普及组 CSP-J 2024 初赛模拟卷 1

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 Z 的 ASCII 码十进制表示是 90，那么英文字母 B 的 ASCII 码二进制表示是（ ）。

   A. 01000001 B. 01000010 C. 01000011 D. 01000000

2. 以下关于 CSP 与 NOIP 的描述正确的是（ ）。

   A. CSP 属于非专业认证，只有在校生才能参加

   B. CSP 是中国电子学会举办的程序设计竞赛

   C. CSP 和 NOIP 毫无关系，没参加 CSP 也可以直接参加 NOIP

   D. CSP 和 NOIP 都是 CCF 旗下的程序设计赛事

3. 以下不能用作 C++程序中的标识符的是（ ）。

   A. `private` B. `friends` C. `news` D. `pascal`

4. NOI 复赛测评机所用的 Linux 系统属于（ ）。

   A. UML B. IDE C. OS D. Database

5. 如果 65 536 种颜色用二进制编码来表示，至少需要（ ）个二进制位。

   A. 16 B. 8 C. 12 D. 10

6. 搜索算法中的 BFS 算法经常用到的数据结构是（ ）。

   A. 堆 B. 栈 C. 链表 D. 队列

7. 在已经从小到大排好序的 n 元素单向链表中查询是否存在关键字为 k 的元素，最坏情况下运行的时间复杂度是（ ）。

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

8. 在下列各种排序算法中，不是以“比较”作为主要操作的算法是（ ）。

   A. 归并排序 B. 快速排序 C. 冒泡排序 D. 桶排序

9. 关于计算机网络，下面的说法中正确的是（　　）。

A. 现在的计算机必须连接到互联网才能正常运行

B. 192.168.0.1 是 A 类 IP 地址

C. 互联网的诞生用到了现代计算机技术和现代通信技术

D. 接入互联网的计算机的 IP 地址已经全部升级到了 IPv6 地址

10. 将(2, 6, 10, 17)分别存储到某个地址区间为 0~10 的哈希表中，如果哈希函数 h(x) = (　　)，将不会产生冲突，其中 a%b 表示 a 除以 b 的余数，sqrt 表示开平方，floor 表示向下取整。

A. x%11　　B. x^2%11　　C. 2x%11　　D. floor(sqrt(x))%11

11. 现在有一个十六进制数 27，它等于二进制数的（　　）。

A. 100011　　B. 100101　　C. 100111　　D. 100011

12. 以下逻辑表达式中，不管 A、B 如何取值，恒为假的是（　　）。

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

C. $A \wedge ((\neg A \vee B) \vee (A \vee \neg B)) \wedge \neg A$　　D. $((\neg A \vee B) \vee (A \vee \neg B)) \wedge A \wedge \neg B$

13. 某二叉树有 16 个结点都同时有左孩子结点和右孩子结点，则该二叉树中的叶子结点数是（　　）个。

A. 19　　B. 17　　C. 18　　D. 16

14. 现有 16 张不同的卡片，其中红、黄、蓝、绿色卡片各 4 张。从中任取 3 张，要求红色最多有 1 张并且 3 张卡片不能是同一种颜色，不同的取法组合共有（　　）种。

A. 232　　B. 472　　C. 256　　D. 484

15. 有 8 个结点的非连通无向图最多有（　　）条边。

A. 8　　B. 7　　C. 21　　D. 49

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 int gcd(int a, int b){
04     int tmp;
05     if(b) tmp = a%b;
06     else return a;
07     while(tmp){
08         a = b;
09         b = tmp;
10         tmp = a%b;
11     }
12     return b;
13 }
14 int lcm(int a, int b){
15     return a/gcd(a,b)*b;
16 }
17 int main(){
18     int a,b;
19     cin >> a >> b;
20     cout << gcd(a, b) << endl;
21     return 0;
22 }
```

■ 判断题

16. 若输入 0 2024，则输出结果为 0。 ( )
17. 将第 5 行中的 if(b)改为 if(0 != b)，程序的运行结果不会改变。 ( )
18. 若输入 2.4 4.8，则输出错误。 ( )
19. 将第 15 行 return a/gcd(a,b)*b 替换成 return a*b/gcd(a,b)，程序的运行结果不会改变。 ( )

■ 选择题

20. 若输入数据为 20244204 12348，则输出为 ( )。

A. 18 B. 36 C. 12 D. 24

21. 若将第 20 行 cout << gcd(a,b) << endl 替换成 cout << lcm(a, b) << endl，输入数据为 20244204 12348，则输出为 ( )。

A. 6,943,761,972 B. 程序出错，无输出

C. 3,471,880,986 D. 某个负数

(2)

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 int main()
04 {
05     char s[128] = {0};
06     cin.getline(s,128);
07     for(int i = 0; i < strlen(s); ++i){
08         if(s[i] >= 65 && s[i] <= 90){
09             if(s[i] == 90)
10                 s[i] = 65;
11             else
12                 s[i] += 1;
13             s[i] ^= ' ';
14         }
15     }
16     cout << s << endl;
17     return 0;
18 }
```

■ **判断题**

22. 输入一个长度大于 128 的字符串，程序的输出一定会出错。 ( )
23. 将第 6 行 cin.getline(s,128)更换为 getline(cin,s)，程序的运行结果不变。 ( )
24. 将第 13 行 s[i] ^= ' '更换为 s[i] ^= 32，程序的运行结果不变。 ( )
25. 将第 9 行 if(s[i] == 90)更换为 if(s[i] == 'Z')，程序的运行结果不变。 ( )

■ **选择题**

26. 若输入字符串 s 为 CSPjs2024，则输出为 ( )。

A. dtqjs2024 B. cspjs2024 C. DTQjs2024 D. CSPjs2024

27. 若输出 bcdea，则输入字符串 s 为 ( )。

A. BCDEA B. ABCDZ C. abcde D. bcdez

(3)

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

```
03 int used[20],a[20],n;
04 long long ret=0;
05 bool flag;
06 void dfs(int x)
07 {
08    int i;
09    if(x > n)
10    {
11       flag=1;
12       for(i=1;i<=n;i++)
13          if(a[i]+i > n+2)
14          {
15             flag=0;
16             break;
17          }
18       if(flag) ret++;
19       return;
20    }
21    for(i=1;i<=n;i++)
22       if(used[i]==0)
23       {
24          used[i]=1, a[x]=i;
25          dfs(x+1);
26          used[i]=0, a[x]=0;
27       }
28 }
29 int main()
30 {
31    cin>>n;
32    dfs(1);
33    cout<<ret;
34    return 0;
35 }
```

■ 判断题

28. 如果输入 n 的值为 0，那么程序在运行过程中一定会出现错误。 ( )
29. 如果将第 26 行的 a[x]=0 去掉，输出的结果不会改变。 ( )
30. 该程序算法的时间复杂度是 $O(n!*n)$。 ( )
31. 输入某个正整数 n，程序运行的输出结果可能会等于 0。 ( )

■ 选择题

32. 若输入 n=2，那么输出结果是（　　）。

A. 1　　B. 2　　C. 3　　D. 0

33. 若输入 n=5，那么输出结果是（　　）。

A. 16　　B. 5　　C. 10　　D. 12

34.（4 分）若输出结果为 128，则输入 n 是（　　）。

A. 8　　B. 7　　C. 16　　D. 32

三、完善程序（单选题，每小题 3 分，共计 30 分）

（1）输入一个十进制正整数 n，然后将 n 转换为二进制数，最后统计二进制数的各位数字，看看一共有多少位为 1，然后打印出总数。

输入格式：

第 1 行输入十进制正整数 n。

输出格式：

输出一个整数，表示十进制正整数 n 转换成的二进制数中有多少位为 1。

输入样例：

127

输出样例：

7

样例说明：

十进制数 127 转换为二进制数 1111111，二进制位一共有 7 个 1，所以输出 7。

```
01 #include<bits/stdc++.h>
02 using namespace std;
03 int a[33],cnt;
04 int main()
05 {
06     int n,sum,x;
07     cin>>n;
08     cnt=0;
09     ①;
10     while(x>0)
```

```
11    {
12        a[②]=x%2;
13        ③;
14    }
15    sum=0;
16    for(int i=1;④;i++)
17    if(a[i]==1)
18        ⑤;
19    cout<<sum<<endl;
20    return 0;
21 }
```

35. ①处应填（　　）。

A. x=n　　B. x=1　　C. x=0　　D. x=n-1

36. ②处应填（　　）。

A. --cnt　　B. ++cnt　　C. cnt--　　D. cnt

37. ③处应填（　　）。

A. x/=2　　B. n++　　C. x++　　D. n--

38. ④处应填（　　）。

A. i<cnt　　B. i<cnt/2　　C. i<=cnt　　D. i<=cnt/2

39. ⑤处应填（　　）。

A. sum--　　B. sum=x　　C. sum=0　　D. sum++

（2）在一个 $n \times n$ 的棋盘上布满了 0 和 1，如图(a)所示（n=7）。为叙述方便，将 0 用字母表示，如图(b)所示。

	1	2	3	4	5	6	7
1	1	1	0	1	1	0	1
2	0	1	1	1	0	1	1
3	1	1	0	1	1	0	1
4	1	0	1	1	0	1	1
5	1	0	1	0	1	1	1
6	1	1	0	1	1	0	1
7	1	0	1	1	0	1	0

(a)

1	1	*A*	1	1	*B*	1
C	1	1	1	*D*	1	1
1	1	*E*	1	1	*F*	1
1	*G*	1	1	*H*	1	1
1	*I*	1	*J*	1	1	1
1	1	*K*	1	1	*L*	1
1	*M*	1	1	*N*	1	*P*

(b)

跳棋规则如下。

(i) 从某个 0 格出发，可以向上、下、左、右 4 个方向连续越过若干个（至少 1 个）1 格后跳入下一个 0 格。如图(b)所示，从 A 出发，可跳到 B，或者跳到 E，但不能直接跳到 K。在跳到 B 之后还可以继续跳到 F，在跳到 E 之后可继续跳到 F 或 K，直到不能再跳为止。

(ii) 每个 0 格只能到达一次，给出的起始点不能再次到达，也不能越过。跳过的距离为跳过 1 格的个数加 1，如从 A 到 B，跳过距离为 3，从 B 到 F，跳过距离为 2。

问题：当给出棋盘和起始点之后，最远能跳的距离是多少？

如图(b)所示，从 A 出发，可跳的路线不止一条，其中一条为：

A — B — F — L — K — E （可能不唯一）

3 2 3 3 3

它的跳过距离为 14。

输入格式：

第 1 行 3 个整数 n（$1 \leqslant n \leqslant 100$）、$x$、$y$（$x$,$y$ 是起始点坐标，图(b)中 A 处坐标为 1,3）。接下来 n 行，每行 n 个数（0 或 1），数与数之间用一个空格分隔。

输出格式：

一个整数，即最大可跳距离（若不能跳，则输出 0）。

输入样例：

```
4 3 2
1 0 1 0
1 1 1 1
0 0 1 0
1 1 0 1
```

输出样例：

```
6
```

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 int n,i,j,k,ans;
04 int a[105][105],vis[105][105]; //a 表示棋盘，vis 统计点是否走过
05 int dx[4]={-1,1,0,0},dy[4]={0,0,1,-1};
06 void dfs(int x,int y,int step)
07 {
08    ans = ①;
```

```
09    for(int i=0;i<4;i++)
10    {
11        int tx=x,ty=y,s=0;
12        while(tx+dx[i]>0 && tx+dx[i]<=n
              && ty+dy[i]>0 && ty+dy[i]<=n)
13        {
14            tx+=dx[i];
15            ty+=dy[i];
16            s++;
17            if(②)
18                break;
19        }
20        if(tx>0 && tx<=n && ty>0 && ty<=n && vis[tx][ty]==0
              && a[tx][ty]==0 && s!=1)
21        {
22            vis[tx][ty]=1;
23            dfs(tx,ty,③);
24            ④;
25        }
26    }
27 }
28 int main()
29 {
30    int x,y,i,j;
31    cin>>n>>x>>y;
32    for(i=1;i<=n;i++)
33        for(j=1;j<=n;j++)
34            cin>>a[i][j];
35    ⑤;
36    dfs(x,y,0);
37    cout<<ans<<endl;
38    return 0;
39 }
```

40. ①处应填（　　）。

A. 0　　B. max(ans,step)

C. 1　　D. step

41. ②处应填（　　）。

A. `vis[tx][ty] == 1`　　B. `vis[tx][ty] == 0`

C. `a[tx][ty] == 1`　　D. `a[tx][ty] == 0`

42. ③处应填（　　）。

A. `step+s`　　B. `step+1`　　C. `step`　　D. `step-1`

43. ④处应填（　　）。

A. `vis[tx][ty] = 1`　　B. `vis[tx][ty] = 0`

C. `a[tx][ty] = 1`　　D. `a[tx][ty] = 0`

44. ⑤处应填（　　）。

A. `a[x][y] = 1`　　B. `a[x][y] = 0`

C. `vis[x][y] = 1`　　D. `vis[x][y] = 0`