
# 普及组 CSP-J 2025 初赛模拟卷 9

## 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. $(1ACF)_{16}$ 和 $(0456)_{16}$ 这两个十六进制数做加法的结果是（　　）。

A. $(1F25)_{16}$　　B. $(7975)_{10}$　　C. $(17455)_{8}$　　D. $(1111100100111)_{2}$

2. 一个 64 位无符号长整型变量占用（　　）字节。

A. 32　　B. 4　　C. 16　　D. 8

3. 下列选项中，（　　）判断字符串 s1 是否为回文串，如果是就输出“yes”，否则输出“no”。

```
int main()
{
    string s1, s2;
    cin>>s1;
    s2 = s1;
    ____________;
    if(s1 == s2)
        cout<<"yes";
    else
        cout<<"no";
    return 0;
}
```

A. `reverse(s1.begin(), s1.end());`

B. `reverse(s1[0], s1[s1.size()]);`

C. `s1.reverse(begin(), end());`

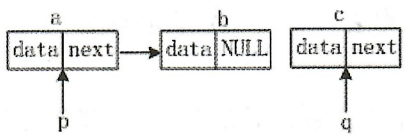
D. `reverse(s1, s1+s1.size());`

4. 已知 x、y、z 都是 int 类型的整数，x=1、y=1、z=3。那么执行 bool ans = x++||--y&&++z 后，x、y、z 和 ans 的值各为多少？（　　）

A. x=2, y=0, z=4, ans=1　　B. x=2, y=1, z=3, ans=1

C. x=2, y=1, z=3, ans=0　　D. x=2, y=0, z=4, ans=0

5. 指针 p 指向变量 a，q 指向变量 c。能够把 c 插入到 a 和 b 之间并形成新链表的语句组是（　　）。




A. `p.next = q; q.next = p.next;`

B. `p->next = &c; q->next = p->next;`

C. `(*p).next = q; (*q).next = &b;`

D. `a.next = c; c.next = b;`

6. 以下哪个特性是数组和链表共有的？（　　）

A. 动态分配　　B. 元素之间的次序关系

C. 通过索引访问　　D. 存储连续

7. 下面关于哈夫曼树的描述中，正确的是（　　）。

A. 哈夫曼树一定是完全二叉树

B. 哈夫曼树一定是平衡二叉树

C. 哈夫曼树中权值最小的两个结点互为兄弟结点

D. 哈夫曼树中左子结点小于父结点，右子结点大于父结点

8. 已知一棵二叉树有 2025 个结点，则其中至多有（　　）个结点有 2 个子结点。

A. 1010　　B. 1011　　C. 1012　　D. 1013

9. 下面的说法中正确的是（　　）。

A. 计算机网络按照拓扑结构分为星型、环型、总线型等

B. 互联网的基础是 OSI 七层协议而不是 TCP/IP 协议族

C. 现代计算机网络主要采用电路交换技术

D. 10.10.1.1 是 D 类 IP 地址

10. 下面关于图的说法中正确的是（　　）。

A. 所有点数为奇数的连通图，一定可以一笔画成

B. 所有只有两个奇度点[其余均为偶度点]的连通图，一定可以一笔画成

C. 哈密顿图一定是欧拉图，而欧拉图未必是哈密顿图

D. 哈密顿图不一定是欧拉图，而欧拉图一定是哈密顿图

11.（　　）是一种选优搜索法，按选优条件向前搜索以达到目标。当搜索到某一步时，如果发现原先的选择并不优或者达不到目标，就后退一步重新选择。

A. 二分算法　　B. 动态规划　　C. 回溯法　　D. 贪心算法

12. 动态规划是将一个问题分解为一系列子问题后来求解，下面（　　）属于动态规划问题。

A. 多重背包　　B. 排队打水　　C. 有序数组找数　　D. 全排列

13. 设无向图 G 的邻接矩阵如下图所示，则 G 的顶点数和边数分别为（　　）。

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

A. 4, 5　　B. 5, 8　　C. 4, 10　　D. 5, 5

14. 某条道路从东到西有 8 个路灯，巡查员为了维护方便，在每根灯杆上都安装了开关，第 $t$ 个开关能够切换前 $t$ 个灯的状态（$t$=1~8，灯开或关），一开始灯全是开的。巡查员通过控制开关一共能得到（　　）种不同灯的开或者关的组合状态。

A. 128　　B. 256　　C. 127　　D. 255

15. 某四位正整数 abcd 满足如下条件（a,n 也是正整数，b,c,d 是非负整数）：$abcd=1^3+2^3+\cdots+n^3$，$abcd=(1+2+3+\cdots+n)^2$，$abcd=(ab+cd)^2$，这样的正整数 abcd 共有（　　）个。

A. 0　　B. 1　　C. 2　　D. 3

**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，判断题每小题 1.5 分，选择题每小题 3 分，共计 40 分）**

**（1）**

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int N = 100 + 5;
04 int n, c, x, y, len, l[N], r[N], cha[N];
```

```
05 char a[N];
06 int main() {
07     scanf("%d%d%s", &n, &c, a + 1);
08     len = n;
09     for (int i = 1; i <= c; i++) {
10         scanf("%d%d", &l[i], &r[i]);
11         cha[i] = len - l[i] + 1;
12         len += r[i] - l[i] + 1;
13     }
14     scanf("%d", &x);
15     for (int i = c; i; i--)
16         if (x >= l[i] + cha[i] && x <= r[i] + cha[i])
17             x -= cha[i];
18     printf("%c\n", a[x]);
19     return 0;
20 }
```

输入保证$1 \leqslant l_i \leqslant r_i \leqslant n \leqslant 100$，$1 \leqslant c \leqslant 100$。回答以下问题。

■ **判断题**

16. 第 17 行最多会运行一次。 ( )
17. （2 分）当程序运行至第 19 行时，x 一定在[1,n]范围内。 ( )
18. 若将第 3 行改成 `const int N = 100;`，一定不会出现数组越界问题。 ( )

■ **选择题**

19. 若输入 `4 2 mark 1 4 5 7 10`，则输出为（ ）。

A. m B. a C. r D. k

20. （4 分）若输入 `7 3 creamii 2 3 3 4 2 9 11`，则输出为（ ）。

A. m B. e C. a D. i

**（2）**

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int N = 2e5 + 5;
04 int n, ans, a[N], cnt[20];
05 int main() {
06     scanf("%d", &n);
```

```
07    for (int i = 1; i <= n; ++i) {
08        scanf("%d", &a[i]);
09        for (int j = 0; j <= 14; ++j) {
10            cnt[j] += a[i] & 1;
11            a[i] /= 2;
12        }
13    }
14    for (int i = 1; i <= n; ++i) {
15        int sum = 0, x = 1;
16        for (int j = 0; j <= 14; ++j) {
17            if (cnt[j])
18                sum += x, --cnt[j];
19            x *= 2;
20        }
21        ans += sum * sum;
22    }
23    return printf("%d\n", ans), 0;
24 }
```

已知 $1 \leqslant n, a < 2^{15}$，完成下列各题。

■ 判断题

21. 第 10 行可以写成 `cnt[j] += a[i] % 2`。（　　）
22. 第 21 行一定不会溢出 `int` 上界。（　　）
23. 若输入为 `1` $a_1$，则输出为 $a_1^2$。（　　）
24. 若输入为 `3 1 3 5`，则输出为 `51`。（　　）

■ 选择题

25. 该程序的时间复杂度为（　　）。

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

26. 若输入为 `2 123 69`，则程序运行至第 13 行时，`cnt` 数组的和为（　　）。

A. 6　　B. 7　　C. 9　　D. 10

**（3）**

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int N = 10005, M = 15;
```

```
04 char c[N];
05 int d, num[N], dp[N][M][2];
06 int dfs(int pos, int res, int sta) {
07     if (pos == 0)
08         return res == 0;
09     if (dp[pos][res][sta] != -1)
10         return dp[pos][res][sta];
11     int ret = 0, maxx = 9;
12     if (sta) maxx = num[pos];
13     for (int i = 0; i <= maxx; i++)
14         ret += dfs(pos - 1, (res + i) % d, sta && (i == maxx));
15     dp[pos][res][sta] = ret;
16     return ret;
17 }
18 int main() {
19     scanf("%s%d", c + 1, &d);
20     memset(dp, -1, sizeof(dp));
21     for (int i = 1; i <= strlen(c + 1); i++)
22         num[i] = c[strlen(c + 1) - i + 1] - '0';
23     printf("%d\n", dfs(strlen(c + 1), 0, 1) - 1);
24     return 0;
25 }
```

已知$1 \leqslant d < 10$，$1 \leqslant |c| \leqslant 10\,000$，完成下列各题。

■ **判断题**

27. 将程序中的第 2 行去除，程序依然能正常运行。（　　）

28. 该程序的时间复杂度为$O(|c|^2)$。（　　）

■ **选择题**

29. 若将程序中的第 15 行去除，则程序的时间复杂度为（　　）。

A. $O(10^{|c|})$　　B. $O(100d|c|)$　　C. $O(10d|c|)$　　D. $O(10^{d|c|})$

30. 若输入为 9 2，则输出为（　　）。

A. 1　　B. 2　　C. 4　　D. 7

31. 若输入为 30 4，则输出为（　　）。

A. 3　　B. 4　　C. 6　　D. 7

32.（4分）若输入为 2025 6，则输出为（　　）。

A. 240　　B. 256　　C. 280　　D. 338

## 三、完善程序（单选题，每小题3分，共计30分）

**（1）题目描述：**

有一个长度为 n 的数组 a，满足 a[i]只能是 0、1 或 2，一开始所有元素均为蓝色。可以执行如下操作：

（i）用一枚硬币，把一个蓝色元素涂成红色；

（ii）选择一个不等于 0 的红色元素和一个与其相邻的蓝色元素，将所选的红色元素减少 1，并将所选的蓝色元素涂成红色。

要将所有元素涂红，最少需要多少枚硬币？

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int N = 2e5 + 5;
04 int n, pre[N], a[N], dp[N][3];
05 int main() {
06    scanf("%d", &n);
07     memset(dp, 0x3f, sizeof dp);
08     dp[0][0] = dp[0][1] = dp[0][2] = 0;
09     for (int i = 1; i <= n; i++)
10        scanf("%d", &a[i]);
11     ①;
12     for (int i = 1; i <= n; i++)
13        pre[i] = ②;
14     for (int i = 2, j; i <= n; i++) {
15        dp[i][a[i]] = min({ ③ });
16        if (④)
17           dp[i][a[i] - 1] = 1 + min({⑤});
18     }
19     printf("%d", min({ dp[n][0], dp[n][1], dp[n][2] }));
20 }
```

33. ①处应填（　　）。

A. dp[1][a[1]==2]=1　　B. dp[1][a[1]==1]=1

C. dp[1][a[1]==0]=1　　D. dp[1][a[1]]=1

34. ②处应填（　　）。

A. a[i]?pre[i-1]:i　　B. a[i]?i:pre[i-1]

C. a[i]==2?pre[i-1]:i　　D. a[i]==2?i:pre[i-1]

35. ③处应填（　　）。

A. dp[i - 1][0] + 1, dp[i - 1][1], dp[i - 1][2]

B. dp[i - 1][0] + 2, dp[i - 1][1] + 1, dp[i - 1][2]

C. dp[i - 1][0] + 2, dp[i - 1][1], dp[i - 1][2]

D. dp[i - 1][0], dp[i - 1][1], dp[i - 1][2]

36. ④处应填（　　）。

A. a[i]　　B. a[i]==2　　C. a[i]==1　　D. a[i-1]

37. ⑤处应填（　　）。

A. dp[pre[i] - 1][0] + 1, dp[pre[i] - 1][1], dp[pre[i] - 1][2]

B. dp[pre[i] - 1][0] + 2, dp[pre[i] - 1][1] + 1, dp[pre[i] - 1][2]

C. dp[pre[i] - 1][0] + 2, dp[pre[i] - 1][1], dp[pre[i] - 1][2]

D. dp[pre[i] - 1][0], dp[pre[i] - 1][1], dp[pre[i] - 1][2]

**（2）题目描述：**

给你一个长度为 n（n≤300000）的整数数组 a。

你可以执行以下操作：选择数组中的一个元素，并用其邻近元素的值替换它。

计算在执行上述操作最多 k（k≤10）次的情况下，数组的总和可能达到的最小值。

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int N = 3e5 + 5;
04 int n, k, a[N], p[N][11], ans[N][11];
05 int main() {
06     scanf("%d%d", &n, &k);
07     for (int i = 1; i <= n; i++) {
08         scanf("%d", &a[i]);
09         ①;
10     }
11     for (int j = 1; j <= k; j++)
12         for (int i = 1; i + j <= n; i++)
```

```
13            p[i][j] = min(p[i][j - 1], a[i + j]);
14        for (int j = 1; j <= k; j++)
15            for (int i = 1; i + j <= n; i++)
16                ②;
17        for (int i = 1; i <= n; i++) {
18            ans[i][0] = ③;
19            for (int j = 1; j <= k; j++) {
20                ans[i][j] = min(ans[i - 1][j] + a[i], ans[i][j - 1]);
21                for (int h = 0; ④; h++)
22                    ans[i][j] = min(ans[i][j], ⑤);
23            }
24        }
25        printf("%d\n", ans[n][k]);
26        return 0;
27    }
```

38. ①处应填（　　）。

A. p[i][0]=i　　B. p[i][0]=a[i]

C. p[i][i]=i　　D. p[i][i]=a[i]

39. ②处应填（　　）。

A. p[i][j]*=j　　B. p[i][j]*=(j+1)

C. p[i][j]*=i　　D. p[i][j]*=(i+1)

40. ③处应填（　　）。

A. ans[i-1][0]+a[i]　　B. ans[i-k][0]+p[i-k+1][k]

C. ans[i-1][0]+a[i]*i　　D. ans[i-k][0]+p[i-k+1][k]*k

41. ④处应填（　　）。

A. h<=i&&h<=j　　B. h<i&&h<=j　　C. h<i&&h<j　　D. h<=i&&h<j

42. ⑤处应填（　　）。

A. ans[i-h-1][j-h]+p[i-h][h]　　B. ans[i-h][j-h]+p[i-h][h]

C. ans[i][j-h]+p[i][h]　　D. ans[i][j-h]+p[i-h][h]