号，程序输出结果一定不会改变。 ( )
18. 若将头文件#include <bits/stdc++.h>改成#include <stdio.h>，程序仍能正常运行。 ( )

■ 选择题

19. 若输入为 1 12，则输出是什么？（　　）

A. 1 21　　B. 1 20　　C. 1 22　　D. 2 22

20. 程序中的 sum 函数实现了什么功能？（　　）

A. 计算了[l, r]区间内的每个数二进制位上 1 的个数之和

B. 计算了[l, r]区间内的每个数二进制位上 0 的个数之和

C. 计算了(l, r]区间内的每个数二进制位上 1 的个数之和

D. 计算了(l, r]区间内的每个数二进制位上 0 的个数之和

（2）

```
#include <bits/stdc++.h>
using namespace std;

const int inf = 0x3f3f3f3f;

int solve(vector<int> &cur)
{
    int n = cur.size();
    vector<vector<int>> dp(n + 1, vector<int>(n + 1, inf));
    for(int i = 0; i <= n; i++)
       dp[0][i] = dp[i][0] = 0;
    for(int i = 1; i <= n; i++)
       dp[i][i] = cur[i - 1];
    for(int i = 1; i <= n; i++)
       for(int j = 1; j <= n; j++)
          if(i != j)
             dp[i][j]=min(dp[i][j],dp[i-1][j]+dp[i][j-1]);
    int ans = 0;
    for(int i = 1; i <= n; i++)
       ans = max(ans, dp[n][i]);
    return ans;
}

int main()
{
    int n;
    cin >> n;
```

```
28    vector<int> cost(n);
29    for(int i = 0; i < n; i++)
30        cin >> cost[i];
31    cout << solve(cost) << endl;
32    return 0;
33 }
```

■ 判断题

21. 若输入为 3 1 2 3，则输出为 3。（　　）
22. 计算 dp 数组的时间复杂度为 $O(n^2)$。（　　）
23. （2 分）若将第 28 行改为 vector<int> cost(n+1)，则当输入 3 1 2 3 时，solve 函数中的 n=3。（　　）

■ 选择题

24. 当输入的 cost 数组为{4,0,0,5,6}时，程序的输出为（　　）。

A. 23　　B. 25　　C. 24　　D. 22

25. 若将第 17 行改为 dp[i][j] = min(dp[i][j], dp[i-1][j]-dp[i][j-1]);，则当输入的 cost 数组为{4,0,0,5,6}时，程序的输出为（　　）。

A. 20　　B. 21　　C. 22　　D. 23

26. （4 分）当输入的 cost 数组为{4,0,0,5,6}时，在 solve 函数中，dp[2][3]的值为（　　）。

A. 1　　B. 2　　C. 3　　D. 4

（3）

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 int func(int a, int b)
05 {
06    if(a == 0)
07       return b;
08    if(b == 0)
09       return a;
10    return a + func(b, a % b);
11 }
```

```
12
13 int main()
14 {
15     int x, y;
16     cin >> x >> y;
17     cout << func(x, y) << endl;
18     return 0;
19 }
```

假设输入均为非负整数，完成下面的问题。

■ **判断题**

27. 当输入为 2 3 时，程序的输出为 5。（　　）
28. 若输入只有一个为 0，则程序的输出为输入的另一个数字。（　　）
29. 当输入为 6 8 时，func 函数将会被进入 4 次。（　　）

■ **选择题**

30. 当输入为 6 8 时，程序的输出为（　　）。

A. 20　　B. 21　　C. 22　　D. 23

31. 当输入为 3 5 时，func 函数的调用顺序是（　　）。

A. func(3,5)-func(5,3)-func(3,2)-func(2,1)-func(1,0)

B. func(3,5)-func(5,3)-func(3,2)-func(2,1)-func(1,1)-func(1,0)

C. func(3,5)-func(5,2)-func(2,1)-func(1,1)-func(1,0)

D. func(3,5)-func(5,2)-func(2,1)-func(1,0)

32. （4 分）若将第 10 行的代码改为 return a + func(b,a-b)，则当输入为 3 5 时，得到的输出为（　　）。

A. 14　　B. 8

C. 6　　D. 产生未定义行为，结果未知

三、完善程序（单选题，每小题 3 分，共计 30 分）

（1）题目描述：

给定一个整数数组 colors 和一个整数 k，其中 colors 表示一个由红色瓷砖和蓝色

瓷砖组成的环，第 i 块瓷砖的颜色为 colors[i]（1 代表红色，0 代表蓝色）。环中连续 k 块瓷砖的颜色如果是**交替**颜色（除了第一块和最后一块瓷砖以外，中间瓷砖的颜色与它**左边**瓷砖和**右边**瓷砖的颜色都不同），那么它被称为一个交替组。现在，请你找出交替组的个数。

```
01 #include <iostream>
02 #include <①>
03
04 using namespace std;
05
06 int main()
07 {
08     int n, k;
09     cin >> n >> k;
10     vector<int> colors(n);
11     for(int i = 0; i < n; i++)
12         cin >> colors[i];
13     int ans = 0, cnt = ②;
14     for(int i = 0; i < ③; i++)
15     {
16         if(i > 0 && ④)
17             cnt = 0;
18         cnt++;
19         ans += (⑤ && cnt >= k);
20     }
21     cout << ans << endl;
22     return 0;
23 }
```

33. ①处应填（　　）。

A. vector　　B. set　　C. string　　D. map

34. ②处应填（　　）。

A. -1　　B. 0　　C. 1　　D. 2

35. ③处应填（　　）。

A. n　　B. n-1　　C. 2*n　　D. 2*(n-1)

36. ④处应填（　　）。

A. `colors[i] == colors[i-1]`

B. `colors[i] != colors[i-1]`

C. `colors[i % n] == colors[(i-1) % n]`

D. `colors[i % n] != colors[(i-1) % n]`

37. ⑤处应填（　　）。

A. `i > n`　　B. `i >= n`　　C. `i < n`　　D. `i <= n`

（2）题目描述：

在国际象棋中，马的一次移动定义为：垂直移动两个方格后再水平移动一个方格，或者水平移动两个方格后再垂直移动一个方格（两者都形成一个 L 的形状）。

现在，我们有一个马和一个电话垫（如下所示），马只能站在数字单元格上。你可以将马放置在**任何数字单元格**上，然后你应该执行 n-1 次移动来获得长度为 n 的号码。马所有的移动应该是符合规则的有效的移动。马在一次移动的过程中可以经过符号单元格，但必须保证这次移动结束后马站在数字单元格上。

给定一个整数 n，请你计算可以得到多少个长度为 n 的数字串。由于答案可能很大，请你输出答案对 10^9+7 取模后的结果。


```
01 #include <iostream>
02 #include <vector>
03
04 using namespace std;
05
06 const int mod = 1E9 + 7;
07 vector<vector<int>> pos = {{4, 6}, {6, 8}, {7, 9}, {4, 8}, {0, 3, 9},
   ①, {0, 1, 7}, {2, 6}, {1, 3}, {2, 4}};
08
09 int main()
```

```
10 {
11     int n;
12     cin >> n;
13     vector<vector<int>> dp(10, vector<int>(n + 1, 0));
14     for(int i = 0; i < 10; i++)
15         ② = 1;
16     for(int j = 2; j <= n; j++)
17     {
18         for(int i = 0; i < 10; i++)
19         {
20             for(int k = 0; k < pos[i].size(); k++)
21             {
22                 dp[i][j] += dp[③][j - 1];
23                 ④;
24             }
25         }
26     }
27     int ans = 0;
28     for(int i = 0; i < 10; i++)
29     {
30         ⑤;
31         ans %= mod;
32     }
33     cout << ans << endl;
34     return 0;
35 }
```

38. ①处应填（　　）。

A. {1, 3, 7, 9}　　B. {*, #}

C. {2, 8, 0}　　D. {}

39. ②处应填（　　）。

A. dp[i][1]　　B. dp[1][i]

C. dp[i][0]　　D. dp[0][i]

40. ③处应填（　　）。

A. k　　B. pos[k][i]

C. pos[i][k]　　D. pos[i−1][k]

41. ④处应填（　　）。

A. `dp[i][k] %= mod`　　B. `dp[j][i] -= mod`

C. `dp[i][j] %= mod`　　D. `dp[i][j] -= mod`

42. ⑤处应填（　　）。

A. `ans += dp[i][n]`　　B. `ans += dp[i][n-1]`

C. `ans += dp[n][i]`　　D. `ans += dp[n-1][i]`