2019 CCF 非专业级别软件能力认证第一轮

（CSP-J）入门级 C++ 语言试题参考答案与解析

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. 中国的国家顶级域名是（ ）

A. .cn　　B. .ch　　C. .chn　　D. .china

答案：A

考点：计算机基础－计算机网络－域名

解析：典型的国家顶级域名有.cn（中国）、.us（美国）、.uk（英国）、.jp（日本）、.sg（新加坡）等。典型的通用顶级域名有.edu（教育机构）、.gov（政府部门）、.net（网络组织）、.com（商业组织）、.org（非营机构）、.mil（军事部门）等。

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（ ）。

A. 01 0010 1000 1011　　B. 01 0010 1001 0011

C. 01 0010 1000 0001　　D. 01 0010 1000 0011

答案：D

考点：计算机基础－数制与编码－二进制－位运算

解析：逐位进行与运算。对于每一位，0 与 0 得 0，1 与 0 得 0，0 与 1 得 0，1 与 1 得 1。

3. 一个 32 位整型变量占用（ ）个字节。

A. 32　　B. 128　　C. 4　　D. 8

答案：C

考点：计算机基础－计算机体系结构－内存空间

解析：8 位是 1 字节，因此 32 位是 4 字节。在 C++语言中，int 是最常用的带符号 32 位整型变量，可表示数值$[-2^{31}, 2^{31}-1]$，unsigned int 是最常用的无符号 32 位整型变量，可表示数值$[0, 2^{32}-1]$。

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. s = a - c;　　B. s = a - b;　　C. s = s - c;　　D. s = b - c;

答案：A

考点：程序设计基础－C++语法基础－循环语句

解析：s 初始化为 a，紧接着 for 循环 c 次，每次 s 减 1，因此该程序段相当于 s=a-c。

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. 7　　B. 10　　C. 6　　D. 8

答案：A

考点：程序设计基础 - 算法与数据结构 - 折半查找（二分查找）

解析：对 100 个有序元素进行折半查找，每次查找可将检索范围缩小一半。由 $2^6-1<100<=2^7-1$ 可知，最大比较次数为 7。

6. 链表不具有的特点是（ ）

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

答案：D

考点：程序设计基础 - 算法与数据结构 - 链表

解析：链表是通过记录每个元素的后继位置来实现数据存储，所需空间与元素个数成正比，优点是不必事先估计存储空间、插入或删除指定位置元素的时间复杂度为 O(1)；但缺点是由于其元素的内存地址不连续，无法进行 O(1)的随机访问。

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？（ ）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

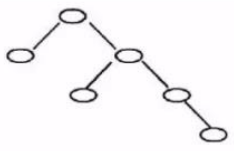
A. 22　　B. 24　　C. 18　　D. 20

答案：C

考点：数学 - 计数问题

解析：枚举法求解，8 个同样的球分 1 个袋子共 1 种方案，分 2 个袋子共 4 种方案，分 3 个袋子共 5 种方案，分 4 个袋子共 5 种方案，分 5 个袋子共 3 种方案，合计 18 种。

8. 一棵二叉树如右图所示，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（ ）。



A. 6　　B. 10　　C. 15　　D. 12

答案：C

考点：程序设计基础 - 算法与数据结构 - 树结构

解析：根据题目给定的规则可知，下标最大的结点为树中深度最大且最靠右的结点，其下标为((1*2+1)*2+1)*2+1=15。

9. 100 以内最大的素数是（ ）。

A. 89　　B. 97　　C. 91　　D. 93

答案：B

考点：数学 - 素数与合数

解析：98 - 100 均为合数，97 为素数。

10. 319 和 377 的最大公约数是（　）。

A. 27　　B. 33　　C. 29　　D. 31

答案：C

考点：数学 - 公约数与公倍数

解析：使用辗转相除法可得 GCD(319,377)=GCD(319,58)=GCD(58,29)=29。或将两数分解质因数后，提取公共部分亦可求解。

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？（　）

A. 3000　　B. 2500　　C. 2400　　D. 2520

答案：C

考点：程序设计基础 - 算法与数据结构 - 枚举算法

解析：设方案 1 执行 x 天，方案 2 执行 y 天，则有 3x+5y<=21、x+y<=7、y<=3。要求 300x+600y 的最大值，枚举可得最优方案为 x=2、y=3，此时 300x+600y 为 2400。或使用线性规划亦可求解。

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（　）张牌的花色一致。

A. 4　　B. 2　　C. 3　　D. 5

答案：A

考点：数学 - 抽屉原理

解析：最坏情况，13 张牌对应四种花色的牌数为 3、3、3、4。

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 9，9 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？（　）

A. 60　　B. 125　　C. 75　　D. 100

答案：C

考点：数学 - 乘法原理

解析：前 2 位有 0、1、8、6、9 共 5 种选择，第 3 位只能放 0、1、8，后 2 位由前 2 位决定，因此总方案数为 5*5*3*1*1=75。

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（　）。

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

答案：B

考点：程序设计基础 - 算法与数据结构 - 树结构

解析：后序遍历的规则是“左右根”、中序遍历的规则是“左根右”，因此可知，A 是树根、DBGEHJ 是 A 左子树的中序遍历（对应后续遍历 DGJHEB）、CIF 是 A 右子树的中序遍历（对应后续遍历 IFC），递归画出对应的二叉树，再根据前序遍历规则“根左右”即可求出答案。

15. 以下哪个奖项是计算机科学领域的最高奖？（ ）

A. 图灵奖　　B. 鲁班奖　　C. 诺贝尔奖　　D. 普利策奖

答案：A

考点：计算机基础 - 常识 - 重要人物

解析：图灵奖由美国计算机协会于 1966 年设立，其名称取自计算机科学之父图灵，专门奖励对计算机事业作出重要贡献的个人，被誉为“计算机界的诺贝尔奖”。

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 1.5 分，选择题 4 分，共计 40 分）

程序一

```
#include <cstdio>
#include <cstring>
using namespace std;
char st[100];
int main() {
  scanf("%s", st);
  int n = strlen(st);
  for (int i = 1; i <= n; ++i) {
    if (n % i == 0) {
      char c = st[i - 1];
      if (c >= 'a')
        st[i - 1] = c - 'a' + 'A';
    }
  }
  printf("%s", st);
  return 0;
}
```

考点：程序设计基础 - 算法与数据结构 - 字符串

概述：程序用于将字符串下标（如果从 1 开始编号，但 C++语言中实际是从 0 开始编号）是 n 约数的对应小写字母转换为大写字母。

- 判断题

1) 输入的字符串只能由小写字母或大写字母组成。（ ）

答案：错

解析：输入的字符串也可以包含数字等其他字符。

2）若将第 8 行的“`i = 1`”改为“`i = 0`”，程序运行时会发生错误。（ ）

答案：对

解析：若 i 可以为 0，则第 9 行的 if 语句条件“n%i==0”将发生运行时错误 RE。

3）若将第 8 行的“`i <= n`”改为“`i * i <= n`”，程序运行结果不会改变。（ ）

答案：错

解析：当第 8 行的循环条件为“i<=n”时，字符串的末尾字符会被程序加工，但若改为“i*i<=n”，字符串的末尾字符将不会被程序加工（除非字符串长度为 1）。

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。（ ）

答案：对

解析：大写字母的 ASCII 编码值小于小写字母的。若输入的字符串全部由大写字母组成，则程序不会对其进行加工。

- 选择题

5）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

答案：B

解析：18 的正约数共有 6 个，因此程序至多修改输入字符串中的 6 个字符，即输出字符串与输入字符串至多有 6 个字符不同。。

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

A. 36　　B. 100000　　C. 1　　D. 128

答案：B

解析：根据程序的作用可知，要使输出字符串和输入字符串之间至多有 36 个字符不同，36 应当是字符串长度 n 的约数个数。本题选项中，仅有 100000 满足要求，将其分解质因数得 $100000=2^5*5^5$，得其的正约数共有(5+1)*(5+1)=36 个。

程序二

```cpp
#include <cstdio>
using namespace std;
int n, m;
int a[100], b[100];

int main() {
  scanf("%d%d", &n, &m);
  for (int i = 1; i <= n; ++i)
    a[i] = b[i] = 0;
  for (int i = 1; i <= m; ++i) {
    int x, y;
    scanf("%d%d", &x, &y);
    if (a[x] < y && b[y] < x) {
      if (a[x] > 0)
        b[a[x]] = 0;
      if (b[y] > 0)
        a[b[y]] = 0;
      a[x] = y;
      b[y] = x;
    }
  }
  int ans = 0;
  for (int i = 1; i <= n; ++i) {
    if (a[i] == 0)
      ++ans;
    if (b[i] == 0)
      ++ans;
  }
  printf("%d\n", ans);
  return 0;
}
```

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：

考点：程序设计基础 - 算法与数据结构 - 模拟算法

概述：程序可以视作通过模拟算法选出一系列互不冲突的数对——若数对(x_1, y_1)和(x_2, y_2)之间有$x_1==x_2$或$y_1==y_1$，则认为这两个数对存在冲突，现按顺序考虑每一个数对(x_i, y_i)（要求满足前提：在已经选用的数对中，与左值x_i匹配的右值小于y_i、与右值y_i匹配的左值小于x_i），若该数对与此前已经选用的数对冲突，则用当前数对替换所冲突的原数对；若无冲突，则直接选用当前数对。程序中的 a[x]用于记录，在已选用的数对中，与左值 x 相匹配的右值；b[y]用于记录，在已选用的数对中，与右值 y 相比配的左值；n 表示数对左值和右值的取值范围为[1,n]；最后的 ans 用于统计剩余多少左值或右值，没有相应数对被我们选中。

- 判断题

1) 当 m>0 时，输出的值一定小于 2n。（ ）

答案：对

解析：由限定条件 0<x,y<=n 可知，当 m>0 时，一定存在某个数对被我们选中，此时 ans<2n

2) 执行完第 27 行的“++ans”时，ans 一定是偶数。（ ）

答案：错

解析：由于数对是一个左值与一个右值相匹配，因此 ans 最终一定是偶数。但第 27 行的“++ans”在第 23 行的 for 循环的内部，其中间结果可能为奇数。

3) a[i]和 b[i]不可能同时大于 0。（ ）

答案：错

解析：a[i]用于记录与左值 i 相匹配的右值，不存在则为 0；b[i]用于记录与右值 i 相匹配的左值，不存在则为 0。当存在数对(i,y)和(x,i)都被我们选中时，a[i]和 b[i]就会同时大于 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。（ ）

答案：错

解析：存在反例——依次考虑数对(1,2)(1,3)时，第 15 行程序会被执行。

- 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

答案：A

解析：此时，输入的数对两两互不冲突，因此程序会将它们全部选中，根据上述 ans 的意义可知，其结果为 2n-2m。

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

答案：A

解析：此时，输入的数对两两存在冲突，因此程序最终只会选用一个数对，根据上述 ans 的意义可知，其结果为 2n-2。

程序三

```cpp
#include <iostream>
using namespace std;
const int maxn = 10000;
int n;
int a[maxn];
int b[maxn];
int f(int l, int r, int depth) {
  if (l > r)
    return 0;
  int min = maxn, mink;
  for (int i = l; i <= r; ++i) {
    if (min > a[i]) {
      min = a[i];
      mink = i;
    }
  }
  int lres = f(l, mink - 1, depth + 1);
  int rres = f(mink + 1, r, depth + 1);
  return lres + rres + depth * b[mink];
}
int main() {
  cin >> n;
  for (int i = 0; i < n; ++i)
    cin >> a[i];
  for (int i = 0; i < n; ++i)
    cin >> b[i];
  cout << f(0, n - 1, 1) << endl;
  return 0;
}
```

考点：程序设计基础 - 算法与数据结构 - 树结构

概述：程序可以视作根据二叉树的中序遍历，构造一棵满足要求的树，并输出各结点深度与 b 值的加权和——要求二叉树的根结点最小，并递归要求左右子树的根结点最小（除非相应子树为空）。

- 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。（ ）

答案：错

解析：若 a 数组有重复数字，则程序在根据 a 数组递归构造符合要求的二叉树时，对于相同结点值，会优先考虑位于左侧的。

2) 如果 b 数组全为 0，则输出为 0。（ ）

答案：对
解析：程序最终输出的是各结点深度与 b 值的加权和，因此若 b 数组全为 0，则加权和显然为 0。

- 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. 600　　C. 6　　D. 100

答案：A
解析：最坏情况下，程序所构造的二叉树的每个结点至多仅有一个子结点，此时，程序将递归 100 层，其中第 i 层进行 100-i+1 次第 12 行的比较运算，总执行次数为 100+99+98+…+1≈5000。

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

答案：D
解析：最佳情况下，程序构造二叉树时，对于每个结点会尽可能均分其左右子树。定义根结点深度为 1，则含 n=100 个结点的树的深度最小为 logn≈7，此时每选定一层结点，程序都需要执行约 n 次的第 12 行的比较运算，因此总执行次数约为 nlogn≈600。

5）当 n=10 时，若 b 数组满足，对任意$0 \le i < n$，都有$b[i] = i + 1$，那么输出最大为（ ）。

A. 386　　B. 383　　C. 384　　D. 385

答案：D
解析：此时，要使输出的 ans 值尽可能大，程序所构造的二叉树的深度应尽可能地大。定义根结点深度为 1，则含 10 个结点的二叉树的最大深度为 10，因此 ans 的最大值为 1*1+2*2+3*3+…+10*10=385。

6）（4 分）当 n=100 时，若 b 数组满足，对任意$0 \le i < n$，都有$b[i] = 1$，那么输出最小为（ ）。

A. 582　　B. 580　　C. 579　　D. 581

答案：B
解析：此时，要使输出的 ans 值尽可能小，程序应参照完全二叉树构造此树，其中深度为 1 的结点共 1 个，深度为 2 的结点共 2 个，深度为 3 的结点共 4 个……深度为 6 的结点共 32 个，剩余 37 个结点的深度为 7，因此 ans 的最小值为 (1*1+2*2+3*4+…+6*32)+7*37=580。

三、完善程序（单选题，每题 3 分，共计 30 分）

程序一

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的二进制位一一进行比较，若两个二进制位相同，则运算结果的对应二进制位为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;
}
```

考点：程序设计基础 - 算法与数据结构 - 分治算法

概述：程序采用分治算法，递归模拟矩阵的变换过程。递归函数 recursive(x,y,n,t)表示计算左上角(x,y)，大小 2^n*2^n，由单个数字 t 变幻而来的矩阵。

1) ①处应填（ ）

A. n % 2　　B. 0　　C. t　　D. 1

答案：C

解析：此处为递归边界，当需要计算的是单位矩阵时，相应元素应赋值为 t，即无需再经任何变换。

2) ②处应填（ ）

A. `x - step, y - step`　　B. `x, y - step`

C. `x - step, y`　　D. `x, y`

答案：D

解析：左上角(x,y)，且大小 2^n*2^n 的矩阵，可以分成 4 个 $2^{n-1}*2^{n-1}$ 的矩阵分别计算。此处需要计算的是 4 个矩阵中位于左上方的矩阵，该矩阵的左上角坐标为(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`

答案：B

解析：左上角(x,y)，且大小 2^n*2^n 的矩阵，可以分成 4 个 $2^{n-1}*2^{n-1}$ 的矩阵分别计算。此处需要计算的是 4 个矩阵中位于右下方的矩阵，该矩阵的左上角坐标为 $(x+2^{n-1}, y+2^{n-1})$。

4) ④处应填（ ）

A. `n - 1, n % 2`　　B. `n, 0`

C. `n, n % 2`　　D. `n - 1, 0`

答案：B

解析：此处是递归计算的入口，即题目最终所求的是大小 2^n*2^n，由单个数字 0 变幻而来的矩阵，因此递归函数的后两个参数应设为 n 和 0。

5) ⑤处应填（ ）

A. `1 << (n + 1)`　　B. `1 << n`

C. `n + 1`　　D. `1 << (n - 1)`

答案：B

解析：此处是计算最终所求的矩阵大小，即边长 size 为 2^n，位运算写做“1<<n”。

程序二

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，将 n 对 10000 以内的整数，从小到大排序。

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键字。
从小到大排序后输出。

数据范围 $1 \le n \le 10^7, 1 \le a[i], b[i] \le 10^4$。

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。

```cpp
#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
  memset(cnt, 0, sizeof(cnt));
  for (int i = 0; i < n; ++i)
    ③;   // 利用 cnt 数组统计数量
  for (int i = 0; i < maxs; ++i)
    cnt[i + 1] += cnt[i];
  for (int i = n - 1; i >= 0; --i)
    ④;   // 记录最终排序结果
  for (int i = 0; i < n; ++i)
    printf("%d %d\n", ⑤);
  return 0;
}
```

考点：程序设计基础 - 算法与数据结构 - 排序算法

解析：基于计数排序，程序实际实现了近似于基数排序的算法——依次以低位到高位的每一位数为关键词，进行计数排序，前一次的排序结果是下一次的初始序列——本题对应着先根据第二关键词 b 进行计数排序，再根据第一关键词 a 进行计数排序。

1) ①处应填（ ）

A. `++cnt[i]`

B. `++cnt[b[i]]`

C. `++cnt[a[i] * maxs + b[i]]`

D. `++cnt[a[i]]`

答案：B
解析：此处是根据第二关键词 b 进行计数排序，并做各关键词的数量统计工作，因此将 b[i]对应的元素数量自增 1。

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`

答案：D
解析：此处是根据第二关键词 b 进行计数排序，并记录排序结果。此时的 cnt[key]用于表示关键词为 key 的元素在结果数组中的位置，因此这里的程序应将关键词为 b[i]的元素 i 放在 ord 数组里。

3) ③处应填（ ）

A. `++cnt[b[i]]`
B. `++cnt[a[i] * maxs + b[i]]`
C. `++cnt[a[i]]`
D. `++cnt[i]`

答案：C
解析：此处是根据第一关键词 a 进行计数排序，并做各关键词的数量统计工作，因此将 a[i]对应的元素数量自增 1。

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]`

答案：A
解析：此处是根据第一关键词 a 进行计数排序，并记录排序结果。由于此前已经根据第二关键词 b 进行计数排序，此时第 i 个元素的原始下标实际为 ord[i]，因此这里的程序应将关键词为 a[ord[i]]的元素 ord[i]放在 res 数组里。

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]]]`

答案：B
解析：此处是按顺序输出排序结果。由于此前已经按照第二关键词、第一关键词完成了计数排序，此时第 i 个元素的原始下标实际为 res[i]。