普及组 CSP-J 2024 初赛模拟卷 2

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++程序中用到的一个常量 `a = 5e-6` 在内存中占（ ）空间。
   A. 2 字节 B. 1 字节 C. 4 字节 D. 8 字节

2. 以下关于 CSP 与 NOIP 的描述正确的是（ ）。
   A. CSP 属于专业认证，只有计算机专业在校生才能参加
   B. CSP-J/CSP-S 是中国通信学会举办的程序设计竞赛
   C. CSP-J 初赛零分也可以直接报名参加 NOIP
   D. CSP-J 和 CSP-S 都是 CCF 牵头举办的程序设计赛事

3. 某单位安装一条电信宽带进行上网，运营商说下行速度是 500 Mbps。要下载大小为 10 GB 的软件，最快大约需要（ ）秒。
   A. 2 B. 20 C. 200 D. 2000

4. 大写字母 M 的 ASCII 码整数值和空格的 ASCII 码整数值之和，是字母 m 的 ASCII 码整数值。空格的 ASCII 码整数值是（ ）。
   A. 32 B. 31 C. 30 D. 29

5. 在微型计算机中，（ ）的存取速度最快。
   A. RAM B. CD-ROM C. 高速缓存 D. 寄存器

6. 搜索算法中的 DFS 算法经常用到的数据结构是（ ）。
   A. 堆 B. 栈 C. 链表 D. 队列

7. 以下哪个说法是正确的？（ ）
   A. 花括号“{”和“}”只能作为 C++函数体的定界符
   B. 构成 C++程序的基本单位是函数，所有函数名都可以由用户命名
   C. 分号是 C++语句之间的分隔符，不是语句的一部分
   D. C++程序中的注释部分可以出现在程序中任意合适的地方

8. 在下列排序算法中，STL 中的 sort()函数采用的主要算法是（　　）。

A. 选择排序　　B. 快速排序　　C. 冒泡排序　　D. 拓扑排序

9. 以下哪个说法是正确的？（　　）

A. 第一台电子计算机 ENIAC 是基于集成电路的产物

B. 计算机必须要同时有 IP 地址和域名才能接入互联网

C. david@163.com 是一个正确的电子邮箱地址

D. 手机上收到的短信，里面的链接可以随意点击打开

10. 以下不能对二维数组 a 进行正确初始化的语句是（　　）。

A.
```
int a[2][3]={{1,2},{3,4},{5,6}};
```

B.
```
int a[][3]={{1,2},{0}};
```

C.
```
int a[2][3]={0};
```

D.
```
int a[][3]={1,2,3,4,5,6};
```

11. 现在有一个八进制数 274，其转换成的二进制数是（　　）。

A. 10111011　　B. 10111101　　C. 10111100　　D. 10101100

12. 设 A = true，B = false，C = false，D = true，以下逻辑运算表达式的值为假的是（　　）。

A. ((A ∧ B) ∨ C) ∧ D　　B. (A ∨ B) ∧ (C ∨ D)

C. A ∧ ((B ∨ C) ∨ D)　　D. (A ∧ (B ∨ C)) ∨ D

13. 二叉树的中序序列为 ABCEFGHD，后序序列为 ABFHGEDC，则其前序序列为（　　）。

A. CBADEGHF　　B. CBADEGFH

C. CBDAEGFH　　D. CBADGEFH

14. 从班级中体育比较好的 12 人中选 5 人去参加运动会，其中甲、乙、丙最多同时选两人，不同的选法共有（　　）种。

A. 792　　B. 756　　C. 720　　D. 676

15. 以下哪个结构可以用来存储图？（　　）

A. 栈　　B. 二叉树　　C. 邻接表　　D. 队列

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<cstring>
03 using namespace std;
04 char s1[1005],s2[1005];
05 int a[1005],b[1005],c[1005];
06 int main()
07 {
08     int la,lb,lc;
09     scanf("%s",s1);
10     scanf("%s",s2);
11     la = strlen(s1);
12     lb = strlen(s2);
13     lc = max(la,lb) + 1;
14     for(int i=0;i<la;++i)
15         a[la-i] = s1[i] - '0';
16     for(int i=0;i<lb;++i)
17         b[lb-i] = s2[i] - '0';
18     for(int i=1;i<=lc;i++)
19     {
20         c[i] += a[i] + b[i];
21         c[i+1] = c[i]/10;
22         c[i] = c[i]%10;
23     }
24     if(c[lc]==0 && lc>0)
25         lc--;
26     for(int i=lc;i>0;i--)
27         printf("%d",c[i]);
28     return 0;
29 }
```

■ 判断题

16. 将第 2 行代码改为#include<stdio.h>，程序的运行结果不会改变。 （ ）
17. 将第 9~10 行代码改为 cin>>s1>>s2;，程序的运行结果不会改变。 （ ）
18. 若输入两个都超过 1005 位长的正整数，则程序一定会出错且无输出。 （ ）

19. 在输入 0 0 的情况下，将第 24 行代码中的 lc>0 去掉，程序的运行结果不会改变。（ ）

■ 选择题

20. 若输入数据为 1024 1000，则输出为（ ）。

A. 24　　B. 2024　　C. 1024　　D. 1000

21. 若输入数据为 1 -1，则输出为（ ）。

A. 1　　B. 0　　C. -1　　D. 以上都不是

（2）

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 const int MAXN = 10005;
04 int n, a[MAXN], b[MAXN];
05 void mergesort(int *a, int l, int r)
06 {
07     int i,j,cnt,mid;
08     if (l == r)
09         return;
10     mid = (l + r)/2;
11     mergesort(a, l, mid);
12     mergesort(a, mid + 1, r);
13     i = l, j = mid + 1, cnt = 0;
14     while (i <= mid && j <= r)
15     {
16         if (a[i] <= a[j])
17             b[++cnt] = a[i++];
18         else
19             b[++cnt] = a[j++];
20     }
21     while (i <= mid)
22         b[++cnt] = a[i++];
23     while (j <= r)
24         b[++cnt] = a[j++];
25     for (i = l; i <= r; i++)
26         a[i] = b[i - l + 1];
27 }
```

```
28 int main(void)
29 {
30     cin >> n;
31     for (int i = 1; i <= n; i++)
32         cin >> a[i];
33     mergesort(a, 1, n);
34     for (int i = 1; i <= n; i++)
35         cout << a[i] << (i == n ? '\n' : ' ');
36     return 0;
37 }
```

■ 判断题

22. 该排序算法用到的是不稳定的排序算法。 ( )
23. 将第 10 行改为 `mid = l + r >> 1;`，程序的输出结果不变。 ( )
24. 该排序算法用到了分治的思想。 ( )
25. 第 35 行代码用到的三目运算符处理代码可以用等价的条件语句来写。 ( )

■ 选择题

26. 在最坏情况下，该算法的时间复杂度和下面哪个算法相当？ ( )

A. 插入排序 B. 选择排序 C. 堆排序 D. 快速排序

27. 若输出 2 3 5 7 8，则输入可能为 ( )。

A. 1 2 4 6 7 B. 8 7 5 2 3 C. 3 4 2 5 7 D. 8 2 3 4 5

(3)

```
01 #include<bits/stdc++.h>
02 using namespace std;
03 int t,x[100],a[100];
04 void dfs(int d,int i,int n)
05 {
06     if(n==1)
07     {
08         for(int k=0;k<d;k++)
09             printf("%4d",a[k]);
10         printf("\n");
11     }
12     else
13         for(int k=i;k<t;k++)
```

```
14          if(n%x[k]==0)
15          {
16              a[d]=x[k];
17              dfs(d+1,k,n/x[k]);
18          }
19 }
20 int main()
21 {
22     int n;
23     cin>>n;
24     for(int i=n;i>1;i--)
25         if(n%i==0)
26             x[t++]=i;
27     dfs(0,0,n);
28     return 0;
29 }
```

■ 判断题

28. 该程序的作用是对 n 进行质因数分解并从小到大依次打印。 ( )
29. 将第 9 行代码 printf("%4d",a[k]);中的 4 去掉，程序输出不变。 ( )
30. 第 24~26 行的作用是求出 n 的所有因子。 ( )
31. 程序运行过程中，若输入 n 为 0 或者负数，程序一定会打印错误，崩溃退出。 ( )

■ 选择题

32. 若输入 6，则输出为 ( )。

A. 6
3 2

B. 72
36 2

C. 6
2 3

D. 72
2 36

33. 若输入 n=1，那么输出结果可能是 ( )。

A. 2　　B. 1　　C. 0　　D. 什么也不输出

34. （4 分）若输入 2024，则输出有 ( ) 行。

A. 18　　B. 20　　C. 21　　D. 19

三、完善程序（单选题，每小题 3 分，共计 30 分）

（1）扫雷游戏是一款十分经典的单机小游戏。在 n 行 m 列的雷区中有一些格子含有地雷（称为地雷格），其他格子不含地雷（称为非地雷格）。玩家翻开一个非地雷格时，该格子中将会出现一个数字，提示周围格子中有多少个是地雷格。玩家的目标是在不翻出任何地雷格的条件下，找出所有的非地雷格。请将程序补充完整。

现在给出 n 行 m 列的雷区中的地雷分布，要求计算出每个非地雷格周围的地雷格数。

注：一个格子的周围格子包括其上、下、左、右、左上、右上、左下、右下这 8 个方向上与之直接相邻的格子。

输入格式：

第 1 行是用一个空格隔开的两个整数 n 和 m，分别表示雷区的行数和列数。接下来 n 行，每行 m 个字符，描述了雷区中的地雷分布情况。字符*表示相应格子是地雷格，字符?表示相应格子是非地雷格。相邻字符之间无分隔符。

输出格式：

输出文件包含 n 行，每行 m 个字符，描述整个雷区。用*表示地雷格，用周围的地雷个数表示非地雷格。相邻字符之间无分隔符。

输入样例：

```
3 3
*??
???
?*?
```

输出样例：

```
*10
221
1*1
```

```
01 #include<bits/stdc++.h>
02 using namespace std;
03 const int dx[] = {1, 1, 1, 0, 0, -1, -1, -1};
04 const int dy[] = {-1, 0, 1, -1, 1, -1, 0, 1};
05 char g[101][101];
06 int main()
07 {
08     int n,m,cnt;
09     cin>>n>>m;
```

```
10    for(int i=0;i<n;i++)
11       for(int j=0;j<m;j++)
12          cin>>g[i][j];
13    for(int i=0;i<n;i++)
14    {
15       for(int j=0;j<m;j++)
16          if ( ① )
17          {
18             ②;
19             for (int k = 0; ③ ; k++)
20                if ( ④ )
21                   cnt++;
22             cout << cnt;
23          }
24          else
25             cout<<"*";
26       if( ⑤ )
27          cout<<endl;
28   }
29   return 0;
30 }
```

35. ①处应填（　　）。

A. g[i][j] != '?'　　B. g[i][j] != '\0'

C. g[i][j] != '*'　　D. g[i][j] == '*'

36. ②处应填（　　）。

A. cnt++　　B. cnt = 0　　C. cnt = 0　　D. ++cnt = 0

37. ③处应填（　　）。

A. k < 8　　B. k < m　　C. k < n　　D. k < min(m,n)

38. ④处应填（　　）。

A. g[i + dy[k]][j + dx[k]] == '*'

B. g[i - dx[k]][j - dy[k]] == '*'

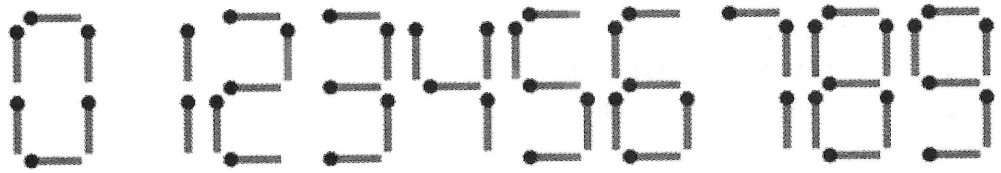
C. g[i + dx[k]][j + dy[k]] == '*'

D. g[i - dy[k]][j - dx[k]] == '*'

39. ⑤处应填（　　）。

A. `j != m`　　B. `j != m-1`　　C. `i != n`　　D. `i != n-1`

（2）给你 n 根火柴棍，你可以拼出多少个形如 A+B=C 的等式？等式中的 A、B、C 是用火柴棍拼出的整数（若该数非零，则最高位不能是 0）。用火柴棍拼数字 0~9 的拼法如图所示。



注意：

1. 加号与等号各自需要两根火柴棍；
2. 如果 A 不等于 B，则视 A+B=C 与 B+A=C 为不同的等式（A,B,C≥0）；
3. n 根火柴棍必须全部用上。

输入格式：

一个整数 n（$1 \leqslant n \leqslant 24$）。

输出格式：

一个整数，表示能拼成的不同等式的数目。

输入样例：

18

输出样例：

9

样例说明：

9 个等式为 0+4=4、0+11=11、1+10=11、2+2=4、2+7=9、4+0=4、7+2=9、10+1=11、11+0=11。

```
01 #include<bits/stdc++.h>
02 using namespace std;
03 int hcb[10]={6,2,5,5,4,5,6,3,7,6};
04 int n;
05 int matches(int num)
06 {
07     int i,k=0;
08     for(i=num;i!=0;①)
09         ② ;
```

```
10    if( ③ )
11       k+=hcb[0];
12    return k;
13 }
14 int main()
15 {
16    int i,j,count;
17    ④;
18    cin>>n;
19    for(i=0;i<=1000;i++)
20       for(j=0;j<=1000;j++)
21          if( ⑤ )
22             count++;
23    cout<<count;
24    return 0;
25 }
```

40. ①处应填（　　）。

A. i%=10　　B. i/=10　　C. i++　　D. i--

41. ②处应填（　　）。

A. k += hcb[i]　　B. k += hcb[i/10]

C. k += hcb[i/10%10]　　D. k += hcb[i%10]

42. ③处应填（　　）。

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

43. ④处应填（　　）。

A. count=1　　B. count=match(n)

C. count=0　　D. count=n

44. ⑤处应填（　　）。

A. matches(i)+matches(j)+matches(i+j)+6==n

B. matches(i)+matches(j)+matches(i+j)+3==n

C. matches(i)+matches(j)+matches(i+j)+4==n

D. matches(i)+matches(j)+matches(i+j)+5==n