# 普及组 CSP-J 2025 初赛模拟卷 2

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 C++程序中，假设一个字符占用的内存空间是 1 字节，则下列程序中，s 占用的内存空间是（　　）字节。

```
char s[] = "hello oiers";
size_t cnt = strlen(s);
cout << cnt << endl;
```

A. 10　　B. 11　　C. 13　　D. 12

2. 十六进制数 B2025 转换成八进制数是（　　）。

A. 2620045　　B. 2004526　　C. 729125　　D. 2420045

3. 以下能正确定义二维数组的是（　　）。

A. `int a[3][];`　　B. `int a[][];`

C. `int a[][4];`　　D. `int a[][2] = {{1,2},{1,2},{3,4}};`

4. 二进制[10000011]原码和[10000011]补码表示的十进制数值分别是（　　）。

A. −125，−3　　B. −3，−125　　C. −3，−3　　D. −125，−125

5. 在 C++中，下列定义方式中，变量的值不能被修改的是（　　）。

A. `unsigned int a = 5;`　　B. `static double d = 3.14;`

C. `string s = "ccf csp-j";`　　D. `const char c = 'k';`

6. 走迷宫的深度优先搜索算法经常用到的数据结构是（　　）。

A. 向量　　B. 栈　　C. 链表　　D. 队列

7. 关于递归，以下叙述中正确的是（　　）。

A. 动态规划算法都是用递归实现的

B. 递归比递推更高级，占用的内存空间更少

C. A 函数调用 B 函数，B 函数再调用 A 函数不属于递归的一种

D. 递归是通过调用自身来求解问题的编程技术

8. 以下不属于计算机输入设备的是（　　）。

A. 扫描仪　　B. 显示器　　C. 鼠标　　D. 麦克风

9. 关于排序算法，下面的说法中正确的是（　　）。

A. 快速排序算法在最坏情况下的时间复杂度是 $O(n\log n)$

B. 插入排序算法的时间复杂度是 $O(n\log n)$

C. 归并排序算法的时间复杂度是 $O(n\log n)$

D. 冒泡排序算法是不稳定的

10. 下列关于C++语言的叙述中不正确的是（　　）。

A. 变量没有定义也能使用

B. 变量名不能以数字开头，且中间不能有空格

C. 变量名不能和C++语言中的关键字重复

D. 变量在定义的时候可以不用赋值

11. 如果x和y均为int类型的变量，下列表达式中能正确判断“x等于y”的是(　　)。

A. `(1 == (x / y))`　　B. `(x == (x & y))`

C. `(0 == (x ^ y))`　　D. `(y == (x | y))`

12. 在如今的智能互联网时代，AI如火如荼，除了计算机领域以外，通信领域的技术发展也做出了很大贡献。被称为“通信之父”的是（　　）。

A. 克劳德·香农　　B. 莱昂哈德·欧拉

C. 约翰·冯·诺依曼　　D. 戈登·摩尔

13. 一棵满二叉树的深度为3（根结点的深度为1），按照后序遍历的顺序从1开始编号，根结点的右子结点的编号是（　　）。

A. 3　　B. 6　　C. 7　　D. 5

14. 三头奶牛Bessie、Elise和Nancy去参加考试，考场是连续的6间牛棚，用栅栏隔开。为了防止作弊，任意两头奶牛都不能在相邻的牛棚，则考场安排共有（　　）种不同的方法。

A. 18　　B. 24　　C. 30　　D. 48

15. 为强化安全意识，某学校准备在某消防月连续 10 天内随机抽取 3 天进行消防紧急疏散演习，抽取的 3 天为连续 3 天的概率为（　　）。

A. 3/10　　B. 3/20　　C. 1/15　　D. 1/18

**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 <iostream>
02 #include <vector>
03 #include <algorithm>
04 using namespace std;
05
06 using i64 = long long;
07
08 int clz(i64 x)
09 {
10     for(int i = 0; i != 64; i++)
11         if((x >> (63 - i)) & 1)
12             return i;
13     return 64;
14 }
15
16 bool cmp(i64 x, i64 y)
17 {
18     if(clz(x) == clz(y))
19         return x < y;
20     return clz(x) < clz(y);
21 }
22
23 int main()
24 {
25     int n;
26     cin >> n;
27     vector<int> a(n);
28     for(int i = 0; i < n; i++)
29         cin >> a[i];
30     sort(a.begin(), a.end(), cmp);
31     for(int i = 0; i < n; i++)
```

```
32          cout << a[i] << " \n"[i == n - 1];
33      return 0;
34  }
```

■ **判断题**

16. 若程序输入 5 0 4 2 1 3，则程序输出 4 2 3 1 0。（　　）
17. 若将第 19 行中的 < 换为 >，则当程序输入 5 0 4 2 1 3 时，程序输出为 4 3 2 1 0。（　　）
18. 当调用 cmp(3, 3)时，函数的返回值为 false。（　　）

■ **选择题**

19. 若输入 5 4 2 1 3 1，则输出是什么？（　　）

A. 3 4 2 1 1　　B. 3 2 4 1 1　　C. 4 3 2 1 1　　D. 4 2 3 1 1

20. 这个程序实现了什么功能？（　　）

A. 将输入的数组按照二进制位上从左到右第一个 1 前 0 的个数由多到少进行排序

B. 将输入的数组按照二进制位上从左到右第一个 1 前 0 的个数由少到多进行排序

C. 将输入的数组按照二进制位上从左到右第一个 1 前 0 的个数由多到少进行排序，当 0 的个数相同时，按照原数字由小到大进行排序

D. 将输入的数组按照二进制位上从左到右第一个 1 前 0 的个数由少到多进行排序，当 0 的个数相同时，按照原数字由小到大进行排序

**（2）**

```
01 #include <iostream>
02 #include <vector>
03 #include <algorithm>
04 #include <set>
05 #include <string>
06 using namespace std;
07
08 const int inf = 0x3f3f3f3f;
09
10 int calc(vector<vector<int>> &grid)
11 {
12     int m = grid.size(), n = grid[0].size();
13     vector<vector<int>> dp(m + 1, vector<int>(n + 1, inf));
14    dp[0][0] = grid[0][0];
```

```
15    for(int i = 0; i < m; i++)
16        for(int j = 0; j < n; j++)
17        {
18            if(i > 0)
19                dp[i][j]=min(dp[i][j], dp[i-1][j] + grid[i][j]);
20            if(j > 0)
21                dp[i][j]=min(dp[i][j], dp[i][j-1] + grid[i][j]);
22        }
23    return dp[m - 1][n - 1];
24 }
25
26 int main()
27 {
28     int m, n;
29     cin >> m >> n;
30     vector<vector<int>> a(m, vector<int>(n));
31     for(int i = 0; i < m; i++)
32         for(int j = 0; j < n; j++)
33             cin >> a[i][j];
34     cout << calc(a) << endl;
35     return 0;
36 }
```

假设 m≤100，n≤10000，完成下面的问题。

■ **判断题**

21. 若输入 2 3 1 2 3 4 5 6，则输出为 10。 ( )

22. 计算 dp 数组的时间复杂度为 $O(n^2)$。 ( )

23. （2 分）在 calc 函数中，访问 dp[m][n]不会发生越界错误。 ( )

■ **选择题**

24. 当输入的 a 数组为{{1, 3, 1}, {1, 5, 1}, {4, 2, 1}}时，程序输出为 ( )。

A. 4　　B. 7　　C. 6　　D. 5

25. 若将第 19 行改为 dp[i][j] = min(dp[i][j], dp[i-1][j] - grid[i][j]);，则当输入的 a 数组为{{1, 2, 3}, {4, 5, 6}}时，程序的输出为 ( )。

A. -3　　B. -2　　C. -1　　D. 0

26.（4 分）若将第 10 行中的&符号去除，可能出现什么情况？（　　）

A. dp 数组计算错误　　B. calc 函数中的 grid 数组和 a 数组不一致

C. 无影响　　D. 发生编译错误

**（3）**

```
01 #include <iostream>
02 #include <vector>
03 #include <algorithm>
04 #include <set>
05 #include <string>
06 using namespace std;
07
08 const int N = 1010;
09
10 vector<int> E[N];
11 int V[N];
12 int n;
13
14 void add(int x, int y)
15 {
16    E[x].push_back(y);
17 }
18
19 int gcd(int x, int y)
20 {
21    return !y ? x : gcd(y, x % y);
22 }
23
24 void calc(int cur, int fa)
25 {
26    V[cur] = (gcd(cur, fa) != 1);
27    for(auto v: E[cur])
28    {
29       if(v == fa)
30          continue;
31       calc(v, cur);
32       V[cur] += V[v];
33    }
34    return;
```

```
35 }
36
37 int main()
38 {
39     cin >> n;
40     for(int i = 1; i < n; i++)
41     {
42         int x, y;
43         cin >> x >> y;
44         add(x, y);
45         add(y, x);
46     }
47     calc(1, 1);
48     for(int i = 1; i <= n; i++)
49         cout << V[i] << " \n"[i == n];
50     return 0;
51 }
```

已知 gcd(x, y)的时间复杂度为 $O(\log(\min(x, y)))$，输入中 1≤x,y≤n 且 x!=y，回答下面的问题。

■ 判断题

27. 当输入为 4 1 2 1 3 1 4 时，程序的输出为 0 0 0 0。 ( )
28. gcd 函数用来计算两个数 x 和 y 的最大公约数。 ( )
29. 对于树上的一条边(x, y)，若 x 为 y 的父结点，则必然有 V[x]≤V[y]。 ( )

■ 选择题

30. 当输入为 4 1 2 1 3 2 4 时，程序的输出为 ( )。

A. 0 0 0 0　　B. 1 1 0 1　　C. 0 0 1 0　　D. 1 1 1 1

31. (4 分) calc 函数的时间复杂度为 ( )。

A. $O(n\log n)$　　B. $O(n^2)$　　C. $O(n)$　　D. $O(\log n)$

32. 若将第 32 行中的代码改为 V[cur] *= V[v]，则当输入为 4 1 2 1 3 2 4 时，得到的输出为 ( )。

A. 1 1 1 0　　B. 1 1 0 1　　C. 0 0 1 0　　D. 0 0 0 1

## 三、完善程序（单选题，每小题 3 分，共计 30 分）

**（1）题目描述：**

给定一个长为 n（$1 \leqslant n \leqslant 2 \times 10^5$）的数组 a（$-10^9 \leqslant a[i] \leqslant 10^9$），执行如下操作，直到 a 中只剩下 1 个数：

删除 a[i]。如果 a[i]左右两边都有数字，则把 a[i-1]和 a[i+1]合并成一个数。

输出最后剩下的那个数的最大值。

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 using i64 = long long;
05
06 void solve()
07 {
08     int n, flag = 0, mx = ①;
09     cin >> n;
10     vector<int> a(n + 1);
11     for(int i = 1; i <= n; i++)
12     {
13         cin >> a[i];
14         if(a[i] < 0)
15             flag++;
16         mx = max(mx, a[i]);
17     }
18     if(②)
19     {
20         cout << mx << endl;
21         return;
22     }
23     ③ sum1 = 0, sum2 = 0;
24     for(int i = 1; i <= n; i += 2)
25         sum1 += max(a[i], 0);
26     for(int i = 2; i <= n; i += 2)
27         ④;
28     cout << ⑤ << endl;
29     return;
30 }
31
```

```
32 int main()
33 {
34     int t = 1;
35     cin >> t;
36     while(t--)
37         solve();
38 }
```

33. ①处应填（　　）。

A. 0　　B. 1E9　　C. -1E8　　D. -2E9

34. ②处应填（　　）。

A. flag==n　　B. flag==0　　C. flag!=0　　D. flag!=n

35. ③处应填（　　）。

A. int　　B. i64　　C. i32　　D. unsigned int

36. ④处应填（　　）。

A. sum2 += max(a[i], 0)　　B. sum2 += min(a[i], 0)

C. sum2 -= min(a[i], 0)　　D. sum2 -= max(a[i], 0)

37. ⑤处应填（　　）。

A. min(sum1, sum2)　　B. max(sum1, sum2)

C. sum1 + sum2　　D. sum1 − sum2

**（2）题目描述：**

给定一个字符串 t 和一个字符串列表 s 作为字典。保证 s 中的字符串互不相同，且 t 和 s[i]中均只包含小写英文字母。

如果可以利用字典中出现的一个或多个单词拼接出 t，则返回 true。注意：不要求字典中出现的单词全部使用，并且字典中的单词可以重复使用。

数据限制：1≤t.length()≤300，1≤s.size()≤1000，1≤s[i].length()≤20。

```
01 #include <iostream>
02 #include <vector>
03 #include <algorithm>
04 #include <set>
```

```
#include <string>
using namespace std;

const int N = 1010;

int n, mx, m;
vector<string> s;
vector<int> mem;
string t;
set<string> st;

int dfs(int i)
{
    if(i == 0)
        return 1;
    if(①)
        return mem[i];
    for(int j = i - 1; j >= max(i - mx, 0); j--)
        if(st.find(②) != st.end() && dfs(j))
            return mem[i] = 1;
    return ③;

}

int main()
{
    cin >> n;
    s.resize(n);
    for(int i = 0; i < n; i++)
    {
        cin >> s[i];
        mx = max(mx, ④);
    }
    st = set<string>(s.begin(), s.end());
    cin >> t;
    m = (int)t.length();
    mem.resize(m + 1, -1);
    if(⑤)
        cout << "Yes\n";
    else
```

```
45        cout << "No\n";
46
47    return 0;
48 }
```

38. ①处应填（　　）。

A. mem[i] != -1　　B. mem[i] == -1

C. mem[i] == 0　　D. mem[i] != 0

39. ②处应填（　　）。

A. t.substr(i, j - i)　　B. t.substr(j, i - j)

C. t.substr(j)　　D. t.substr(i)

40. ③处应填（　　）。

A. 1　　B. mem[i] = 1

C. 0　　D. mem[i] = 0

41. ④处应填（　　）。

A. s[i].length()　　B. (int)s[i].length()

C. s[i].length() - 1　　D. (int)s[i].length() - 1

42. ⑤处应填（　　）。

A. !dfs(m)　　B. !dfs(n)　　C. dfs(m)　　D. dfs(n)