2022 LGR 非专业级别软件能力认证第一轮

（SCP-J）入门级 C++语言模拟试题

认证时间：2022 年 8 月 23 日 09:30~11:30

考生注意事项：

- 试题纸共有 12 页，答题纸共有 1 页，满分 100 分。请在答题纸上作答，写在试题纸上的一律无效。
- 不得使用任何电子设备（如计算器、手机、电子词典等）或查阅任何书籍资料。

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．$(1047)_8 =$（　　）。

A．$(1011011101)_2$　　B．$(11010)_5$

C．$(20213)_4$　　D．$(308)_{16}$

2．若逻辑变量 A、C 为真，B、D 为假，以下逻辑表达式的值为假的是（　）。

A．(B∨C∨D)∨D∧A　　B．((¬A∧B)∨C)∧¬B

C．(A∧B)∨¬(C∧D∨¬A)　　D．A∧(D∨¬C)∧B

3．小恺编写了如下函数，希望计算斐波那契数列 f(n)第 n 项对 10000 取余数的值：

```
int f(int x) {
    if(x <= 2)
        return 1;
    int ans = f(x - 1) + f(x - 2);
    ans %= 10000;
    return ans;
}
```

在运行空间限制 128MB、栈空间不超过空间限制、运行时限 1 秒的情况下，在主函数中运行函数 f(12345)，则最有可能首先发生什么问题？

A．运行时间超时　　B．栈溢出

C．访问无效内存　　D．返回错误的答案

4．表达式 a+b*(c-d)/e-f 的后缀表达式为（　）。

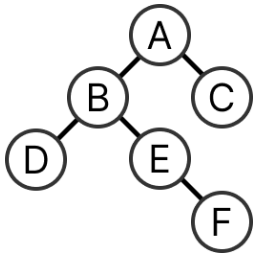
A．-+a/*b-c-cdef　　B．abcd-*e/+f-

C．+ab*-cd/e-f　　D．f-e/d-d*b+a

5. 某个 MV 是一段时长 4 分整的视频文件。它每秒播放 10 帧图像，每帧图像是一幅分辨率为 2048×1152 像素（长宽比 16:9）的 32 位真彩色图像，其画面没有被压缩。这个视频没有音频。这个视频文件大约需要占用多大的存储空间？（　　）。

A. 21 GiB　　B. 27 GiB
C. 168 GiB　　D. 2 GiB

6. 下图是一棵二叉树，它的后序遍历是（　　）。



A. ABDEFC　　B. DBEFAC　　C. DFEBCA　　D. ABCDEF

7. 五个本质不同的点在没有重边或者自环的情况下，组成不同的无向图的个数是（　　）？

A. 10　　B. 1024　　C. 15　　D. 120

8. 设元素 a,b,c,d,e,f 依次入栈，则下列不合法的出栈序列为（　　）？

A. d,c,b,e,f,a　　B. f,e,d,c,b,a
C. c,d,f,e,b,a　　D. e,d,b,a,f,c

9. 同时扔出 3 枚完全相同的六面骰子，每个骰子上有 1 到 6 的数字。将得到的点数排序后，有（　）种不同的结果？

A. 208　　B. 56　　C. 216　　D. 120

10. 在编程时（使用任一种高级语言，不一定是 C++），如果需要从磁盘文件中输入一个很大的二维数组（例如 1000*1000 的 double 型数组），按行读（即外层循环是关于行的）与按列读（即外层循环是关于列的）相比，在输入效率上（　）。

A. 没有区别　　B. 按行读的方式更高
C. 按列读的方式更高　　D. 取决于数组的存储方式

11. 不考虑稳定性，下列排序方法中平均时间复杂度最大的是（　　）。

A. 插入排序　　B. 希尔排序
C. 归并排序　　D. 快速排序

12. 将数组 12,23,-1,19,117,-103,79,602 中的元素按从大到小的顺序排列，每次可以交换任意两个元素，最少需要交换（　　）次

A. 4　　B. 5　　C. 6　　D. 7

13．3 名男生和 3 名女生围成一个圈，男生和男生不相邻，女生和女生不相邻。如果两个围成的圈经过旋转可以重合，则视为同一种方案。请问一共有几种方案？

A．18　　B．15　　C．12　　D．9

14．以下关于 C++字符串的说法，错误的是（　）。

A．定义 string 类型的字符串时，不需要预先确定它的最大长度。

B．字符数组和 string 类型的字符串是可以相互转化的。

C．定义字符数组 char a[100] 时并从键盘读入字符串，则读入的字符串长度不能超过 99。

D．定义一个字符串 string s 后，获得它长度的方式就是 strlen(s)。

15．中国计算机协会成立于（　）年。

A．1961　　B．1962　　C．1971　　D．1972

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 2 分，选择题 3 分，共计 40 分）

1.

```
#include <iostream>
using namespace std;
const int MAXN = 1000050;
int n, a[MAXN], a1[MAXN], b[MAXN], lim;
void solve1() {
    for (int i = 1; i <= n; i++)
        b[a[i]]++; //①
    for (int i = 1; i <= lim; i++) {
        if (b[i]) //②
            cout << i << " ";
    }
    cout << endl;
}
void solve2() {
    int cnt = 0, flag;
    for (int i = 1; i <= n; i++) {
        flag = false;
        for (int j = 1; j <= n - 1; j++) {
            if (a[j] > a[j + 1]) {
                swap(a[j], a[j + 1]);
                cnt++;
                flag = true;
            }
        }
        //if (flag==false)
        //  break;
```

```
    }
    for (int i = 1; i <= n; i++)
        cout << a[i] << " ";
    cout << endl;
}
int main() {
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
        a1[i] = a[i]; //③
        lim = max(a[i], lim); //④
    }
    solve1();
    for (int i = 1; i <= n; i++)
        a[i] = a1[i];
    solve2();
    return 0;
}
```

已知，$1 \le n \le 10^6, 1 \le a_i \le 10^6$。完成下面的判断题和单选题：

- 判断题

1）solve2 函数实现了选择排序。（　）

2）solve1 函数的时间复杂度为 $O(n+V)$，其中 V 指的是 a_i 的最大值。（　）

3）当输入数据为：7 2 3 5 7 1 4 6 时，solve2 函数中的变量 cnt 最终值为 9。（　）

4）若将 solve2 函数中的双斜杠全部移除，不会影响输出结果。（　）

- 单选题

5）下列哪组数据，会使得 solve1 函数与 solve2 函数的输出结果**不同**？（　）（假设已经输入了 $n = 8$）。

A. 1 10 100 1000 10000 888 8888 88888
B. 6321 158987 16305 68486 50556 847 156505 15610
C. 777 888 999 888 777 888 999 666
D. 999993 999994 999995 999996 999997 999998 999999 1000000

6）若要使得 solve1 和 solve2 函数的输出结果**相同**，则应当修改程序中的哪一处？

A.① B.② C.③ D④

2.

```
1  #include <cstdio>
2  #include <cstring>
3
4  const int maxn = 1003;
5
6  int type, n, m;
7  char s[maxn], t[maxn];
8
9  int main() {
10     scanf("%d %s %s", &type, t, s);
11     n = strlen(s); m = strlen(t);
12     if (type < 2) {
13         for (int i = 0; i < m; ++i) s[i] = t[i];
14     } else if (type == 2) {
15         strcpy(s, t);
16 // 提示：如果此时调用 printf("%s\n", s)，则本次输出结果为整个 t 字符串和换行，没有其他字符。
17     } else {
18         for (int i = 0; i < m; i += 4) {
19             unsigned int code = 0, pos = i;
20             for (int j = 1; pos < i+4; j*=100, ++pos) {
21                 if (pos == m) break;
22                 code += t[pos] * j;
23             }
24             pos = i;
25             while (code != 0) {
26                 s[pos++] = code % 100;
27                 code /= 100;
28             }
29         }
30     }
31     for (int i = 0; i < n; ++i) printf("%c", s[i]);
32     printf("\n");
33 }
```

输入保证 t 的长度不大于 s 的长度，且两字符串均只含有大小写字母，不是空串，type = 1,2,3，完成下面的判断题和单选题：

- 判断题

1) 将程序中所有的小于号 (<) 改为不等于号 (!=)，则程序对所有符合要求的输入的输出结果**不变**。 （ ）

2) 当输入为 `1 xyz abcd` 时，程序的输出为 `xyzd`。 （ ）

3) 程序在输入为 `1 xyz abcd` 时的输出与输入为 `2 xyz abcd` 的输出**相**

同。（　　）

4）将程序第 25~28 行的 while 循环替换为 do-while 循环（判断条件和循环体不变），则程序对同一合法输入的输出结果**一定不变**。（　　）

- 单选题

5）（**2 分**）若将程序第 13 行改为 `for (int i = 0; i < strlen(t); ++i) s[i] = t[i];`，且已知输入的 type 一定为 1 的情况下，用 n 表示 s 的长度，m 表示 t 的长度，则程序的时间复杂度为（　　）。

A. $\Theta(n+m)$　　B. $\Theta(n+m^2)$

C. $\Theta(n^2+m)$　　D. $\Theta(n^2+\mathrm{m}^2)$

6）给程序分别输入选项（　）的两组输入数据，得到的输出**不同**。

A. 1 ab abc 和 3 ab abc

B. 1 AB ABC 和 3 AB ABC

C. 1 de fgh 和 3 de fgh

D. 1 DE FGH 和 3 DE FGH

3.

```
1   #include <iostream>
2   using namespace std;
3
4   const int INF = 1000000000;
5   #define Front 0
6   #define Back  1
7   #define Left  2
8   #define Right 3
9   #define Up    4
10  #define Down  5
11  int w[6], a[1003][1003];
12  const int way1[] = {Up, Right, Down, Left};
13  const int way2[] = {Up, Front, Down, Back};
14  const int way3[] = {Left, Front, Right, Back};
15  int get_max(int &a, int b) {
16      return a = max(a, b);
17  }
18  int right_rotate(int &u) {
19      for (int i = 0; i < 4; ++ i)
20          if (u == way1[i])
21              return u = way1[(i + 1) % 4];
22      return u;
23  }
24  int front_rotate(int &u) {
25      for (int i = 0; i < 4; ++ i)
```

```
        if (u == way2[i])
            return u = way2[(i + 1) % 4];
    return u;
}
const int anchorX = Up;
const int anchorY = Front;
const int anchorZ = Right;
int find_down(int u, int v) {
    if (u == Down || u == Up) return anchorX ^ (u == Up);
    if (v == Down || v == Up) return anchorY ^ (v == Up);
    for (int i = 0; i < 4; ++ i)
        if (u == way3[i])
            return anchorZ ^ (v == way3[(i + 1) % 4]);
    return -1;
}
int n, m, dp[1003][1003][6][6];
int main() {
    cin >> n >> m;
    for (int i = 0; i < n; ++ i)
        for (int j = 0; j < m; ++ j)
            cin >> a[i][j];
    for (int i = 0; i < 6; ++ i)
        cin >> w[i];
    for (int i = 0; i < n; ++ i)
        for (int j = 0; j < m; ++ j)
            for (int a = 0; a < 6; ++ a)
                for (int b = 0; b < 6; ++ b)
                    dp[i][j][a][b] = -INF;
    dp[0][0][anchorX][anchorY] = a[0][0] * w[Down];
    for (int i = 0; i < n; ++ i)
        for (int j = 0; j < m; ++ j)
            for (int p = 0; p < 6; ++ p)
                for (int q = 0; q < 6; ++ q)
                    if (dp[i][j][p][q] != -INF) {
                        int x = dp[i][j][p][q];
                        int u1 = p, v1 = q;
                        right_rotate(u1);
                        right_rotate(v1);
                        get_max(dp[i][j + 1][u1][v1],
              x + w[find_down(u1, v1)] * a[i][j + 1]);
                        int u2 = p, v2 = q;
                        front_rotate(u2);
                        front_rotate(v2);
                        get_max(dp[i + 1][j][u2][v2],
```

```
                x + w[find_down(u2, v2)] * a[i + 1][j]);
                    }
    int ans = -INF;
    for (int p = 0; p < 6; ++ p)
        for (int q = 0; q < 6; ++ q)
            ans = max(ans, dp[n - 1][m - 1][p][q]);
    printf("%d\n", ans);
    return 0;
}
```

以下程序的输入数据的绝对值均不超过10^3。完成下面的判断题和单选题：

- 判断题

1) **存在**一种合法的输入数据，使得运行程序时，某次 find_down 函数的返回值是 -1。(　　)

2) 该程序的时间复杂度为 $\Theta(n^2m^2)$。(　　)

3) 对于**任意** $u \in [0,6)$，「先执行 front_rotate(u)，再执行 right_rotate(u)」，与「先执行 right_rotate(u)，再执行 front_rotate(u)」，最终 u 的值相同。(　　)

- 单选题

4) 将 anchorX、anchorY、anchorZ **依次**更换为（　）时，对于全部合法数据，与改变之前的输出结果无异。

A. Left、Front、Down
B. Left、Up、Front
C. Left、Down、Back
D. Down、Right、Front

5) （**2 分**）对于以下的输入数据，输出结果为 （　　）。

```
5 5
2 8 15 1 10
5 19 19 3 5
6 6 2 8 2
12 16 3 8 17
12 5 3 14 13
1 1 1 1 1 1
```

A. 95　　B. 97　　C. 94　　D. 103

6) （**2 分**）对于以下的输入数据，输出结果为 （　　）。

```
2 5
2 8 15 3 10
5 19 19 3 5
```

1 2 3 4 5 6

A. 194　　B. 157　　C. 193　　D. 201

三、完善程序（单选题，每小题 3 分，共计 30 分）

1.（**支付问题**）有 n 种纸币，其中第 i 种纸币的面值为 a_i 元。每种纸币只有一张。求能支付多少种金额（不包括 0 元）。数据范围满足 n≤200，a_i 的总和不超过 5000。

试补全程序。

```
1   #include <iostream>
2
3   using namespace std;
4
5   const int MAXN = 210;
6   const int MAXM = 5010;
7   int n, m;
8   int f[MAXM], a[MAXN];
9
10  int main() {
11      cin >> n;
12      for (int i = 1; i <= n; i++) {
13          cin >> a[i];
14          ①;
15      }
16      ②;
17      for (int i = 1; i <= n; i++)
18          ③
19              f[j] = ④;
20      int ans = 0;
21      for (int i = 1; i <= m; i++)
22          if (⑤) ans++;
23      cout << ans;
24      return 0;
25  }
```

1) ① 处应填（　　）。

A. n += a[i]　　B. m += a[i]

C. n = a[i]　　D. m = a[i]

2) ② 处应填（　　）。

A. f[0] = 1　　B. f[1] = 1

C. a[0] = 1　　D. a[1] = 1

3) ③ 处应填（　　）。

A. for (int j = a[i]; j <= n; j++)
B. for (int j = n; j >= a[i]; j--)
C. for (int j = a[i]; j <= m; j++)
D. for (int j = m; j >= a[i]; j--)

4) ④ 处应填（　　）。

A. f[j - 1] + 1　　B. f[j - a[i]] + 1
C. f[j] || f[j - a[i]]　　D. f[j] && f[j - a[i]]

5) ⑤ 处应填（　　）。

A. f[i]　　B. f[i - 1]
C. f[i] == f[i + 1]　　D. f[i] == f[i - 1]

2. （**凑出 17**）给定 $n(1 \le n \le 20)$ 个互不相同的正整数 $a_1, a_2, \ldots a_n (1 \le a_i \le 10^9)$，将之排成一行。你需要在每个 a_i 前加上一个加号(+)或减号(-)，使这 n 个数字组成一个算式。请问是否存在一种添加符号的方案，使该算式的值为 17？如果存在，请输出 Yes，否则输出 No。

例如，给定 $n = 5, a_1 = 1, a_2 = 4, a_3 = 5, a_4 = 9, a_5 = 8$，则 $-a_1 - a_2 + a_3 + a_4 + a_5 = 17$。

提示：使用穷举法解决这个问题。

试补全程序。

```
1   #include <cstdio>
2
3   using namespace std;
4
5   const int maxn = 25;
6   const int aim = 17;
7
8   int n;
9   int a[maxn];
10  bool ans;
11
12  int getBit(const int s, int p) {
13    return ①;
14  }
15
16  int main() {
```

```
17    scanf("%d", &n);
18    for (②) scanf("%d", a + i);
19    for (int s=0, upperBound = ③; s <= upperBound; ++s) {
20      ④;
21      for (int j = 0; j < n; ++j) if (getBit(s, j) == 1) {
22        sum += a[j];
23      } else {
24        ⑤;
25      }
26      if (int(sum) == aim) {
27        ans = true;
28        break;
29      }
30    }
31    printf("%s\n", ans ? "Yes" : "No");
32  }
```

1) ① 处应填（　　）。

A. (s >> p) & 1　　B. (s << p) & 1

C. s & (1 << p) & 1　　D. s & (1 >> p) & 1

2) ② 处应填（　　）。

A. int i = 0; i <= n; ++i　　B. int i = 1; i <= n; ++i

C. int i = 0; i < n; ++i　　D. int i = 1; i < n; ++i

3) ③ 处应填（　　）。

A. 1 << n　　B. (1 << n) | 1

C. (1 << n) + 1　　D. (1 << n) -1

4) ④ 处应填（　　）。

A. int sum = 0　　B. unsigned long long sum = 0

C. unsigned short sum = 0　　D. unsigned int sum = 0

5) ⑤ 处应填（　　）。

A. sum = a[j] + sum　　B. sum = a[j] - sum

C. sum = -a[j] + sum　　D. sum = -a[j] - sum

试题到此结束

广告

1. 第一轮（初赛课程）https://www.luogu.com.cn/contest/79418

2022 年 CSP 第一轮（初赛）课程系统的梳理 CSP J/S 第一轮测试的题型和常考内容，并提供模拟赛和讲评用于查缺补漏。对于希望熟悉第一轮考点、提升第一轮能力的同学均可报名。

本套试题的讲评将在这个课程中获得。此外之前的回放也可以获得。

2. 洛谷语言入门计划·基础算法计划 https://class.luogu.com.cn/course/yugu22rmjc

适用于小学初中生的 NOIP/CSP 的基础算法进阶课程，包括课堂讲授与实验、课后练习答疑解惑与考评环节，完善语言算法知识体系。

3. 基础提高衔接计划 2022 暑期课程报满，欢迎报名 2023 寒假课程。

计划包括集中授课、题单作业布置、定期模拟比赛讲评，巩固算法基础和举一反三能力，目标 CSP-J 高分，为提高级打基础。

4. 洛谷秋令营（基础组·提高组）9 月公开
面向已经掌握基础/进阶算法学员，通过讲题和模拟增加经验，提升 CSP-S 组应试能力。
请关注公众号获得最新的课程资讯。