CCF CSP-J 2019 第 1 轮试题详解

摘　要 CSP-J/S 是 CCF 创办的 CSP（软件能力认证）中面向非专业级的软件能力认证，于 2019 年首次开设，分为 CSP-J（入门级，Junior）和 CSP-S（提高级，Senior）两组，分别进行两轮认证，涉及算法和编程。本文题目来源于 2019 年 10 月 19 日 CCF 首次举办的（CSP-J）入门级认证第一轮试题，本文在解题思路及算法设计做了较为详细的描述和解析。

关键词 题解；分析；算法；代码

The Task Analysis of CCF CSP-J 2019

1 单项选择题

1.1 中国的国家顶级域名是(A)

A.cn　　B. .ch

C. .chn　　D. .China

【解析】域名常识，中国的顶级域名是.cn，china 的缩写。

1.2 二进制数 11 1011 1001 0111 和 01 0110 1110 1011 进行逻辑与运算的结果是(D)

A.01 0010 1000 1011

B.01 0010 1001 0011

C.01 0010 1000 0001

D.01 0010 1000 0011

【解析】考察二进制的逻辑与运算，只有两位同时为“1”，结果才为“1”，其他情况都为 0。所以答案是：01 0010 1000 0011。

1.3 一个 32 位整型变量占用(C)个字节。

A.32　　B.128

C.4　　D.8

【解析】1 Byte(字节) = 8 bit(位)，即 1 个字节占 8 位二进制，计算：32/8=4(字节)，所以 32 位整数变量是占用 4 个字节。

1.4 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)

A.s=a-c;　　B.s=a-b;

C.s=s-c;　　D.s=b-c;

【解析】本题是考察循环结构的 For 语句，s 初始化为 a; for 循环执行 c 次，每次 s 都会-1，一共是减 c 次，所以答案是 s=a-c。

1.5 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)

A.7　B.10　C.6　D.8

【解析】考察二分算法，每次都是折半查找，即对 100 个元素进行折半查找，第一次比较范围缩小到 50，第二次缩小到 25，第三次缩小到 12，第四次缩小到 6，第五次缩小到 3，第六次缩小到 1，最多七次就可以查找到所要元素[1]。

郑德强，男，1970 年生，泉州七中信息技术教师，信息学奥赛教练，现为福建省计算机学会普及工作委员会委员。Email:26532772@qq.com。

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

1.6 链表不具有的特点是(D)

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

【解析】考察数据结构中链表的特性，链表只记录前一个元素和后一个元素，链表在内存中不是连续存储的，所以可以根据需要来分配空间，链表的优势在于定点删除 / 插入元素，因为链表影响的最多就是给定元素的左右的两个链，插入和删除的时候，也不存在移动元素这个说法。但是，它和数组不一样（数组有下标，可随机访问），只能通过链一个一个的找，不能随机访问。

1.7 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？(C)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

【解析】考察组合数学、枚举算法，把整数 8 拆分成 5 个数字之和，允许有 0，我们可以按照非零数字个数进行枚举，1 个：1 种，2 个：4 种，3 个：5 种，4 个：5 种，5 个：3 种，累加起来一共 18 种。

1.8 一棵二叉树如图 1 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(C)

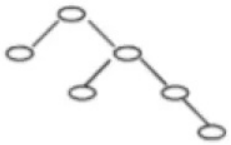

图 1 二叉树

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

【解析】考察数据结构二叉树的存储，据题目描述直接计算，((1*2+1)*2+1)*2+1=15。

1.9 100 以内最大的素数是(B)。

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

【解析】考察素数的判断，可以从大往下排除，得到 97。

1.10 319 和 377 的最大公约数是(C)。

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

【解析】考察约数判断和求最大公约数，377-328=58，58 很显然是所求最大公约数的倍数。所以答案是 29。也可使用辗转相除法：

(319,377)=(319,58)=(58,29) = 29。

1.11 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？(C)

A.3000 B.2500

C.2400 D.2520

【解析】考察贪心算法或枚举算法，设按照方案一、二训练各 i 和 j 天，由题意分析得，3*i+5*j<=21，i+j<=7，i<=3。现要求 300*i+600*j 的最大值。枚举所有情况当 i=2，j=3 时，最大值 2400。

1.12 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(A)张牌的花色一致

A.4 B.2 C.3 D.5

【解析】考察抽屉原理，13 张牌最坏情况就是 4 种花色分别为 3、3、3、4 张，也就是至少 4 张一样花色。

1.13 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，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？(C)

A.60 B.125 C.75 D.100

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

【解析】考察乘法原理，第 1，2 位有 5 种选法(0，1，6，8，9)，第三位有三种 0，1，8，第 4，5 位由前两位决定，所以答案位 5*5*3=75。

1.14 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(B)。

A. ABCDEFGHIJ B. ABDEGHJCFI
C. ABDEGJHCFI D. ABDEGHJFIC

【解析】考察数据结构二叉树遍历，根据中序遍历和后序遍历的特性，后序遍历决定根是 A，中序遍历中看 A 的左边 DBGEH 是左子树，右边 CIF 是右子树，依次类推可画出完整的树。得到前序遍历是 ABDEGHJCFI。

1.15 以下哪个奖项是计算机科学领域的最高奖？(A)

A.图灵奖 B.鲁班奖
C.诺贝尔奖 D.普利策奖

【解析】考察计算机常识，艾伦·麦席森·图灵（Alan Mathison Turing，1912 年 6 月 23 日—1954 年 6 月 7 日），英国数学家、逻辑学家，被称为被称为计算机科学之父，人工智能之父。

2 阅读程序

2.1 问题描述

```
1  #include <cstdio>
2  #include <cstring>
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 }
```

2.2.1 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。 ( )

2.1.2 选择题

（1）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

（2）若输入的字符串长度为( )，那么输入的字符串跟输出的字符申相比，至多有 36 个字符不同。

A.36 B.100000 C.1 D.128

2.1.3 考察的知识点

循环语句及 if 语句的理解，字符数组的存储和遍历，ASCII 编码，约数个数定理的运用。

2.1.4 解题思路

该程序先读入字符串，接着遍历字符串，若 i 是 n 的约数且 st[i-1]的 ASCll 码大于等于 97，则将 st[i-1]变为 st[i-1]-32 的字符（‘A’的 ASCll 码为 65，‘a’ 的 ASCll 码为 97），最后将修改后的字符串输出。即程序可实现将字符串中下标为字符串长度约数的小写字母改为大写字母[2]。

2.1.5 试题解析

（1）考察选手对程序的理解，该程序只是对于字符进行操作，并没有将字符限制于大小写字母，还可能是数字和符号，只要字符不为空格换行都是符合题意的，故答案为×。

（2)若改为“i = 0”，则程序运行到第 9 行“n % i== 0”时出现整数被零除的情况，运行时发生错误，故答案为√。

（3）第 8 行为“i<= n”，则该循环遍历[1,n]中的所有整数，若将第 8 行的“i<=n”改为

“i*i<=n”，则循环遍历[1,$\sqrt{n}$]中的整数，显然后一个范围较小，程序运行结果可能不同，故答案为×。

（4）若输入的字符串全部由大写字母组成，则运行过程中所有字符的 ASCll 码均小于 97，所以字符串不变，原样输出，故答案为√。

（5）假设输入的字符串的所有字母都为小写字母，则改变的字母个数为字符串长度的约数个数，若输入字符串长度为 18，因为 18 的因数有：1、2、3、6、9、18 一共 6 个数，则最多 6 个位置被修改，故选 B。

（6）输入的字符串中，被修改的字符数最多为 n 的因数个数，依次观察 4 个选项，其中 36= 22 □ 32 ，计有(2+1) □(2+1)=9 个因数，100000= 25 □ 55 ，共有(5+1)□ (5+1)=36 个因数，1 只有一个因子，128= 27，共有 8 个因数，故选择 B。

2.2 问题描述

```
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 }
```

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：

2.2.1 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。(　　)

2.2.2 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

2.2.3 考察的知识点

程序的模拟，特殊值的构造。

2.2.4 解题思路

这道题是这样的一个模型，有 2n 个点，X1,X2……Xn 和 Y1,Y2……Yn。加入 m 条边，每条边只能在 X 和 Y 之间相连，且每个点只连一条边，若 Xi 与 Yj 和 Yk(j<k)之间都有边相连，则只保留 Xi 与 Yk 这条边。答案为寻找没有连边的点的个数。

阅读程序，该程序对于初始为 0 的数组，读入 m 组 x、y，若 a[x]被操作过则将 b[a[x]]赋为 0，若 b[y]被操作过则将 a[b[y]]赋为 0，再将 a[x]赋为 y，b[y]赋为 x，统计两数组中 0 的个数。

2.2.5 试题分析

（1）当 m>0 时，则 m 至少为 1，至少有两个位置不为 0，则 ans<2n，可以知道 a 数组和 b 数组中一定会有值被修改，故答案为√。

（2）可以举出反例，当 n=2，m=2，x1=1，y1=1，

x2=1，y2=2 时，最后的结果数组为 a[ ]={2,0}，b[ ]={0,1}，当 i=1 时运行到 27 行的“++ans”时, ans=1，被修改的位置的个数为奇数，故答案为×。

（3）当 n=1，m=1，x1=1，y1=1 时，则有 a[1]=b[1]=1>0，故答案为×。

（4）当输入的数据为

3 2

1 2

1 3

程序的第 15 行被执行，故答案为×。

（5）当 m 个 x 两两不同,且 m 个 y 两两不同，则对于每次前修改 a[x]=b[y]=0，修改后 a[x]=y，b[y]=x，一次修改 2 个位置，m 次后值为 0 的位置剩余(2n-2m)个。故选择 **A**。

（6）当 m 个 x 两两不同,且 m 个 y 都相等，则每次修改的 b 数组的位置都相同，只有当新的 x 满足 x>b[y]时，可能进行修改，修改时 a[b[y]]=0，则前一次对于 a 数组的修改效果被消除，变为 a[x]=y，即在这种情况下，只有 x 最大的那次修改能对 a 数组产生影响，则 a，b 数组共有两个位置受到影响，故答案为 ans=2n-2，选择 **A**。

2.3 问题描述

```
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 I, int r, int depth) {
8       if (l > r)
9           return 0;
10      int min = maxn, mink;
11      for (int i = I; 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  }
```

2.3.1 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。(　　)

2.3.2 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.6000　　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≤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≤i<n,都有 b[i]=1, 那么输出最小为(　　)

A.582　　B.580　　C.579　　D.581

2.3.3 考察的知识点

对分治算法、递归程序的理解，二叉树，程序复杂度计算。

2.3.4 解题思路

F 函数本质是构筑一棵二叉树，序列中最小的数做二叉树的根，左边的子序列做左子树，右边的树做右子树，然后反复做此操作，直到构筑出整棵树。程序对 a 数组进行分治，每次执行时找到当前区间的最小值的下标，将这个下标为界分成左右两边再分别计算值， 最后加上递归层数乘上 b[mink]进行回溯，每个子树以该区间的最小值点为根，

答案就为□dep[i]*b[i]（dep[i]为 i 点在树上的深度）。

2.3.5 试题分析

（1）观察程序可知，若 a 中有重复数字，每次都会取下标较靠前的那个最小值，且该程序中存在判断超界的“if (l>r) return 0;”，所以不会运行时错误，故答案为×。

（2）由分析可知，答案为□dep[i]*b[i] 。若 b[i]全为 0，则对于函数 f 的每一个返回值一定都为 0，故答案为√。

（3）当 n=100 时，最坏情况下该数组 a 为单调递增，每次都必须从当前区间的第一个位置比较到最后一个位置，每次比较都要更新 min 和 mink 的值，则需要判断 100+99+98+…+2+1=5050，最接近 5000，故选择 **A**。

（4）当 n=100 时，最好情况下，该数组每个区间的最小值都位于区间的中点，这样层数最小，每次都可以将 a 数组平均分为两份。此时构建出的二叉树高度为 log2(n)，总复杂度为 n*log2(n),当 n=100 时，比较运算执行的次数为 100*log2(100)≈600 次，故选择 **D**。

（5）当 n=10，0<=i<n，b[i]=i+1 时，要使得答案最大，则 b[i]越大对应深度越大越好。构造 a[i]为一个递增数组，则 0<=i<n，时，i 的深度由 1 到 n 依次递增，则 1*b[0]+2*b[1]+…+9*b[8]+10*b[9]=1*1+2*2+3*3+…+10*10=385，故选 **D**。

（6）当 n=100,0<=i<n,b[i]=1 时，答案最少的情况下递归树为一颗完全二叉树，设根节点（原数组）为第 0 层，则第 1 层有 1 个划分点，第 2 层有 2 个划分点，第 3 层有 4 个划分点，第 n 层有 2n-1 个划分点，最后一层即第 7 层剩下 100-(27-1)=37 个节点，通过模拟运算，可以计算出函数最终返回值为 1*1+2*2+4*3+…+32*6+37*7=580，故选择 **B**。

3 完善程序

3.1 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二进制位一一进行比较，若两个二进制相同，则运算结果的对应二进制位位 0，反之为 1。

```
#include <cstdio>
using namespace std;
int n;
const int max_size = 1<<10;

int res[max_size][max_size];

void recursive(int x,int y,int n,int t){
if (n==0){
res[x][y] = ①
return;
 }
int step = 1<< (n - 1);
recursive( ②, n - 1, t);
recursive( x, y + step, n - 1, t);
recursive( x + step, y, n – 1, t);
recursive( ③, n – 1,!t);
 }

int main ( ){
scanf("%d",&n);
```

```
recursive( 0,   0, ④ );
int size = ⑤;
for ( int i = 0; i < size; ++i ) {
    for ( int j = 0; j < size; ++j )
        printf( "%d", res[i][j]);
    puts( "");
    }
    return 0;
}
```

1) ①处应填（　）

A. n % 2　　B. 0　　C. t　　D. 1

2) ②处应填（　）

A．x – step, y – step

B. x, y – step

C. x – step, y　　D. x, y

3) ③处应填（　）

A．　x – step, y – step

B.　x + step, y + step

C.　x – step, y

D.　x, y – 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 << (n + 1)　　B.　1 << n

C. n + 1　　D.　1 << (n - 1)

3.1.1 试题解析

本题是考察分治法，通过递归的方式对矩阵进行填充，从最大的开始递归往下做。每次将当前矩阵划分为 4 个子矩阵，对应一次变换，根据变换规则得到 4 个子矩阵的元素。

（1）语句中，x，y 指当前矩阵的左上角，n 是指阶数，这里如果选择常数的话，那整个矩阵就成了常数矩阵，此处赋值和 n 没有必然联系，所以选择答案 C 。

（2）分成 4 个子矩阵，参数分别是（x，y）、（x，y+step）、（x+step，y）、（x+step，y+step），只有 x，y 坐标加与不加 step 两种可能，没有减的可能，所以选择答案 D 。

（3）同上，4 个子矩阵中，第二个是右上，第三个是左下，第四个就是右下，（x+step，y+step），所以选择答案 B 。

（4）第一次调用 recursive 函数，在这里 n 是矩阵规模，初始为 n，t 是取反次数，所以 t 初始为 0 或者 1，所以选择答案 B。

（5）在这里 size 是输出矩阵的边长，最终大小是 2 的 n 次幂，位运算的结果就是 1<<n。所以选择答案 B。

3.2 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，对 n 对 10000 以内的整数，从小达到排序。

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关键字。从小到大排序后输出。

数据范围 $1 \le n \le 107$, $1 \le a[i],b[i] \le 10^4$。

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。

```
#include <cstdio>
# include <cstring>
using namespace std;
const int maxn = 10000000
const int maxs = 10000

int n;
    unsigned a[maxn], b[maxn], res[maxn], ord[maxn];
unsigned cnt[maxs + 1];

int main( ) {
    scanf(" %d "&n);
    for (int i = 0; i < n; ++i)
        scanf(" %d%d ", &a[i], &b[i]);
    memset(cnt, 0, sizeof(cnt));
    for (int i =0; i < n; ++i )
      ①;  // 利用 cnt 数组统计数量
    for (int i = 0; i < maxs; ++i )
        cnt[i + 1] += cnt[i];
    for (int i = 0; i < n; ++i)
      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  }
```

3.2.1 解答

（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]]]

3.2.2 试题解析

本题是考察计数排序，和桶排序的原理类似。核心是先统计每个 x 值出现的次数，再计算小于等于每个 x 值得元素个数 t，t 的值决定了排序时值为 x 的元素应该从第 t 位开始逐个往前排序，完成单关键字排序。而双关键字排序，要先按第二关键字排序后，再根据第一关键字重新排序，对于第一关键字相同的元素，按照第二关键字排序，最后的结果就是双关键字排序。

（1）根据程序可知，先对第二关键字排序，即按照 b 数组排序，只能用 b 数组做关键字，所以答案选择 B。

（2）按照第二关键字排序，答案还是与 b[i]有关，cnt[b[i]]表示第 i 个数按第二关键字排的位，所以答案选择 D。

（3）在这里，要对第一关键字进行计数，答案就与 a[i]有关，所以答案选择 C。

（4）同（2），res[i]是记录第一关键字第 i 个数的原位置，数组 ord 存的是元素编号，所以答案选择 A。

（5）要输出最终顺序的 a 与 b，肯定与 res 或者 ord 有关，而数组 res 和 ord 存的都是元素编号，肯定不能嵌套使用，所以答案选择 B。

4 总结

2019CCF 非专业级别软件能力认证第一轮（CSP-J)入门级的试题，延续了以往 NOIP 出题风格，题目具有一定的思维难度，考察的知识点也很全面，注重考察原理。学生对于一些题目，即使无法将知识点理解透彻，也可以根据上下文的线索推导出答案。本文的提供的解题思路和分析可能不够全面，仅供读者学习和参考。

参 考 文 献

[1] 郑德强.再谈信息学竞赛活动中激发学生的乐学精神.福建电脑, 2016,32(10):182-183

[2] 吴文虎.信息学奥林匹克竞赛指导.北京:清华大学出版社出版,2004