

CSP-J 历年初赛真题汇总 及视频解析讲义 (2019 – 2025)

视频解析：陈远龙老师

平台	链接
陈远龙老师网站	https://www.chenyuanlong.com
陈远龙老师微信	陈远龙讲信奥赛真... 江苏 南京  微信号：ichenyuanlong
陈远龙老师抖音	@陈远龙讲信奥赛真题 CSP-J&G... 抖音号：ichenyuanlong 抖音号：@陈远龙讲信奥赛真题 CSP-J&GESP

系统性学习信奥赛知识点、真题解析、答疑、直播、1对1等，可以通过上面方式联系到陈远龙老师本人，内部学习资料，请勿外传

本讲义版本号：v2026.08.12

目录

版权声明 vii

CSP-J 2025 年初赛真题 1

 一、单项选择题..... 1

 题号 CSP-J-2025-1-1..... 1

 题号 CSP-J-2025-1-2 1

 题号 CSP-J-2025-1-3 2

 题号 CSP-J-2025-1-4 2

 题号 CSP-J-2025-1-5 2

 题号 CSP-J-2025-1-6 3

 题号 CSP-J-2025-1-7.....3

 题号 CSP-J-2025-1-8 4

 题号 CSP-J-2025-1-9 4

 题号 CSP-J-2025-1-10 4

 题号 CSP-J-2025-1-11.....5

 题号 CSP-J-2025-1-12 5

 题号 CSP-J-2025-1-13 6

 题号 CSP-J-2025-1-14 6

 题号 CSP-J-2025-1-15 6

 二、阅读程序 8

 题号 CSP-J-2025-2-1 8

 题号 CSP-J-2025-2-2 10

 题号 CSP-J-2025-2-3..... 12

 三、完善程序 14

 题号 CSP-J-2025-3-1 14

 题号 CSP-J-2025-3-2..... 16

CSP-J 2024 年初赛真题 20

 一、单项选择题.....20

 题号 CSP-J-2024-1-1.....20

 题号 CSP-J-2024-1-2 20

 题号 CSP-J-2024-1-3 21

 题号 CSP-J-2024-1-4 21

题号 CSP-J-2024-1-5	21
题号 CSP-J-2024-1-6	22
题号 CSP-J-2024-1-7	22
题号 CSP-J-2024-1-8	22
题号 CSP-J-2024-1-9	23
题号 CSP-J-2024-1-10.....	23
题号 CSP-J-2024-1-11.....	23
题号 CSP-J-2024-1-12.....	24
题号 CSP-J-2024-1-13.....	24
题号 CSP-J-2024-1-14.....	25
题号 CSP-J-2024-1-15.....	25
二、阅读程序	26
题号 CSP-J-2024-2-1	26
题号 CSP-J-2024-2-2.....	28
题号 CSP-J-2024-2-3.....	30
三、完善程序	32
题号 CSP-J-2024-3-1	32
题号 CSP-J-2024-3-2.....	34
CSP-J 2023 年初赛真题	37
一、单项选择题.....	37
题号 CSP-J-2023-1-1.....	37
题号 CSP-J-2023-1-2	37
题号 CSP-J-2023-1-3	38
题号 CSP-J-2023-1-4	38
题号 CSP-J-2023-1-5	39
题号 CSP-J-2023-1-6	39
题号 CSP-J-2023-1-7	40
题号 CSP-J-2023-1-8	40
题号 CSP-J-2023-1-9	40
题号 CSP-J-2023-1-10.....	41
题号 CSP-J-2023-1-11.....	41
题号 CSP-J-2023-1-12.....	42
题号 CSP-J-2023-1-13.....	42
题号 CSP-J-2023-1-14.....	42

题号 CSP-J-2023-1-15	43
二、阅读程序	44
题号 CSP-J-2023-2-1	44
题号 CSP-J-2023-2-2	45
题号 CSP-J-2023-2-3	47
三、完善程序	50
题号 CSP-J-2023-3-1	50
题号 CSP-J-2023-3-2	52
CSP-J 2022 年初赛真题	55
一、单项选择题	55
题号 CSP-J-2022-1-1	55
题号 CSP-J-2022-1-2	55
题号 CSP-J-2022-1-3	56
题号 CSP-J-2022-1-4	56
题号 CSP-J-2022-1-5	57
题号 CSP-J-2022-1-6	57
题号 CSP-J-2022-1-7	57
题号 CSP-J-2022-1-8	58
题号 CSP-J-2022-1-9	58
题号 CSP-J-2022-1-10	59
题号 CSP-J-2022-1-11	59
题号 CSP-J-2022-1-12	59
题号 CSP-J-2022-1-13	60
题号 CSP-J-2022-1-14	60
题号 CSP-J-2022-1-15	61
二、阅读程序	62
题号 CSP-J-2022-2-1	62
题号 CSP-J-2022-2-2	63
题号 CSP-J-2022-2-3	65
三、完善程序	69
题号 CSP-J-2022-3-1	69
题号 CSP-J-2022-3-2	71
CSP-J 2021 年初赛真题	75
一、单项选择题	75

题号 CSP-J-2021-1-1.....	75
题号 CSP-J-2021-1-2	75
题号 CSP-J-2021-1-3	76
题号 CSP-J-2021-1-4	76
题号 CSP-J-2021-1-5	76
题号 CSP-J-2021-1-6	77
题号 CSP-J-2021-1-7	77
题号 CSP-J-2021-1-8	77
题号 CSP-J-2021-1-9	78
题号 CSP-J-2021-1-10	78
题号 CSP-J-2021-1-11	78
题号 CSP-J-2021-1-12	79
题号 CSP-J-2021-1-13	79
题号 CSP-J-2021-1-14	80
题号 CSP-J-2021-1-15	80
二、阅读程序	81
题号 CSP-J-2021-2-1	81
题号 CSP-J-2021-2-2	82
题号 CSP-J-2021-2-3	85
三、完善程序	88
题号 CSP-J-2021-3-1	88
题号 CSP-J-2021-3-2	90
CSP-J 2020 年初赛真题	94
一、单项选择题	94
题号 CSP-J-2020-1-1	94
题号 CSP-J-2020-1-2	94
题号 CSP-J-2020-1-3	95
题号 CSP-J-2020-1-4	95
题号 CSP-J-2020-1-5	95
题号 CSP-J-2020-1-6	96
题号 CSP-J-2020-1-7	97
题号 CSP-J-2020-1-8	97
题号 CSP-J-2020-1-9	97
题号 CSP-J-2020-1-10	98

题号 CSP-J-2020-1-11.....	98
题号 CSP-J-2020-1-12.....	99
题号 CSP-J-2020-1-13.....	99
题号 CSP-J-2020-1-14.....	100
题号 CSP-J-2020-1-15.....	100
二、阅读程序.....	101
题号 CSP-J-2020-2-1.....	101
题号 CSP-J-2020-2-2.....	103
题号 CSP-J-2020-2-3.....	105
三、完善程序.....	108
题号 CSP-J-2020-3-1.....	108
题号 CSP-J-2020-3-2.....	110
CSP-J 2019 年初赛真题.....	114
一、单项选择题.....	114
题号 CSP-J-2019-1-1.....	114
题号 CSP-J-2019-1-2.....	114
题号 CSP-J-2019-1-3.....	115
题号 CSP-J-2019-1-4.....	115
题号 CSP-J-2019-1-5.....	115
题号 CSP-J-2019-1-6.....	116
题号 CSP-J-2019-1-7.....	116
题号 CSP-J-2019-1-8.....	117
题号 CSP-J-2019-1-9.....	117
题号 CSP-J-2019-1-10.....	117
题号 CSP-J-2019-1-11.....	118
题号 CSP-J-2019-1-12.....	118
题号 CSP-J-2019-1-13.....	119
题号 CSP-J-2019-1-14.....	119
题号 CSP-J-2019-1-15.....	119
二、阅读程序.....	121
题号 CSP-J-2019-2-1.....	121
题号 CSP-J-2019-2-2.....	122
题号 CSP-J-2019-2-3.....	124
三、完善程序.....	127

题号 CSP-J-2019-3-1.....	127
题号 CSP-J-2019-3-2	130
CSP-J 历年真题参考答案及视频解析汇总	133
CSP-J 官方大纲（NOI 大纲-入门级）	134
本讲义勘误声明.....	135

版权声明

1. 本讲义所收录的 CSP-J 初赛真题均来源于公开网络，仅为教学研究目的进行整理汇编。相关真题版权归 CSP-J 认证组织方中国计算机学会（CCF）所有。
2. 本讲义中的视频解析均由陈远龙老师原创录制，其著作权及相关权益归陈远龙老师所有。未经授权，不得转载、传播、复制、剪辑、改编或用于其他商业用途。
3. 本讲义中的视频解析、文字解析、解题思路、知识点总结及相关说明均为陈远龙老师个人观点，仅供学习参考，不代表官方立场或官方标准答案。如与官方解释存在差异，请以官方公布的内容为准。
4. 本讲义及相关资料仅供个人学习、研究和交流使用，请勿用于其他用途。
5. 本讲义中的二维码仅用于访问对应题目的视频解析或相关学习资源，是本讲义的重要组成部分。未经授权，请勿对二维码进行篡改、替换或删除。
6. 本讲义电子版（PDF）为免费公益分享，纸质版讲义仅作为课程配套赠品，均不单独进行商业售卖。不得出售、转卖本讲义电子版（PDF）或纸质版，任何形式的售卖、转卖本讲义均属侵权行为。

感谢您的理解与配合，祝同学们学习进步，在 CSP-J 认证中取得优异成绩！

本讲义电子版（PDF）获取渠道：

<https://www.chenyuanlong.com/CSP-J-CSZT>



提示：本讲义若有内容勘误或更新，将发布到上述链接中，建议保存以获取最新版本

CSP-J 2025 年初赛真题

认证时间：2025 年 9 月 20 日 09:30~11:30

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

题号 CSP-J-2025-1-1

1. 一个 32 位无符号整数可以表示的最大值，最接近下列哪个选项？（ ）

- A. 4×10^9
- B. 3×10^{10}
- C. 2×10^9
- D. 2×10^{10}



题号 CSP-J-2025-1-2

2. 在 C++ 中，执行 `int x = 255; cout << (x & (x- 1));` 后，输出的结果是？（ ）

- A. 255
- B. 254
- C. 128
- D. 0



题号 CSP-J-2025-1-3

3. 函数 $\text{calc}(n)$ 的定义如下，则 $\text{calc}(5)$ 的返回值是多少？ ()

```
int calc(int n) {  
    if (n <= 1) return 1;  
    if (n % 2 == 0) return calc(n / 2) + 1;  
    else return calc(n - 1) + calc(n - 2);  
}
```

- A. 5
- B. 6
- C. 7
- D. 8



题号 CSP-J-2025-1-4

4. 用 5 个权值 10, 12, 15, 20, 25 构造哈夫曼树，该树的带权路径长度是多少？ ()

- A. 176
- B. 186
- C. 196
- D. 206



题号 CSP-J-2025-1-5

5. 在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和，这个总和等于？ ()

- A. 顶点数
- B. 边数
- C. 顶点数+边数
- D. 顶点数*2

CSP-J-2025-1-5



视频解析

题号 CSP-J-2025-1-6

6. 从 5 位男生和 4 位女生中选出 4 人组成一个学习小组，要求学习小组中男生和女生都有。有多少种不同的选举方法?()

- A. 126
- B. 121
- C. 120
- D. 100

CSP-J-2025-1-6



视频解析

题号 CSP-J-2025-1-7

7. 假设 a, b, c 都是布尔变量，逻辑表达式 $(a \ \&\& \ b) \ || \ (!c \ \&\& \ a)$ 的值与下列哪个表达式不始终相等? ()

- A. $a \ \&\& \ (b \ || \ !c)$
- B. $(a \ || \ !c) \ \&\& \ (b \ || \ !c) \ \&\& \ (a \ || \ a)$
- C. $a \ \&\& \ (!b \ || \ c)$
- D. $!(a \ || \ !b) \ || \ (a \ \&\& \ !c)$

CSP-J-2025-1-7



视频解析

题号 CSP-J-2025-1-8

8. 已知 $f[0]=1$, $f[1]=1$, 并且对于所有 $n \geq 2$ 有 $f[n]=(f[n-1] + f[n-2]) \% 7$ 那么 $f[2025]$ 的值是多少? ()

- A. 2
- B. 4
- C. 5
- D. 6



题号 CSP-J-2025-1-9

9. 下列关于 C++ string 类的说法, 正确的是? ()

- A. string 对象的长度在创建后不能改变。
- B. 可以使用 + 运算符直接连接一个 string 对象和一个 char 类型的字符。
- C. string 的 length() 和 size() 方法返回的值可能不同。
- D. string 对象必须以 '\0' 结尾, 且这个结尾符计入 length()。



题号 CSP-J-2025-1-10

10. 考虑以下 C++ 函数:

```
void solve(int &a, int b) {  
    a = a + b;  
    b = a - b;  
    a = a - b;  
}  
int main() {
```

```
int x = 5, y = 10;  
solve(x, y);  
}
```

在 main 函数调用 solve 后，x 和 y 的值分别是？（ ）

- A. 5, 10
- B. 10, 5
- C. 10, 10
- D. 5, 5



题号 CSP-J-2025-1-11

11. 一个 8×8 的棋盘，左上角坐标为 (1,1)，右下角为 (8,8)。一个机器人从 (1,1) 出发，每次只能向右或向下走一格。要到达 (4,5)，有多少种不同的路径？（ ）

- A. 20
- B. 35
- C. 56
- D. 70



题号 CSP-J-2025-1-12

12. 某同学用冒泡排序对数组 {6, 1, 5, 2, 4} 进行升序排序，请问需要进行多少次元素交换？（ ）

- A. 5
- B. 6
- C. 7
- D. 8



题号 CSP-J-2025-1-13

13. 十进制数 720_{10} 和八进制数 270_8 的和用十六进制表示是多少? ()

- A. 388_{16}
- B. $3DE_{16}$
- C. 288_{16}
- D. 990_{16}



题号 CSP-J-2025-1-14

14. 一棵包含 1000 个结点的完全二叉树，其叶子结点的数量是多少? ()

- A. 499
- B. 512
- C. 500
- D. 501



题号 CSP-J-2025-1-15

15. 给定一个初始为空的整数栈 S 和一个空的队列 P 。我们按顺序处理输入的整数队列 A : 7, 5, 8, 3, 1, 4, 2。对于队列 A 中的每一个数，执行以下规则：如

果该数是奇数，则将其压入栈 S；如果该数是偶数，且栈 S 非空，则弹出一个栈顶元素，并加入到队列 P 的末尾；如果该数是偶数，且栈 S 为空，则不进行任何操作。当队列 A 中的所有数都处理完毕后，队列 P 的内容是什么？（ ）

- A. 5, 1, 3
- B. 7, 5, 3
- C. 3, 1, 5
- D. 5, 1, 3, 7



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 x;除特殊说

明外，判断题 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2025-2-1

(1)

```
1. #include <algorithm>
2. #include <cstdio>
3. #include <cstring>
4. inline int gcd(int a, int b) {
5.     if (b == 0)
6.         return a;
7.     return gcd(b, a % b);
8. }
9. int main() {
10.    int n;
11.    scanf("%d", &n);
12.    int ans = 0;
13.    for (int i = 1; i <= n; ++i) {
14.        for (int j = i + 1; j <= n; ++j) {
15.            for (int k = j + 1; k <= n; ++k) {
16.                if (gcd(i, j) == 1 && gcd(j, k) == 1
17.                    && gcd(i, k) == 1) {
18.                    ++ans;
19.                }
20.            }
21.        }
22.    }
23.    printf("%d\n", ans);
24.    return 0;
25. }
```

CSP-J-2025-2-1



视频解析

● 判断题

16. (1分) 当输入为 2 时, 程序并不会执行第 16 行的判断语句。()

17. 将第 16 行中的 “`&& gcd(i,k)==1`” 删去不会影响程序运行结果。()

18. 当输入的 $n \geq 3$ 的时候, 程序总是输出一个正整数。()

● 单选题

19. 将第 7 行的 “`gcd(b,a%b)`” 改为 “`gcd(a,a%b)`” 后, 程序可能出现的问题是()。

A. 输出的答案大于原答案。

B. 输出的答案小于原答案。

C. 程序有可能陷入死循环。

D. 可能发生整型溢出问题。

20. 当输入为 8 的时候, 输出为()。

A. 37

B. 42

C. 35

D. 25

21. 调用 `gcd(36,42)` 会返回()。

A. 6

B. 252

C. 3

D. 2

题号 CSP-J-2025-2-2

(2)

```
1. #include <algorithm>
2. #include <cstdio>
3. #include <cstring>
4. #define ll long long
5. int n, k;
6. int a[200007];
7. int ans[200007];
8. int main() {
9.     scanf("%d%d", &n, &k);
10.    for (int i = 1; i <= n; ++i) {
11.        scanf("%d", &a[i]);
12.    }
13.    std::sort(a + 1, a + n + 1);
14.    n = std::unique(a + 1, a + n + 1) - a - 1;
15.    for (int i = 1, j = 0; i <= n; ++i) {
16.        for (; j < i && a[i] - a[j + 1] > k; ++j)
17.            ;
18.        ans[i] = ans[j] + 1;
19.    }
20.    printf("%d\n", ans[n]);
21.    return 0;
22. }
```

CSP-J-2025-2-2



视频解析

● 判断题

22. 当输入为 “3 1 3 2 1” 时,输出结果为 2。()

23. 假设输入的 n 为正整数, 输出的答案一定小于等于 n , 大于等于 1。()

24. 将第 14 行的 “ $n = \text{std::unique}(a+1, a+n+1) - a - 1;$ ” 删去后, 有可能出现

与原本代码不同的输出结果。()

● 单选题

25. 假设输入的 a 数组和 k 均为正整数，执行第 18 行代码时，一定满足的条件不包括()。

- A. $j \leq i$
- B. $a[i] - a[j] > k$
- C. $j \leq n$
- D. $a[j] < a[i]$

26. 当输入的 $n=100$ 、 $k=2$ 、 $a=\{1,2,\dots,100\}$ 时，输出为()。

- A. 34
- B. 100
- C. 50
- D. 33

27. 假设输入的 a 数组和 k 均为正整数，但 a 数组不一定有序，则若误删去第 13 行的 “`std::sort(a+1,a+n+1);`”，程序有可能出现的问题有()。

- A. 输出的答案比原本答案更大
- B. 输出的答案比原本答案更小
- C. 出现死循环行为
- D. 以上均可能发生

题号 CSP-J-2025-2-3

(3)

```
1. #include <algorithm>
2. #include <cstdio>
3. #include <cstring>
4. #define ll long long
5. int f[5007][5007];
6. int a[5007], b[5007];
7. int n;
8. int main() {
9.     scanf("%d", &n);
10.    for (int i = 1; i <= n; ++i) {
11.        scanf("%d", &a[i]);
12.    }
13.    for (int i = 1; i <= n; ++i) {
14.        scanf("%d", &b[i]);
15.    }
16.    for (int i = 1; i <= n; ++i) {
17.        for (int j = 1; j <= n; ++j) {
18.            f[i][j] = std::max(f[i][j], std::max(f[i - 1][j], f[i][j - 1]));
19.            if (a[i] == b[j]) {
20.                f[i][j] = std::max(f[i][j], f[i - 1][j - 1] + 1);
21.            }
22.        }
23.    }
24.    printf("%d\n", f[n][n]);
25.    return 0;
26. }
```

CSP-J-2025-2-3



视频解析

● 判断题

28. 当输入“4 1 2 3 4 1 3 2 2”时，输出为 2。()

29. 当程序运行完毕后，对于所有的 $1 \leq i, j \leq n$ ，都一定有 $f[i][j] \leq f[n][n]$ 。

()

30. 将第 18 行的“ $f[i][j] = \text{std::max}(f[i][j], \text{std::max}(f[i-1][j], f[i][j-1]))$ ；”删去

后，并不影响程序运行结果。()

● 单选题

31. 输出的答案满足的性质有()。

- A. 小于等于 n
- B. 大于等于 0
- C. 不一定大于等于 1
- D. 以上均是

32. 如果在 16 行的循环前加上以下两行：

“`std::sort(a+1,a+n+1);std::sort(b+1,b+n+1)`”，则答案会()。

- A. 变大或不变
- B. 变小或不变
- C. 一定变大
- D. 不变

33. 如果输入的 $a=\{1,2,\cdots,n\}$ ，而且 b 数组中数字均为 $1\sim n$ 中的正整数，则上

述代码等价于下面哪个问题：()。

- A. 求 b 数组去重后的长度
- B. 求 b 数组的最长上升子序列
- C. 求 b 数组的长度
- D. 求 b 数组的最大值

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2025-3-1

(1) (字符串解码) “行程长度编码” (Run-Length Encoding) 是一种无损压缩算法，常用于压缩重复字符较多的数据，以减少存储空间。假设原始字符串不包含数字字符。压缩规则如下：i) 如果原始字符串中一个字符连续出现 N 次 ($N \geq 2$)，在压缩字符串中它被表示为“字符+数字 N ”。例如，编码“A12”代表 12 个连续的字符 A。ii) 如果原始字符串中一个字符只出现 1 次，在压缩字符串中它就表示为该字符本身。例如，编码“B”代表 1 个字符 B。

以下程序实现读取压缩字符串并输出其原始的、解压后的形式。试补全程序。

```
1. #include <cctype>
2. #include <iostream>
3. #include <string>
4. using namespace std;
5.
6. int main() {
7.     string z;
8.     cin >> z;
9.     string s = "";
10.
11.     for (int i = 0; i < z.length(); ) {
12.         char ch = z[i];
13.
14.         if ( ① && isdigit(z[i + 1])) {
```

```

15.         i++;
16.         int count = 0;
17.         while (i < z.length() && isdigit(z[i])) {
18.             count = ②;
19.             i++;
20.         }
21.         for (int j = 0; j < ③; ++j) {
22.             s += ch;
23.         }
24.     } else {
25.         s += ④;
26.         ⑤;
27.     }
28. }
29.
30.     cout << s << endl;
31.     return 0;
32. }

```

CSP-J-2025-3-1



视频解析

33. ①处应填()

- A. $i < z.length()$
- B. $i - 1 \geq 0$
- C. $i + 1 < z.length()$
- D. $isdigit(z[i])$

34. ②处应填()

- A. $count + (z[i] - '0')$
- B. $count * 10 + (z[i] - '0')$
- C. $z[i] - '0'$
- D. $count + 1$

35. ③处应填()

- A. `count - 1`
- B. `count`
- C. `10`
- D. `z[i] - '0'`

36. ④处应填()

- A. `z[i+1]`
- B. `ch`
- C. `z.back()`
- D. `(char)z[i] + 1`

37. ⑤处应填()

- A. `i--`
- B. `i = i + 2`
- C. `i++`
- D. `//` 不执行任何操作

题号 CSP-J-2025-3-2

(2) (精明与糊涂) 有 N 个人，分为两类：i) 精明人：永远能正确判断其他人是精明还是糊涂；ii) 糊涂人：判断不可靠，会给出随机的判断。已知精明人严格占据多数，即如果精明人有 k 个，则满足 $k > N/2$ 。

你只能通过函数 `query(i, j)` 让第 i 个人判断第 j 个人：返回 `true` 表示判断结果

为“精明人”；返回 false 表示判断结果为“糊涂人”。你的目标是，通过这些互相判断，找出至少一个百分之百能确定的精明人。同时，你无需关心 query(i,j)的内部实现。

以下程序利用“精明人占多数”的优势。设想一个“消除”的过程，让人们互相判断并进行抵消。经过若干轮抵消后，最终留下的候选者必然属于多数派，即精明人。

例如，假设有三个人 0、1、2。如果 0 说 1 是糊涂人，而 1 也说 0 是糊涂人，则 0 和 1 至少有一个是糊涂人。程序将同时淘汰 0 和 1。由于三人里至少有两个精明人，我们确定 2 是精明人。

试补全程序。

```
1. #include <iostream>
2. #include <vector>
3. using namespace std;
4.
5. int N;
6. bool query(int i, int j);
7.
8. int main() {
9.     cin >> N;
10.
11.     int candidate = 0;
12.     int count = __①__ ;
13.
14.     for (int i = 1; i < N; ++i) {
15.         if (__②__) {
16.             candidate = i;
17.             count = 1;
18.         } else {
19.             if (__③__) {
```

```

20.         ④ i;
21.     } else {
22.         count++;
23.     }
24. }
25. }
26.
27.     cout << ⑤ << endl;
28.     return 0;
29. }

```

CSP-J-2025-3-2



视频解析

38. ①处应填()

- A. 0
- B. 1
- C. N
- D. -1

39. ②处应填()

- A. count < 0
- B. count == 1
- C. count == 0
- D. query(candidate, i) == false

40. ③处应填()

- A. query(candidate, i) == false
- B. query(i, candidate) == true
- C. query(candidate, i) == false && query(i, candidate) == false
- D. query(candidate, i) == false || query(i, candidate) == false

41. ④处应填()

A. count--

B. break

C. count++

D. candidate = i

42. ⑤处应填()

A. N - 1

B. count

C. candidate

D. 0

CSP-J 2024 年初赛真题

认证时间：2024 年 9 月 21 日 09:30~11:30，共 2 小时

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

题号 CSP-J-2024-1-1

1. 32 位 int 存储范围是 ()

- A. $-2147483647 \sim +2147483647$
- B. $-2147483647 \sim +2147483648$
- C. $-2147483648 \sim +2147483647$
- D. $-2147483648 \sim +2147483648$



题号 CSP-J-2024-1-2

2. 计算 $(14_8 - 1010_2) * D_{16} - 1101_2$ 的结果，并选择答案的十进制值： ()

- A. 13
- B. 14
- C. 15
- D. 16



题号 CSP-J-2024-1-3

3. 某公司有 10 名员工，分为 3 个部门：A 部门有 4 名员工，B 部门有 3 名员工、C 部门有 3 名员工。现需要从这 10 名员工中选出 4 名组成一个工作组，且每个部门至少要有 1 人。问有多少种选择方式？（ ）

- A. 120
- B. 126
- C. 132
- D. 238



题号 CSP-J-2024-1-4

4. 以下哪个序列对应数组 0 至 8 的 4 位二进制格雷码（Gray code）？
（ ）

- A. 0000, 0001, 0011, 0010, 0110, 0111, 0101, 1000
- B. 0000, 0001, 0011, 0010, 0110, 0111, 0100, 0101
- C. 0000, 0001, 0011, 0010, 0100, 0101, 0111, 0110
- D. 0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100



题号 CSP-J-2024-1-5

5. 记 1KB 为 1024 字节（byte），1MB 为 1024KB，那么 1MB 是多少二进制位（bit）？（ ）

- A. 1000000
- B. 1048576
- C. 8000000
- D. 8388608

CSP-J-2024-1-5



视频解析

题号 CSP-J-2024-1-6

6. 以下哪个不是 C++ 中的基本数据类型？（ ）

- A. int
- B. float
- C. struct
- D. char

CSP-J-2024-1-6



视频解析

题号 CSP-J-2024-1-7

7. 以下哪个不是 C++ 中的循环语句？（ ）

- A. for
- B. while
- C. do-while
- D. repeat-untill

CSP-J-2024-1-7



视频解析

题号 CSP-J-2024-1-8

8. 在 C/C++ 中，`(char)('a'+13)` 与下面的哪一个值相等（ ）

- A. 'm'

CSP-J-2024-1-8



视频解析

- B. 'n'
- C. 'z'
- D. '3'

题号 CSP-J-2024-1-9

9. 假设有序表中有 1000 个元素，则用二分法查找元素 x 最多需要比较
() 次

- A. 25
- B. 10
- C. 7
- D. 1



题号 CSP-J-2024-1-10

10. 下面哪一个不是操作系统名字 ()

- A. Notepad
- B. Linux
- C. Windows
- D. macOS



题号 CSP-J-2024-1-11

11. 在无向图中，所有顶点的度数之和等于 ()

- A. 图的边数



- B. 图的边数的两倍
- C. 图的定点数
- D. 图的定点数的两倍

题号 CSP-J-2024-1-12

12. 已知二叉树的前序遍历为[A,B,D,E,C,F,G],中序遍历为[D,B,E,A,F,C,G],求二叉树 的后序遍历的结果是 ()

- A. [D,E,B,F,G,C,A]
- B. [D,E,B,F,G,A,C]
- C. [D,B,E,F,G,C,A]
- D. [D,E,B,F,G,A,C]



题号 CSP-J-2024-1-13

13. 给定一个空栈，支持入栈和出栈操作。若入栈操作的元素依次是 1 2 3 4 5 6，其中 1 最先入栈，6 最后入栈，下面哪种出栈顺序是不可能的 ()

- A. 6 5 4 3 2 1
- B. 1 6 5 4 3 2
- C. 2 4 6 5 3 1
- D. 1 3 5 2 4 6



题号 CSP-J-2024-1-14

14. 有 5 个男生和 3 个女生站成一排，规定 3 个女生必须相邻，问有多少种不同的排列方式？（ ）

- A. 4320 种
- B. 5040 种
- C. 3600 种
- D. 2880 种



题号 CSP-J-2024-1-15

15. 编译器的主要作用是什么（ ）？

- A. 直接执行源代码
- B. 将源代码转换为机器代码
- C. 进行代码调试
- D. 管理程序运行时的内存



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 1.5 分，选择题 3 分，共计 40 分。

题号 CSP-J-2024-2-1

(1)

```
1. #include <iostream>
2. using namespace std;
3.
4. bool isPrime(int n) {
5.     if (n <= 1) {
6.         return false;
7.     }
8.     for (int i = 2; i * i <= n; i++) {
9.         if (n % i == 0) {
10.            return false;
11.        }
12.    }
13.    return true;
14. }
15.
16. int countPrimes(int n) {
17.     int count = 0;
18.     for (int i = 2; i <= n; i++) {
19.         if (isPrime(i)) {
20.             count++;
21.         }
22.     }
23.     return count;
24. }
25.
26. int sumPrimes(int n) {
27.     int sum = 0;
28.     for (int i = 2; i <= n; i++) {
29.         if (isPrime(i)) {
```

```

30.         sum += i;
31.     }
32. }
33.     return sum;
34. }
35.
36. int main() {
37.     int x;
38.     cin >> x;
39.     cout << countPrimes(x) << " " << sumPrimes(x) << endl;
40.     return 0;
41. }

```



● 判断题

16. 当输入为“10”时，程序的第一个输出为“4”，第二个输出为“17”。()

17. 若将 isPrime(i)函数种的条件改为 $i \leq n/2$ ，输入“20”时，

countPrimes(20)的输出将变为“6”()

18. sumPrimes 函数计算的是从 2 到 n 之间的所有素数之和()

● 单选题

19. 当输入为“50”时，sumPrimes(50)的输出为()

A. 1060

B. 328

C. 381

D. 275

20. 如果将 for(int i=2;i*i<=n;i++)改为 for(int i=2;i<=n;i++)，输入“10”

时，程序的输出()

A. 将不能正确计算 10 以内素数个数及其和

- B. 仍然输出 “4” 和 “17”
- C. 输出 “3” 和 10
- D. 输出结果不变，但运行时间更短

题号 CSP-J-2024-2-2

(2)

```
1. #include <iostream>
2. #include <vector>
3. using namespace std;
4.
5. int compute(vector<int>& cost) {
6.     int n = cost.size();
7.     vector<int> dp(n+1, 0);
8.     dp[1] = cost[0];
9.     for (int i = 2; i <= n; i++) {
10.         dp[i] = min(dp[i-1], dp[i-2]) + cost[i-1];
11.     }
12.     return min(dp[n], dp[n-1]);
13. }
14.
15. int main() {
16.     int n;
17.     cin >> n;
18.     vector<int> cost(n);
19.     for (int i = 0; i < n; i++) {
20.         cin >> cost[i];
21.     }
22.     cout << compute(cost) << endl;
23.     return 0;
24. }
```

CSP-J-2024-2-2



视频解析

● 判断题

21. 当输入的 `cost` 数组为{10, 15, 20}时, 程序的输出为 15 ()

22. 如果将 `dp[i-1]`改为 `dp[i-3]`, 程序可能会产生编译错误 ()

23. (2 分) 程序总是输出 `cost` 数组中的最小的元素 ()

● 单选题

24. 当输入的 `cost` 数组为{1, 100, 1, 1, 1, 100, 1, 1, 100 ,1}时, 程序的输出为 ()

A. 6

B. 7

C. 8

D. 9

25. (4 分) 如果输入的 `cost` 数组为{10, 15, 30, 5, 5, 10, 20}, 程序的输出为()

A. "25"

B. "30"

C. "35"

D. "40"

26. 若将代码中的 `min(dp[i-1],dp[i-2])+cost[i-1]`修改为 `dp[i-1]+cost[i-2]`, 输入 `cost` 数组为{5, 10, 15}时, 程序的输出为()

A. 10

B. 15

C. 20

D. 25

题号 CSP-J-2024-2-3

(3)

```
1. #include <iostream>
2. #include <cmath>
3. using namespace std;
4.
5. int customFunction(int a, int b) {
6.     if (b == 0) {
7.         return a;
8.     }
9.     return a + customFunction(a, b-1);
10. }
11.
12. int main() {
13.     int x, y;
14.     cin >> x >> y;
15.     int result = customFunction(x, y);
16.     cout << pow(result, 2) << endl;
17.     return 0;
18. }
```

CSP-J-2024-2-3



视频解析

● 判断题

27. 当输入为“2 3”时，customFunction(2,3)的返回值为“64”。()

28. 当 b 为负数时，customFunction(a,b)会陷入无限递归。()

29. 当 b 的值越大，程序的运行时间越长。()

● 单选题

30. 当输入为 “5 4” 时, customFunction(5,4)的返回值为()

- A. 5
- B. 25
- C. 250
- D. 625

31. 如果输入 $x = 3$ 和 $y = 3$, 则程序的最终输出为()

- A. "27"
- B. "81"
- C. "144"
- D. "256"

32. (4 分) 若将 customFunction 函数改为 “return a + customFunction(a-1, b-1); 并输入 “3 3”, 则程序的最终输出为()。

- A. 9
- B. 16
- C. 25
- D. 36

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2024-3-1

(1) (判断平方数) 问题：给定一个正整数 n ，判断这个数是否为完全平方数，即存在一个正整数 x 使得 x 的平方等于 n

试补全程序。

```
1. #include<iostream>
2. #include<vector>
3. using namespace std;
4. bool isSquare(int num) {
5.     int i = ①;
6.     int bound = ②;
7.     for (; i <= bound; ++i) {
8.         if (③) {
9.             return ④;
10.        }
11.    }
12.    return ⑤;
13. }
14. int main() {
15.     int n;
16.     cin >> n;
17.     if (isSquare(n)) {
18.         cout << n << " is a Square number" << endl;
19.     } else {
20.         cout << n << " is not a Square number" << endl;
21.     }
22.     return 0;
23. }
```

33. ①处应填 ()

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



34. ②处应填 ()

- A. `(int) floor(sqrt(num))-1`
- B. `(int)floor(sqrt(num))`
- C. `floor(sqrt(num/2))-1`
- D. `floor(sqrt(num/2))`

35. ③处应填 ()

- A. `num = 2*i`
- B. `num == 2*i`
- C. `num = i*i`
- D. `num == i*i`

36. ④处应填 ()

- A. `num = 2*i`
- B. `num == 2*i`
- C. `true`
- D. `false`

37. ⑤处应填 ()

- A. num = i*i
- B. num != 2*i
- C. true
- D. false

题号 CSP-J-2024-3-2

(2) (汉诺塔问题) 给定三根柱子，分别标记为 A、B 和 C。初始状态下，柱子 A 上有若干个圆盘，这些圆盘从上到下按从小到大的顺序排列。任务是将这些圆盘全部移到柱子 C 上，且必须保持原有顺序不变。在移动过程中，需要遵守以下规则：

1. 只能从一根柱子的顶部取出圆盘，并将其放入另一根柱子的顶部。
2. 每次只能移动一个圆盘
3. 小圆盘必须始终在大圆盘之上。

试补全程序。

```
1. #include <bits/stdc++.h>
2. using namespace std;
3. void move(char src, char tgt) {
4.     cout << "从柱子" << src << "挪到柱子上" << tgt << endl;
5. }
6. void dfs(int i, char src, char tmp, char tgt) {
7.     if (i == ①) {
8.         move(②);
9.         return;
10.    }
11.    dfs(i - 1, ③);
```

```
12.     move(src, tgt);
13.     dfs(⑤, ④);
14. }
15. int main() {
16.     int n;
17.     cin >> n;
18.     dfs(n, 'A', 'B', 'C');
19.     return 0;
20. }
```

CSP-J-2024-3-2



视频解析

38. ①处应填 ()

- A. 0
- B. 1
- C. 2
- D. 3

39. ②处应填 ()

- A. src, tmp
- B. src, tgt
- C. tmp, tgt
- D. tgt, tmp

40. ③处应填 ()

- A. src, tmp, tgt
- B. src, tgt, tmp
- C. tgt, tmp, src
- D. tgt, src, tmp

41. ④处应填 ()

- A. src, tmp, tgt
- B. tmp, src, tgt
- C. src, tgt, tmp
- D. tgt, src, tmp

42. ⑤处应填 ()

- A. 0
- B. 1
- C. i - 1
- D. i

CSP-J 2023 年初赛真题

认证时间：2023 年 9 月 16 日 09:30~11:30，共 2 小时

一、单项选择题

共 15 题，每题 2 分，共计 30 分；每题有且有一个正确选项

题号 CSP-J-2023-1-1

1. 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



题号 CSP-J-2023-1-2

2. 八进制数 12345670_8 和 07654321_8 的和为()。

- A. 22222221_8
- B. 21111111_8
- C. 22111111_8
- D. 22222211_8



题号 CSP-J-2023-1-3

3. 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;



题号 CSP-J-2023-1-4

4. 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的节点，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;`

题号 CSP-J-2023-1-5

5. 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



题号 CSP-J-2023-1-6

6. 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



题号 CSP-J-2023-1-7

7. 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。

CSP-J-2023-1-7



视频解析

题号 CSP-J-2023-1-8

8. 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. $6 - ((2 + 3) * (3 + 8 / 2)) ^ 2 + 3$

CSP-J-2023-1-8



视频解析

题号 CSP-J-2023-1-9

9. 数 101010_2 和 166_8 的和为()

- A. 10110000_2
- B. 236_8
- C. 158_{10}

CSP-J-2023-1-9



视频解析

D. $A0_{16}$

题号 CSP-J-2023-1-10

10. 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 a,b,c,d,e,f 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



题号 CSP-J-2023-1-11

11. 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?()

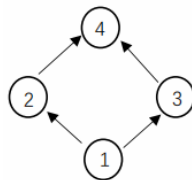
- A. EDBFGCA
- B. EDBGFCA
- C. DEBGFCA
- D. DBEGFCA



题号 CSP-J-2023-1-12

12. 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



题号 CSP-J-2023-1-13

13. 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)



题号 CSP-J-2023-1-14

14. 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

题号 CSP-J-2023-1-15

15. 下哪个不是操作系统?()

A. Linux

B. Windows

C. Android

D. HTML



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 x；除特殊

说明外，判断题 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2023-2-1

(1)

```
1. #include <iostream>
2. #include <cmath>
3. using namespace std;
4.
5. double f(double a, double b, double c) {
6.     double s = (a + b + c) / 2;
7.     return sqrt(s * (s - a) * (s - b) * (s - c));
8. }
9.
10. int main() {
11.     cout.flags(ios::fixed);
12.     cout.precision(4);
13.
14.     int a, b, c;
15.     cin >> a >> b >> c;
16.     cout<< f(a, b, c) <<endl;
17.     return 0;
18. }
```



假设输入的所有数都为不超过 1000 的正整数，完成下面的判断题和单选题

判断题：

● 判断题

16. (2 分) 当输入为 “2 2 2” 时，输出为 “1.7321”。()

17. (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。()

18. (2 分) 程序总是输出四位小数。()

● 单选题

19. 当输入为“3 4 5”时，输出为()。

- A. "6.0000"
- B. "12.0000"
- C. "24.0000"
- D. "30.0000"

20. 当输入为“5 12 13”时，输出为()

- A. "24.0000"
- B. "30.0000"
- C. "60.0000"
- D. "120.0000"

题号 CSP-J-2023-2-2

(2)

```
1. #include <iostream>
2. #include <vector>
3. #include <algorithm>
4. using namespace std;
5.
```

```

6. int f(string x, string y) {
7.     int m = x.size();
8.     int n = y.size();
9.     vector<vector<int>> v(m+1,vector<int>(n+1,0));
10.    for(int i = 1; i <= m; i++) {
11.        for(int j = 1; j <= n; j++) {
12.            if(x[i-1] == y[j-1]) {
13.                v[i][j] = v[i-1][j-1] + 1;
14.            } else {
15.                v[i][j] = max(v[i-1][j], v[i][j-1]);
16.            }
17.        }
18.    }
19.    return v[m][n];
20. }
21.
22. bool g(string x, string y) {
23.     if(x.size() != y.size()) {
24.         return false;
25.     }
26.     return f(x + x, y) == y.size();
27. }
28.
29. int main() {
30.     string x, y;
31.     cin >> x >> y;
32.     cout << g(x, y) << endl;
33.     return 0;
34. }

```

CSP-J-2023-2-2



视频解析

● 判断题

21. f 函数的返回值小于等于 $\min(n,m)$ 。 ()
22. 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。 ()
23. 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

24. 将第 19 行中的 “v[m][n]” 替换为 “v[n][m]”，那么该程序()。

- 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

25. 当输入为 “csp-j p-jcs” 时，输出为()。

- A. “0”
- B. “1”
- C. “T”
- D. “F”

26. 当输入为 “csppsc spsccp” 时，输出为()。

- A. “T”
- B. “F”
- C. “0”
- D. “1”

题号 CSP-J-2023-2-3

(3)

```
1. #include <iostream>
```

```

2. #include <cmath>
3. using namespace std;
4.
5. int solve1(int n) {
6.     return n * n;
7. }
8.
9. int solve2(int n) {
10.    int sum = 0;
11.    for(int i = 1; i <= sqrt(n); i++) {
12.        if(n % i == 0) {
13.            if(n/i == i) {
14.                sum += i*i;
15.            } else {
16.                sum += i*i + (n/i)*(n/i);
17.            }
18.        }
19.    }
20.    return sum;
21. }
22.
23. int main() {
24.    int n;
25.    cin >> n;
26.    cout<<solve2(solve1(n))<<" "<<solve1(solve2(n))<<endl;
27.    return 0;
28. }

```



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

27. 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。 ()

28. 第 13-14 行的作用是避免 n 的平方根因子(或 n/i)进入第 16 行而被计算两次。 ()

29. 如果输入的 n 为质数，`solve2(n)`的返回值为 n^2+1 。 ()

● 单选题

30. (4 分)如果输入的 n 为质数 p 的平方, 那么 $\text{solve2}(n)$ 的返回值为 ()。

- A. $p^2 + p + 1$
- B. $n^2 + n + 1$
- C. $n^2 + 1$
- D. $p^4 + 2p^2 + 1$

31. 当输入为正整数时, 第一项减去第二项的差值一定()。

- A. 大于等于 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

32. 当输入为 “5” 时, 输出为()。

- A. “651 625”
- B. “650 729”
- C. “651 676”
- D. “652 625”

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2023-3-1

(1)（寻找被移除的元素）问题：原有长度为 $n+1$ 、公差为 1 的等差升序数列；将数列输入到程序的数组时移除了一个元素，导致长度为 n 的升序数组可能不再连续，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。

```
1. #include <iostream>
2. #include <vector>
3.
4. using namespace std;
5.
6. int find_missing(vector<int>& nums) {
7.     int left = 0, right = nums.size() - 1;
8.     while (left < right) {
9.         int mid = left + (right - left) / 2;
10.        if (nums[mid] == mid + ①) {
11.            ②;
12.        } else {
13.            ③;
14.        }
15.    }
16.    return ④;
17. }
18.
19. int main() {
20.     int n;
21.     cin >> n;
22.     vector<int> nums(n);
23.     for (int i = 0; i < n; i++) cin >> nums[i];
24.     int missing_number = find_missing(nums);
25.     if (missing_number == ⑤) {
26.         cout << "Sequence is consecutive" << endl;
```

```
27.     } else {  
28.         cout << "Missing number is " << missing_number << endl;  
29.     }  
30.     return 0;  
31. }
```

33. ①处应该填()

- A. 1
- B. nums[0]
- C. right
- D. left



34. ②处应该填()

- A. left = mid + 1
- B. right = mid - 1
- C. right = mid
- D. left = mid

35. ③处应该填()

- A. left = mid + 1
- B. right = mid - 1
- C. right = mid
- D. left = mid

36. ④处应该填()

- A. left + nums[0]
- B. right + nums[0]

C. mid + nums[0]

D. right + 1

37. ⑤处应该填()

A. nums[0]+n

B. nums[0]+n-1

C. nums[0]+n+1

D. nums[n-1]

题号 CSP-J-2023-3-2

(2) (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)个字符，求将第一个字符串转换为第二个字符串所需
的最少操作次数。

试补全动态规划算法。

```
1. #include <iostream>
2. #include <string>
3. #include <vector>
4. using namespace std;
5.
6. int min(int x, int y, int z) {
7.     return min(min(x, y), z);
8. }
9.
10. int edit_dist_dp(string str1, string str2) {
11.     int m = str1.length();
12.     int n = str2.length();
13.     vector<vector<int>> dp(m + 1, vector<int>(n + 1));
14.
15.     for (int i = 0; i <= m; i++) {
16.         for (int j = 0; j <= n; j++) {
```

```

17.         if (i == 0)
18.             dp[i][j] = ①;
19.         else if (j == 0)
20.             dp[i][j] = ②;
21.         else if (③)
22.             dp[i][j] = ④;
23.         else
24.             dp[i][j] = 1 + min(dp[i][j - 1], dp[i - 1][j], ⑤);
25.     }
26. }
27. return dp[m][n];
28. }
29.
30. int main() {
31.     string str1, str2;
32.     cin >> str1 >> str2;
33.     cout << "Mininum number of operation:"
34.         << edit_dist_dp(str1, str2) << endl;
35.     return 0;
36. }

```

CSP-J-2023-3-2



视频解析

38. ①处应该填()

- A. j
- B. i
- C. m
- D. n

39. ②处应该填()

- A. j
- B. i
- C. m
- D. n

40. ③处应该填()

- A. $\text{str1}[i - 1] == \text{str2}[j - 1]$
- B. $\text{str1}[i] == \text{str2}[j]$
- C. $\text{str1}[i - 1] != \text{str2}[j - 1]$
- D. $\text{str1}[i] != \text{str2}[j]$

41. ④处应该填()

- A. $\text{dp}[i - 1][j - 1] + 1$
- B. $\text{dp}[i - 1][j - 1]$
- C. $\text{dp}[i - 1][j]$
- D. $\text{dp}[i][j - 1]$

42. ⑤处应该填()

- A. $\text{dp}[i][j] + 1$
- B. $\text{dp}[i - 1][j - 1] + 1$
- C. $\text{dp}[i - 1][j - 1]$
- D. $\text{dp}[i][j]$

CSP-J 2022 年初赛真题

认证时间：2022 年 9 月 18 日 09:30~11:30

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

题号 CSP-J-2022-1-1

1. 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



题号 CSP-J-2022-1-2

2. 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



题号 CSP-J-2022-1-3

3. 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



题号 CSP-J-2022-1-4

4. 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



题号 CSP-J-2022-1-5

5. 对假设栈 S 和队列 Q 的初始状态为空。存在 $e_1 \sim e_6$ 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 e_1 、 e_2 、 e_3 、 e_4 、 e_5 和 e_6 进栈，队列 Q 依次有数据 e_2 、 e_4 、 e_3 、 e_6 、 e_5 和 e_1 出队列。则栈 S 的容量至少是（ ）个数据。

- A. 2
- B. 3
- C. 4
- D. 6



题号 CSP-J-2022-1-6

6. 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. $abc-+d$



题号 CSP-J-2022-1-7

7. 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

的编码长度为 () 位。

- A. 1
- B. 2
- C. 2 或 3
- D. 3



题号 CSP-J-2022-1-8

8. 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



题号 CSP-J-2022-1-9

9. 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

题号 CSP-J-2022-1-10

10. 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. 栈的访问原则为后进先出，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。



题号 CSP-J-2022-1-11

11. 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. $p \rightarrow next \rightarrow prev = s$; $s \rightarrow prev = p$; $p \rightarrow next = s$; $s \rightarrow next = p \rightarrow next$;

B. $p \rightarrow next \rightarrow prev = s$; $p \rightarrow next = s$; $s \rightarrow prev = p$; $s \rightarrow next = p \rightarrow next$;

C. $s \rightarrow prev = p$; $s \rightarrow next = p \rightarrow next$; $p \rightarrow next = s$; $p \rightarrow next \rightarrow prev = s$;

D. $s \rightarrow next = p \rightarrow next$; $p \rightarrow next \rightarrow prev = s$; $s \rightarrow prev = p$; $p \rightarrow next = s$;



题号 CSP-J-2022-1-12

12. 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



题号 CSP-J-2022-1-13

13. 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



题号 CSP-J-2022-1-14

14. 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



题号 CSP-J-2022-1-15

15. 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



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 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2022-2-1

(1)

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     unsigned short x, y;
8.     cin >> x >> y;
9.     x = (x | x << 2) & 0x33;
10.    x = (x | x << 1) & 0x55;
11.    y = (y | y << 2) & 0x33;
12.    y = (y | y << 1) & 0x55;
13.    unsigned short z = x | y << 1;
14.    cout << z << endl;
15.    return 0;
16. }
```

假设输入的 x 、 y 均是不超过 15 的自然数，完成下面的

题：

● 判断题

16. 删去第 7 行与第 13 行的 `unsigned`，程序行为不变。（ ）

17. 将第 7 行与第 13 行的 `short` 均改为 `char`，程序行为不变。（ ）

CSP-J-2022-2-1



视频解析

18. 程序总是输出一个整数“0”。()
19. 当输入为“2 2”时，输出为“10”。()
20. 当输入为“2 2”时，输出为“59”。()

● 单选题

21. 当输入为“13 8”时，输出为()。
- A. “0”
- B. “209”
- C. “197”
- D. “226”

题号 CSP-J-2022-2-2

(2)

```
1. #include <algorithm>
2. #include <iostream>
3. #include <limits>
4.
5. using namespace std;
6.
7. const int MAXN = 105;
8. const int MAXK = 105;
9.
10. int h[MAXN][MAXK];
11.
12. int f(int n, int m)
13. {
14.     if (m == 1) return n;
15.     if (n == 0) return 0;
16.
17.     int ret = numeric_limits<int>::max();
```

```

18.     for (int i = 1; i <= n; i++)
19.         ret = min(ret, max(f(n - i, m), f(i - 1, m - 1)) + 1);
20.     return ret;
21. }
22.
23. int g(int n, int m)
24. {
25.     for (int i = 1; i <= n; i++)
26.         h[i][1] = i;
27.     for (int j = 1; j <= m; j++)
28.         h[0][j] = 0;
29.
30.     for (int i = 1; i <= n; i++) {
31.         for (int j = 2; j <= m; j++) {
32.             h[i][j] = numeric_limits<int>::max();
33.             for (int k = 1; k <= i; k++)
34.                 h[i][j] = min(
35.                     h[i][j],
36.                     max(h[i - k][j], h[k - 1][j - 1]) + 1);
37.         }
38.     }
39.
40.     return h[n][m];
41. }
42.
43. int main()
44. {
45.     int n, m;
46.     cin >> n >> m;
47.     cout << f(n, m) << endl << g(n, m) << endl;
48.     return 0;
49. }

```



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：

题：

● 判断题

22. 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。（ ）

23. 输出的两行整数总是相同的。（ ）

24. 当 m 为 1 时，输出的第一行总为 n 。（ ）

● 单选题

25. 算法 $g(n,m)$ 最为准确的时间复杂度分析结果为（ ）。

A. $O(n^{3/2}m)$

B. $O(nm)$

C. $O(n^2m)$

D. $O(nm^2)$

26. 当输入为 “20 2” 时，输出的第一行为（ ）。

A. “4”

B. “5”

C. “6”

D. “20”

27. (4 分) 当输入为 “100 100” 时，输出的第一行为（ ）。

A. “6”

B. “7”

C. “8”

D. “9”

题号 CSP-J-2022-2-3

(3)

```

1. #include <iostream>
2.
3. using namespace std;
4.
5. int n, k;
6.
7. int solve1()
8. {
9.     int l = 0, r = n;
10.    while(l <= r) {
11.        int mid = (l + r) / 2;
12.        if (mid * mid <= n) l = mid + 1;
13.        else r = mid - 1;
14.    }
15.    return l - 1;
16. }
17.
18. double solve2(double x)
19. {
20.     if (x == 0) return x;
21.     for (int i = 0; i < k; i++)
22.         x = (x + n / x) / 2;
23.     return x;
24. }
25.
26. int main()
27. {
28.     cin >> n >> k;
29.     double ans = solve2(solve1());
30.     cout << ans << ' ' << (ans * ans == n) << endl;
31.     return 0;
32. }

```



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：

- 判断题

28. 该算法最准确的时间复杂度分析结果为 $O(\log n + k)$ 。()
29. 当输入为“9801 1”时，输出的第一个数为“99”。()
30. 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”。()
31. 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

32. 当输入为“2 1”时，输出的第一个数最接近 ()。
- A. 1
- B. 1.414
- C. 1.5
- D. 2
33. 当输入为“3 10”时，输出的第一个数最接近 ()。
- A. 1.7
- B. 1.732
- C. 1.75
- D. 2
34. 当输入为“256 11”时，输出的第一个数 ()。
- 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

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2022-3-1

(1) (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。

```
1. #include <bits/stdc++.h>
2. using namespace std;
3.
4. int main() {
5.     int n;
6.     cin >> n;
7.
8.     vector<int> fac;
9.     fac.reserve((int)ceil(sqrt(n)));
10.
11.     int i;
12.     for (i = 1; i * i < n; ++i) {
13.         if (①) {
14.             fac.push_back(i);
15.         }
16.     }
17.
18.     for (int k = 0; k < fac.size(); ++k) {
19.         cout << ② << " ";
20.     }
21.     if (③) {
22.         cout << ④ << " ";
23.     }
24.     for (int k = fac.size() - 1; k >= 0; --k) {
25.         cout << ⑤ << " ";
26.     }
27. }
```

35. ①处应填 ()

A. $n \% i == 0$

B. $n \% i == 1$

C. $n \% (i-1) == 0$

D. $n \% (i-1) == 1$

36. ②处应填 ()

A. $n / \text{fac}[k]$

B. $\text{fac}[k]$

C. $\text{fac}[k]-1$

D. $n / (\text{fac}[k]-1)$

37. ③处应填 ()

A. $(i-1) * (i-1) == n$

B. $(i-1) * i == n$

C. $i * i == n$

D. $i * (i-1) == n$

38. ④处应填 ()

A. $n-i$

B. $n-i+1$

C. $i-1$

D. i



39. ⑤处应填 ()

- A. $n / \text{fac}[k]$
- B. $\text{fac}[k]$
- C. $\text{fac}[k]-1$
- D. $n / (\text{fac}[k]-1)$

题号 CSP-J-2022-3-2

(2) (洪水填充) 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和待填充的颜色，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。

```
1. #include <bits/stdc++.h>
2. using namespace std;
3.
4. const int ROWS = 8;
5. const int COLS = 8;
6.
7. struct Point {
8.     int r, c;
9.     Point(int r, int c) : r(r), c(c) {}
10. };
11.
12. bool is_valid(char image[ROWS][COLS], Point pt,
13.               int prev_color, int new_color) {
14.     int r = pt.r;
15.     int c = pt.c;
16.     return (0 <= r && r < ROWS && 0 <= c && c < COLS &&
17.            ① && image[r][c] != new_color);
18. }
```

```

19.
20. void flood_fill(char image[ROWS][COLS], Point cur, int new_color) {
21.     queue<Point> queue;
22.     queue.push(cur);
23.
24.     int prev_color = image[cur.r][cur.c];
25.     ②;
26.
27.     while (!queue.empty()) {
28.         Point pt = queue.front();
29.         queue.pop();
30.
31.         Point points[4] = {③, Point(pt.r - 1, pt.c),
32.                             Point(pt.r, pt.c + 1), Point(pt.r, pt.c - 1)};
33.         for (auto p : points) {
34.             if (is_valid(image, p, prev_color, new_color)) {
35.                 ④;
36.                 ⑤;
37.             }
38.         }
39.     }
40. }
41.
42. int main() {
43.     char image[ROWS][COLS] = {{'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g'},
44.                                {'g', 'g', 'g', 'g', 'g', 'g', 'r', 'r'},
45.                                {'g', 'r', 'r', 'g', 'g', 'r', 'g', 'g'},
46.                                {'g', 'b', 'b', 'b', 'b', 'r', 'g', 'r'},
47.                                {'g', 'g', 'g', 'b', 'b', 'r', 'g', 'r'},
48.                                {'g', 'g', 'g', 'b', 'b', 'b', 'b', 'r'},
49.                                {'g', 'g', 'g', 'g', 'g', 'b', 'g', 'g'},
50.                                {'g', 'g', 'g', 'g', 'g', 'b', 'b', 'g'}};
51.
52.     Point cur(4, 4);
53.     char new_color = 'y';
54.
55.     flood_fill(image, cur, new_color);
56.
57.     for (int r = 0; r < ROWS; r++) {
58.         for (int c = 0; c < COLS; c++) {
59.             cout << image[r][c] << " ";
60.         }
61.         cout << endl;
62.     }
63.     // 输出:
64.     // g g g g g g g g
65.     // g g g g g g r r
66.     // g r r g g r g g
67.     // g y y y y r g r
68.     // g g g y y r g r
69.     // g g g y y y y r
70.     // g g g g g y g g
71.     // g g g g g y y g
72.

```

```
73.     return 0;  
74. }
```

40. ①处应填 ()

- A. image[r][c] == prev_color
- B. image[r][c] != prev_color
- C. image[r][c] == new_color
- D. image[r][c] != new_color

41. ②处应填 ()

- 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

42. ③处应填 ()

- 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)

43. ④处应填 ()

- 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

CSP-J-2022-3-2



视频解析

44. ⑤处应填 ()

A. queue.push(p)

B. queue.push(pt)

C. queue.push(cur)

D. queue.push(Point(ROWS, COLS))

CSP-J 2021 年初赛真题

认证时间：2021 年 9 月 19 日 14:30~16:30

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

题号 CSP-J-2021-1-1

1. 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



题号 CSP-J-2021-1-2

2. 以下奖项与计算机领域最相关的是（ ）。

- A. 奥斯卡奖
- B. 图灵奖
- C. 诺贝尔奖
- D. 普利策奖



题号 CSP-J-2021-1-3

3. 目前主流的计算机储存数据最终都是转换成（ ）数据进行储存。

- A. 二进制
- B. 十进制
- C. 八进制
- D. 十六进制



题号 CSP-J-2021-1-4

4. 以比较作为基本运算，在 N 个数中找出最大数，最坏情况下所需要的最少的比较次数为（ ）。

- A. N^2
- B. N
- C. $N-1$
- D. $N+1$



题号 CSP-J-2021-1-5

5. 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

题号 CSP-J-2021-1-6

6. 对于有 n 个顶点、 m 条边的无向连通图 ($m > n$)，需要删掉 () 条边才能使其成为一棵树。

- A. $n-1$
- B. $m-n$
- C. $m-n-1$
- D. $m-n+1$



题号 CSP-J-2021-1-7

7. 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



题号 CSP-J-2021-1-8

8. 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

题号 CSP-J-2021-1-9

9. 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. $*a*+bcd$



题号 CSP-J-2021-1-10

10. 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



题号 CSP-J-2021-1-11

11. 在数据压缩编码中的哈夫曼编码方法，在本质上是一种 () 的策略。

- A. 枚举



- B. 贪心
- C. 递归
- D. 动态规划

题号 CSP-J-2021-1-12

12. 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



题号 CSP-J-2021-1-13

13. 考虑如下递归算法

```
solve(n)
  if n<=1 return 1
  else if n>=5 return n*solve(n-2)
  else return n*solve(n-1)
```

则调用 solve(7)得到的返回结果为 ()。

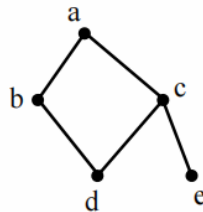
- A. 105
- B. 840
- C. 210
- D. 420



题号 CSP-J-2021-1-14

14. 以 a 为起点，对右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



题号 CSP-J-2021-1-15

15. 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



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 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2021-2-1

(1)

```
1. #include <iostream>
2. using namespace std;
3.
4. int n;
5. int a[1000];
6.
7. int f(int x)
8. {
9.     int ret = 0;
10.    for (; x; x &= x - 1) ret++;
11.    return ret;
12. }
13.
14. int g(int x)
15. {
16.    return x & -x;
17. }
18.
19. int main()
20. {
21.    cin >> n;
22.    for (int i = 0; i < n; i++) cin >> a[i];
23.    for (int i = 0; i < n; i++)
24.        cout << f(a[i]) + g(a[i]) << ' ';
25.    cout << endl;
26.    return 0;
```

27. }

CSP-J-2021-2-1



视频解析

● 判断题

16. 输入的 n 等于 1001 时，程序不会发生下标越界。()
17. 输入的 $a[i]$ 必须全为正整数，否则程序将陷入死循环。()
18. 当输入为 “5 2 11 9 16 10” 时，输出为 “3 4 3 17 5”。()
19. 当输入为 “1 511998” 时，输出为 “18”。()
20. 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

21. 当输入为 “2 -65536 2147483647” 时，输出为 ()。
- A. “65532 33”
- B. “65552 32”
- C. “65535 34”
- D. “65554 33”

题号 CSP-J-2021-2-2

```
1. #include <iostream>
2. #include <string>
3. using namespace std;
4.
5. char base[64];
6. char table[256];
```

```

7.
8. void init()
9. {
10.     for (int i = 0; i < 26; i++) base[i] = 'A' + i;
11.     for (int i = 0; i < 26; i++) base[26 + i] = 'a' + i;
12.     for (int i = 0; i < 10; i++) base[52 + i] = '0' + i;
13.     base[62] = '+', base[63] = '/';
14.
15.     for (int i = 0; i < 256; i++) table[i] = 0xff;
16.     for (int i = 0; i < 64; i++) table[base[i]] = i;
17.     table['='] = 0;
18. }
19.
20. string decode(string str)
21. {
22.     string ret;
23.     int i;
24.     for (i = 0; i < str.size(); i += 4) {
25.         ret += table[str[i]] << 2 | table[str[i + 1]] >> 4;
26.         if (str[i + 2] != '=')
27.             ret += (table[str[i + 1]] & 0x0f) << 4 | table[str[i +
28.                                     2]] >> 2;
29.         if (str[i + 3] != '=')
30.             ret += table[str[i + 2]] << 6 | table[str[i + 3]];
31.     }
32.     return ret;
33. }
34. int main()
35. {
36.     init();
37.     cout << int(table[0]) << endl;
38.
39.     string str;
40.     cin >> str;
41.     cout << decode(str) << endl;
42.     return 0;
43. }

```



● 判断题

22. 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。()

23. 可能存在输入不同，但输出的第二行相同的情形。()

24. 输出的第一行为“-1”。()

● 单选题

25. 设输入字符串长度为 n ，decode 函数的时间复杂度为 ()。

A. $\Theta(\sqrt{n})$

B. $\Theta(n)$

C. $\Theta(n \log n)$

D. $\Theta(n^2)$

26. 当输入为“Y3Nx”时，输出的第二行为 ()。

A. “csp”

B. “csq”

C. “CSP”

D. “Csp”

27. (3.5 分) 当输入为“Y2NmIDlwMjE=”时，输出的第二行为 ()。

A. “ccf2021”

B. “ccf2022”

C. “ccf 2021”

D. “ccf 2022”

题号 CSP-J-2021-2-3

(3)

```
1. #include <iostream>
2. using namespace std;
3.
4. const int n = 100000;
5. const int N = n + 1;
6.
7. int m;
8. int a[N], b[N], c[N], d[N];
9. int f[N], g[N];
10.
11. void init()
12. {
13.     f[1] = g[1] = 1;
14.     for (int i = 2; i <= n; i++) {
15.         if (!a[i]) {
16.             b[m++] = i;
17.             c[i] = 1, f[i] = 2;
18.             d[i] = 1, g[i] = i + 1;
19.         }
20.         for (int j = 0; j < m && b[j] * i <= n; j++) {
21.             int k = b[j];
22.             a[i * k] = 1;
23.             if (i % k == 0) {
24.                 c[i * k] = c[i] + 1;
25.                 f[i * k] = f[i] / c[i * k] * (c[i * k] + 1);
26.                 d[i * k] = d[i];
27.                 g[i * k] = g[i] * k + d[i];
28.                 break;
29.             }
30.             else {
31.                 c[i * k] = 1;
32.                 f[i * k] = 2 * f[i];
33.                 d[i * k] = g[i];
34.                 g[i * k] = g[i] * (k + 1);
35.             }
36.         }
37.     }
38. }
39.
```

```

40. int main()
41. {
42.     init();
43.
44.     int x;
45.     cin >> x;
46.     cout << f[x] << ' ' << g[x] << endl;
47.     return 0;
48. }

```



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

28. 若输入不为“1”，把第 13 行删去不会影响输出的结果。（ ）

29. (2 分) 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。（ ）

30. (2 分) 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

31. `init` 函数的时间复杂度为（ ）。

- A. $\Theta(n)$
- B. $\Theta(n \log n)$
- C. $\Theta(n\sqrt{n})$
- D. $\Theta(n^2)$

32. 在执行完 `init()` 后， $f[1], f[2], f[3] \dots f[100]$ 中有（ ）个等于 2。

- A. 23

B. 24

C. 25

D. 26

33. (4 分) 当输入为 “1000” 时，输出为 ()。

A. “15 1340”

B. “15 2340”

C. “16 2340”

D. “16 1340”

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2021-3-1

(1) (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 $0, 1, 0, 1, \dots$ 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。

试补全模拟程序。

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. const int MAXN = 1000000;
6. int F[MAXN];
7.
8. int main() {
9.     int n;
10.    cin >> n;
11.    int i = 0, p = 0, c = 0;
12.    while (①) {
13.        if (F[i] == 0) {
14.            if (②) {
15.                F[i] = 1;
16.                ③;
17.            }
18.            ④;
19.        }
20.        ⑤;
21.    }
```

```
22.     int ans = -1;
23.     for (i = 0; i < n; i++)
24.         if (F[i] == 0)
25.             ans = i;
26.     cout << ans << endl;
27.     return 0;
28. }
```

CSP-J-2021-3-1



视频解析

34. ①处应填 ()

- A. $i < n$
- B. $c < n$
- C. $i < n - 1$
- D. $c < n - 1$

35. ②处应填 ()

- A. $i \% 2 == 0$
- B. $i \% 2 == 1$
- C. p
- D. $!p$

36. ③处应填 ()

- A. $i++$
- B. $i = (i + 1) \% n$
- C. $c++$
- D. $p \wedge = 1$

37. ④处应填 ()

- A. $i++$
- B. $i = (i + 1) \% n$
- C. $c++$
- D. $p \wedge = 1$

38. ⑤处应填 ()

- A. $i++$
- B. $i = (i + 1) \% n$
- C. $c++$
- D. $p \wedge = 1$

题号 CSP-J-2021-3-2

(2) (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。

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. struct point {
6.     int x, y, id;
7. };
8.
9. bool equals(point a, point b) {
10.     return a.x == b.x && a.y == b.y;
11. }
```

```

12.
13. bool cmp(point a, point b) {
14.     return ①;
15. }
16.
17. void sort(point A[], int n) {
18.     for (int i = 0; i < n; i++)
19.         for (int j = 1; j < n; j++)
20.             if (cmp(A[j], A[j - 1])) {
21.                 point t = A[j];
22.                 A[j] = A[j - 1];
23.                 A[j - 1] = t;
24.             }
25. }
26.
27. int unique(point A[], int n) {
28.     int t = 0;
29.     for (int i = 0; i < n; i++)
30.         if (②)
31.             A[t++] = A[i];
32.     return t;
33. }
34.
35. bool binary_search(point A[], int n, int x, int y) {
36.     point p;
37.     p.x = x;
38.     p.y = y;
39.     p.id = n;
40.     int a = 0, b = n - 1;
41.     while (a < b) {
42.         int mid = ③;
43.         if (④)
44.             a = mid + 1;
45.         else
46.             b = mid;
47.     }
48.     return equals(A[a], p);
49. }
50.
51. const int MAXN = 1000;
52. point A[MAXN];
53.
54. int main() {
55.     int n;

```

```

56.     cin >> n;
57.     for (int i = 0; i < n; i++) {
58.         cin >> A[i].x >> A[i].y;
59.         A[i].id = i;
60.     }
61.     sort(A, n);
62.     n = unique(A, n);
63.     int ans = 0;
64.     for (int i = 0; i < n; i++)
65.         for (int j = 0; j < n; j++)
66.             if (⑤ && binary_search(A, n, A[i].x, A[j].y) &&
                    binary_search(A, n, A[j].x, A[i].y)) {
67.                 ans++;
68.             }
69.     cout << ans << endl;
70.     return 0;
71. }

```

CSP-J-2021-3-2



视频解析

39. ①处应填 ()

- A. $a.x \neq b.x ? a.x < b.x : a.id < b.id$
- B. $a.x \neq b.x ? a.x < b.x : a.y < b.y$
- C. $\text{equals}(a, b) ? a.id < b.id : a.x < b.x$
- D. $\text{equals}(a, b) ? a.id < b.id : (a.x \neq b.x ? a.x < b.x : a.y < b.y)$

40. ②处应填 ()

- A. $i == 0 \parallel \text{cmp}(A[i], A[i - 1])$
- B. $t == 0 \parallel \text{equals}(A[i], A[t - 1])$
- C. $i == 0 \parallel !\text{cmp}(A[i], A[i - 1])$
- D. $t == 0 \parallel !\text{equals}(A[i], A[t - 1])$

41. ③处应填 ()

- A. $b - (b - a) / 2 + 1$

- B. $(a + b + 1) >> 1$
- C. $(a + b) >> 1$
- D. $a + (b - a + 1) / 2$

42. ④处应填 ()

- A. `!cmp(A[mid], p)`
- B. `cmp(A[mid], p)`
- C. `cmp(p, A[mid])`
- D. `!cmp(p, A[mid])`

43. 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`

CSP-J 2020 年初赛真题

认证时间：2020 年 10 月 11 日 14:30~16:30

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

题号 CSP-J-2020-1-1

1. 在内存存储器中每个存储单元都被赋予一个唯一的序号，称为()。

- A. 下标
- B. 地址
- C. 序号
- D. 编号



题号 CSP-J-2020-1-2

2. 编译器的主要功能是()。

- A. 将源程序翻译成机器指令代码
- B. 将一种高级语言翻译成另一种高级语言
- C. 将源程序重新组合
- D. 将低级语言翻译成高级语言



题号 CSP-J-2020-1-3

3. 设 $x=\text{true}$, $y=\text{true}$, $z=\text{false}$, 以下逻辑运算表达式值为真的是 ()。

- A. $(x \wedge y) \wedge z$
- B. $x \wedge (z \vee y) \wedge z$
- C. $(x \wedge y) \vee (z \vee x)$
- D. $(y \vee z) \wedge x \wedge z$



题号 CSP-J-2020-1-4

4. 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. 4MB
- B. 8MB
- C. 32MB
- D. 16MB



题号 CSP-J-2020-1-5

5. 冒泡排序算法的伪代码如下:

输入: 数组 L , $n \geq k$ 。输出: 按非递减顺序排序的 L 。

算法 BubbleSort:

```
1. FLAG ← n //标记被交换的最后元素位置
2. while FLAG > 1 do
3.     k ← FLAG - 1
4.     FLAG ← 1
5.     for j=1 to k do
6.         if L(j) > L(j+1) then do
7.             L(j) ↔ L(j+1)
8.             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
- B. $n-2$
- C. n^2
- D. $n-1$



题号 CSP-J-2020-1-6

6. 设 A 是 n 个实数的数组，考虑下面的递归算法：

```
XYZ (A[1..n])
1. if n=1 then return A[1]
2. else temp ← XYZ (A[1..n-1])
3.     if temp < A[n]
4.         then return temp
5.         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 数组的最大值
- D. A 数组的中值



题号 CSP-J-2020-1-7

7. 链表不具有的特点是()

- A. 插入删除不需要移动元素
- B. 可随机访问任一元素
- C. 不必事先估计存储空间
- D. 所需空间与线性表长度成正比



题号 CSP-J-2020-1-8

8. 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. 10
- B. 12
- C. 9
- D. 11



题号 CSP-J-2020-1-9

9. 二进制数 1011 转换成十进制数是()。

- A. 10
- B. 13
- C. 11
- D. 12



题号 CSP-J-2020-1-10

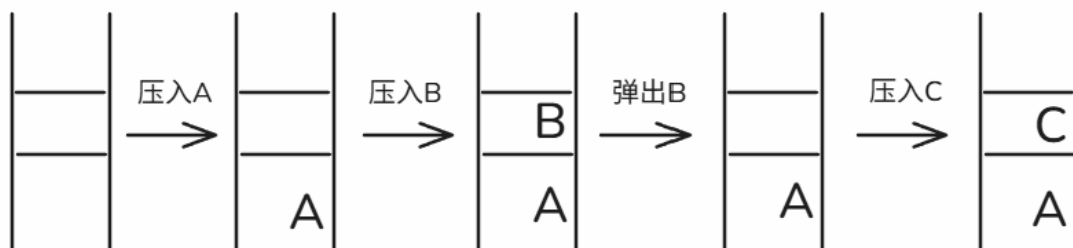
10. 五个小朋友并排站成一列,其中有俩个小朋友是双胞胎,如果要求这两个双胞胎必须相邻,则有()种不同排列方法?

- A. 24
- B. 36
- C. 72
- D. 48



题号 CSP-J-2020-1-11

11. 下图中所使用的数据结构是 ()。



- A. 哈希表
- B. 二叉树
- C. 栈
- D. 队列



题号 CSP-J-2020-1-12

12. 独根树的高度为 1。具有 61 个结点的完全二叉树的高度()。

- A. 7
- B. 5
- C. 8
- D. 6



题号 CSP-J-2020-1-13

13. 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. 己亥
- B. 己丑
- C. 己卯
- D. 己酉



题号 CSP-J-2020-1-14

14. 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. 56
- B. 84
- C. 72
- D. 504



题号 CSP-J-2020-1-15

15. 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. 30
- B. 150
- C. 180
- D. 120



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 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2020-2-1

(1)

```
1. #include <cstdlib>
2. #include <iostream>
3. using namespace std;
4.
5. char encoder[26] = {'C', 'S', 'P', 0};
6. char decoder[26];
7.
8. string st;
9.
10. int main() {
11.     int k = 0;
12.     for (int i = 0; i < 26; ++i)
13.         if (encoder[i] != 0) ++k;
14.     for (char x = 'A'; x <= 'Z'; ++x) {
15.         bool flag = true;
16.         for (int i = 0; i < 26; ++i)
17.             if (encoder[i] == x) {
18.                 flag = false;
19.                 break;
20.             }
21.         if (flag) {
22.             encoder[k] = x;
23.             ++k;
24.         }
25.     }
26.     for (int i = 0; i < 26; ++i)
```

```

27.     decoder[encoder[i] - 'A'] = i + 'A';
28.     cin >> st;
29.     for (int i = 0; i < st.length(); ++i)
30.         st[i] = decoder[st[i] - 'A'];
31.     cout << st;
32.     return 0;
33. }

```



● 判断题

- 1) 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。()
- 2) 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。
()
- 3) 将第 12 行的 “i<26” 改为 “i<16”，程序运行结果不会改变。()
- 4) 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()。
 - A. 输入的字符串中既有 A 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 S 又有 P
 - D. 输入的字符串中既有 A 又有 B
- 6) 若输出的字符串为 “CSPCSPCSPCSP”，则下列说法正确的是()。
 - A. 输入的字符串中既有 J 又有 R

- B. 输入的字符串中既有 P 又有 K
- 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

题号 CSP-J-2020-2-2

(2)

```
1. #include <iostream>
2. using namespace std;
3.
4. long long n, ans;
5. int k, len;
6. long long d[1000000];
7.
8. int main() {
9.     cin >> n >> k;
10.    d[0] = 0;
11.    len = 1;
12.    ans = 0;
13.    for (long long i = 0; i < n; ++i) {
14.        ++d[0];
15.        for (int j = 0; j + 1 < len; ++j) {
16.            if (d[j] == k) {
17.                d[j] = 0;
18.                d[j + 1] += 1;
19.                ++ans;
20.            }
21.        }
22.        if (d[len - 1] == k) {
23.            d[len - 1] = 0;
24.            d[len] = 1;
25.            ++len;
26.            ++ans;
```

```
27.     }
28.     }
29.     cout << ans << endl;
30.     return 0;
31. }
```

CSP-J-2020-2-2



视频解析

假设输入的 n 是不超过 262 的正整数， k 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

- 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

- 4) 若输入的 n 等于 10^{15} ，输入的 k 为 1，则输出等于()。

- A. $(10^{30}-10^{15})/2$
- B. $(10^{30}+10^{15})/2$
- C. 1
- D. 10^{15}

- 5) 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（ ）。

- A. $(3^{30}-1) / 2$
- B. 3^{30}
- C. $3^{30}-1$

D. $(3^{30}+1)/2$

6) 若输入的 n 等于 100,010,002,000,090, 输入的 k 为 10, 则输出等于
()。

A.11,112,222,444,543

B.11,122,222,444,453

C.11,122,222,444,543

D.11,112,222,444,453

题号 CSP-J-2020-2-3

(3)

```
1. #include <algorithm>
2. #include <iostream>
3. using namespace std;
4.
5. int n;
6. int d[50][2];
7. int ans;
8.
9. void dfs(int n, int sum) {
10.     if (n == 1) {
11.         ans = max(sum, ans);
12.         return;
13.     }
14.     for (int i = 1; i < n; ++i) {
15.         int a = d[i - 1][0], b = d[i - 1][1];
16.         int x = d[i][0], y = d[i][1];
17.         d[i - 1][0] = a + x;
18.         d[i - 1][1] = b + y;
19.         for (int j = i; j < n - 1; ++j)
```

```

20.     d[j][0] = d[j + 1][0], d[j][1] = d[j + 1][1];
21.     int s = a + x + abs(b - y);
22.     dfs(n - 1, sum + s);
23.     for (int j = n - 1; j > i; --j)
24.         d[j][0] = d[j - 1][0], d[j][1] = d[j - 1][1];
25.     d[i - 1][0] = a, d[i - 1][1] = b;
26.     d[i][0] = x, d[i][1] = y;
27. }
28. }
29.
30. int main() {
31.     cin >> n;
32.     for (int i = 0; i < n; ++i)
33.         cin >> d[i][0];
34.     for (int i = 0; i < n; ++i)
35.         cin >> d[i][1];
36.     ans = 0;
37.     dfs(n, 0);
38.     cout << ans << endl;
39.     return 0;
40. }

```



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

- 1) 若输入 n 为 0，此程序可能会死循环或发生运行错误。()
- 2) 若输入 n 为 20，接下来的输入全为 0，则输出为 0。()
- 3) 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

- 4) 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()。

A. 1917

B. 1908

C. 1881

D. 1890

5) 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()。

A. 2020

B. 2030

C. 2010

D. 2000

6) (4 分)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()。

A. 2420

B. 2220

C. 2440

D. 2240

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2020-3-1

1. （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 $120=2\times 2\times 2\times 3\times 5$ 。

输入保证 $2\leq n\leq 10^9$ 。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。

```
1. #include <stdio>
2. using namespace std;
3.
4. int n, i;
5.
6. int main() {
7.     scanf("%d", &n);
8.     for(i = ①; ② <= n; i++) {
9.         ③ {
10.             printf("%d ", i);
11.             n = n / i;
12.         }
13.     }
14.     if(④)
15.         printf("%d ", ⑤);
16.     return 0;
17. }
```

CSP-J-2020-3-1



视频解析

1) ①处应填()

A. $n - 1$

B. 0

C. 1

D. 2

2) ②处应填()

A. n / i

B. $n / (i * i)$

C. $i * i * i$

D. $i * i$

3) ③处应填()

A. `if (i * i <= n)`

B. `if(n % i == 0)`

C. `while(i * i <= n)`

D. `while(n % i == 0)`

4) ④处应填()

A. $n > 1$

B. $n <= 1$

C. $i + i <= n$

D. $i < n / i$

5) ⑤处应填()

A. 2

B. i

C. n / i

D. n

题号 CSP-J-2020-3-2

2. (最小区间覆盖) 给出 n 个区间, 第 i 个区间的左右端点是 $[a_i, b_i]$ 。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 $0 \leq i \leq m$ 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 ($1 \leq n \leq 5000, 1 \leq m \leq 10^9$)。

接下来 n 行, 每行两个整数 a_i, b_i ($0 \leq a_i, b_i \leq m$)。

提示: 使用贪心法解决这个问题。先用 $\theta(n^2)$ 的时间复杂度排序, 然后贪心选择这些区间。

试补全程序。

```
1. #include <iostream>
2.
3. using namespace std;
4.
```

```

5. const int MAXN = 5000;
6. int n, m;
7. struct segment { int a, b; } A[MAXN];
8.
9. void sort() // 排序
10. {
11.     for (int i = 0; i < n; i++)
12.         for (int j = 1; j < n; j++)
13.             if (①)
14.                 {
15.                     segment t = A[j];
16.                     ②
17.                 }
18. }
19.
20. int main()
21. {
22.     cin >> n >> m;
23.     for (int i = 0; i < n; i++)
24.         cin >> A[i].a >> A[i].b;
25.     sort();
26.     int p = 1;
27.     for (int i = 1; i < n; i++)
28.         if (③)
29.             A[p++] = A[i];
30.     n = p;
31.     int ans = 0, r = 0;
32.     int q = 0;
33.     while (r < m)
34.     {
35.         while (④)
36.             q++;
37.         ⑤;
38.         ans++;
39.     }
40.     cout << ans << endl;
41.     return 0;
42. }

```



1) ①处应填()

- A. $A[j].b < A[j - 1].b$
- B. $A[j].b > A[j - 1].b$
- C. $A[j].a < A[j - 1].a$
- D. $A[j].a > A[j - 1].a$

2) ②处应填()

- 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) ③处应填()

- A. $A[i].b < A[p - 1].b$
- B. $A[i].b > A[i - 1].b$
- C. $A[i].b > A[p - 1].b$
- D. $A[i].b < A[i - 1].b$

4) ④处应填()

- A. $q + 1 < n \ \&\& \ A[q + 1].b \leq r$
- B. $q + 1 < n \ \&\& \ A[q + 1].a \leq r$
- C. $q < n \ \&\& \ A[q].a \leq r$
- D. $q < n \ \&\& \ A[q].b \leq r$

5) ⑤处应填()

A. $r = \max(r, A[q + 1].a)$

B. $r = \max(r, A[q].b)$

C. $r = \max(r, A[q + 1].b)$

D. $q++$

CSP-J 2019 年初赛真题

认证时间：2019 年 10 月 19 日 14:30~16:30

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

题号 CSP-J-2019-1-1

1. 中国的国家顶级域名是()

- A. .cn
- B. .ch
- C. .chn
- D. .china



题号 CSP-J-2019-1-2

2. 二进制数 11 1011 1001 0111 和 01 0110 1110 1011 进行按位与运算的结果是()。

- A. 01 0010 1000 1011
- B. 01 0010 1001 0011
- C. 01 0010 1000 0001
- D. 01 0010 1000 0011



题号 CSP-J-2019-1-3

3. 一个 32 位整型变量占用()个字节。

- A. 32
- B. 128
- C. 4
- D. 8



题号 CSP-J-2019-1-4

4. 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;



题号 CSP-J-2019-1-5

5. 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

题号 CSP-J-2019-1-6

6. 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



题号 CSP-J-2019-1-7

7. 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

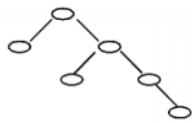
- A. 22
- B. 24
- C. 18
- D. 20



题号 CSP-J-2019-1-8

8. 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()。

- A. 6
- B. 10
- C. 15
- D. 12



题号 CSP-J-2019-1-9

9. 100 以内最大的素数是()。

- A. 89
- B. 97
- C. 91
- D. 93



题号 CSP-J-2019-1-10

10. 319 和 377 的最大公约数是()。

- A. 27
- B. 33
- C. 29
- D. 31



题号 CSP-J-2019-1-11

11. 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



题号 CSP-J-2019-1-12

12. 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



题号 CSP-J-2019-1-13

13. 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



题号 CSP-J-2019-1-14

14. 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



题号 CSP-J-2019-1-15

15. 以下哪个奖项是计算机科学领域的最高奖？（ ）

- A. 图灵奖
- B. 鲁班奖
- C. 诺贝尔奖
- D. 普利策奖



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 1.5 分，选择题 3 分，共计 40 分

题号 CSP-J-2019-2-1

(1)

```
1. #include <stdio>
2. #include <string>
3. using namespace std;
4. char st[100];
5. 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. }
```

CSP-J-2019-2-1



视频解析

● 判断题

- 1) 输入的字符串只能由小写字母或大写字母组成。()
- 2) 若将第 8 行的 “i = 1” 改为 “i = 0”，程序运行时会发生错误。()
- 3) 若将第 8 行的 “i <= n” 改为 “i*i <= n”，程序运行结果不会改变。()

4) 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

5) 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。

A. 18

B. 6

C. 10

D. 1

6) 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。

A. 36

B. 100000

C. 1

D. 128

题号 CSP-J-2019-2-2

(2)

```
1. #include <cstdio>
2. using namespace std;
3. int n, m;
```

```

4. int a[100], b[100];
5.
6. int main() {
7.     scanf("%d%d", &n, &m);
8.     for (int i = 1; i <= n; ++i)
9.         a[i] = b[i] = 0;
10.    for (int i = 1; i <= m; ++i) {
11.        int x, y;
12.        scanf("%d%d", &x, &y);
13.        if (a[x] < y && b[y] < x) {
14.            if (a[x] > 0)
15.                b[a[x]] = 0;
16.            if (b[y] > 0)
17.                a[b[y]] = 0;
18.            a[x] = y;
19.            b[y] = x;
20.        }
21.    }
22.    int ans = 0;
23.    for (int i = 1; i <= n; ++i) {
24.        if (a[i] == 0)
25.            ++ans;
26.        if (b[i] == 0)
27.            ++ans;
28.    }
29.    printf("%d\n", ans);
30.    return 0;
31. }

```

CSP-J-2019-2-2



视频解析

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

- 1) 当 $m > 0$ 时，输出的值一定小于 $2n$ 。 ()
- 2) 执行完第 27 行的 “++ans” 时，ans 一定是偶数。 ()

3) $a[i]$ 和 $b[i]$ 不可能同时大于 0。()

4) 若程序执行到第 13 行时, x 总是小于 y , 那么第 15 行不会被执行。()

● 选择题

5) 若 m 个 x 两两不同, 且 m 个 y 两两不同, 则输出的值为()

A. $2n-2m$

B. $2n+2$

C. $2n-2$

D. $2n$

6) 若 m 个 x 两两不同, 且 m 个 y 都相等, 则输出的值为()

A. $2n-2$

B. $2n$

C. $2m$

D. $2n-2m$

题号 CSP-J-2019-2-3

(3)

```
1. #include <iostream>
2. using namespace std;
3. const int maxn = 10000;
4. int n;
5. int a[maxn];
6. int b[maxn];
7. int f(int l, int r, int depth) {
```

```

8.     if (l > r)
9.         return 0;
10.    int min = maxn, mink;
11.    for (int i = l; i <= r; ++i) {
12.        if (min > a[i]) {
13.            min = a[i];
14.            mink = i;
15.        }
16.    }
17.    int lres = f(l, mink - 1, depth + 1);
18.    int rres = f(mink + 1, r, depth + 1);
19.    return lres + rres + depth * b[mink];
20. }
21. int main() {
22.     cin >> n;
23.     for (int i = 0; i < n; ++i)
24.         cin >> a[i];
25.     for (int i = 0; i < n; ++i)
26.         cin >> b[i];
27.     cout << f(0, n - 1, 1) << endl;
28.     return 0;
29. }

```



● 判断题

- 1) 如果 a 数组有重复的数字，则程序运行时会发生错误。()
- 2) 如果 b 数组全为 0，则输出为 0。()

● 选择题

- 3) 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：()。

- A. 5000
- B. 600
- C. 6

D. 100

4) 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: ()。

A. 100

B. 6

C. 5000

D. 600

5) 当 $n=10$ 时, 若 b 数组满足, 对任意 $0 \leq i < n$, 都有 $b[i] = i + 1$, 那么输出最大为()。

A. 386

B. 383

C. 384

D. 385

6) (4 分) 当 $n=100$ 时, 若 b 数组满足, 对任意 $0 \leq i < n$, 都有 $b[i] = 1$, 那么输出最小为()。

A. 582

B. 580

C. 579

D. 581

三、完善程序

单选题，每小题 3 分，共计 30 分

题号 CSP-J-2019-3-1

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 $\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ ，数字 1 变成矩阵 $\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ 。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]；矩阵变幻 1 次后： $\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ ；矩阵变幻 2 次后：

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

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。

```
1. #include <stdio>
2. using namespace std;
3. int n;
```

```

4. 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. }

```

CSP-J-2019-3-1



视频解析

1) ①处应填()

- A. $n \% 2$
- B. 0
- C. t
- D. 1

2) ②处应填()

A. $x - \text{step}$, $y - \text{step}$

B. x , $y - \text{step}$

C. $x - \text{step}$, y

D. x , y

3) ③处应填()

A. $x - \text{step}$, $y - \text{step}$

B. $x + \text{step}$, $y + \text{step}$

C. $x - \text{step}$, y

D. x , $y - \text{step}$

4) ④处应填()

A. $n - 1$, $n \% 2$

B. n , 0

C. n , $n \% 2$

D. $n - 1$, 0

5) ⑤处应填()

A. $1 \ll (n + 1)$

B. $1 \ll n$

C. $n + 1$

D. $1 \ll (n - 1)$

题号 CSP-J-2019-3-2

2. (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，将 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。

从小到大排序后输出。

数据范围 $1 \leq n \leq 10^7$, $1 \leq a[i], b[i] \leq 10^4$ 。

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.
11. int main() {
```

```

12.     scanf("%d", &n);
13.     for (int i = 0; i < n; ++i)
14.         scanf("%d%d", &a[i], &b[i]);
15.     memset(cnt, 0, sizeof(cnt));
16.     for (int i = 0; i < n; ++i)
17.         ①; // 利用 cnt 数组统计数量
18.     for (int i = 0; i < maxs; ++i)
19.         cnt[i + 1] += cnt[i];
20.     for (int i = 0; i < n; ++i)
21.         ②; // 记录初步排序结果
22.     memset(cnt, 0, sizeof(cnt));
23.     for (int i = 0; i < n; ++i)
24.         ③; // 利用 cnt 数组统计数量
25.     for (int i = 0; i < maxs; ++i)
26.         cnt[i + 1] += cnt[i];
27.     for (int i = n - 1; i >= 0; --i)
28.         ④; // 记录最终排序结果
29.     for (int i = 0; i < n; ++i)
30.         printf("%d %d\n", ⑤);
31.     return 0;
32. }

```

CSP-J-2019-3-2



视频解析

1) ①处应填()

- A. ++cnt[i]
- B. ++cnt[b[i]]
- C. ++cnt[a[i] * maxs + b[i]]
- D. ++cnt[a[i]]

2) ②处应填()

- 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) ③处应填()

A. `++cnt[b[i]]`

B. `++cnt[a[i] * maxs + b[i]]`

C. `++cnt[a[i]]`

D. `++cnt[i]`

4) ④处应填()

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) ⑤处应填()

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]]]`

CSP-J 历年真题参考答案及视频解析汇总

CSP-J 初赛 真题年份	参考答案及视频解析链接	扫二维码 看解析
CSP-J 2025 答案及解析	https://www.chenyuanlong.com/CSP-J-2025-answer	 CSP-J 2025
CSP-J 2024 答案及解析	https://www.chenyuanlong.com/CSP-J-2024-answer	 CSP-J 2024
CSP-J 2023 答案及解析	https://www.chenyuanlong.com/CSP-J-2023-answer	 CSP-J 2023
CSP-J 2022 答案及解析	https://www.chenyuanlong.com/CSP-J-2022-answer	 CSP-J 2022
CSP-J 2021 答案及解析	https://www.chenyuanlong.com/CSP-J-2021-answer	 CSP-J 2021
CSP-J 2020 答案及解析	https://www.chenyuanlong.com/CSP-J-2020-answer	 CSP-J 2020
CSP-J 2019 答案及解析	https://www.chenyuanlong.com/CSP-J-2019-answer	 CSP-J 2019

CSP-J 官方大纲（NOI 大纲-入门级）

最新官方 NOI 大纲版本为 2025 年发布

NOI 大纲官方 PDF 链接：

https://www.noi.cn/upload/resources/file/2025/04/18/NOI_Syllabus_Edition_2025.pdf



本讲义勘误声明

本讲义中的 CSP-J 真题均整理自网络公开资料，受限于人工整理与排版，难免存在疏漏或错误之处。如发现内容有误，欢迎通过以下方式查看最新勘误：

勘误链接：

<https://www.chenyuanlong.com/CSP-J-KW>



感谢您的理解，如有其他问题或建议，也欢迎随时反馈。