

CSP-J 2020 第一轮试题详解

林荣辉

(福建省福州第八中学 福州 350004)

摘要 本文面向 CSP 软件能力测试 J 组选手, 针对题型设置和知识分布, 对 CSP2020-J 组的测试做详细分析。

关键词 信息学; CSP 2020; J 组; 普及组; 奥林匹克; C++; 算法

中图法分类号 TP399 DOI:10.16707/j.cnki.fjpc.2020.11.068

The Task Analysis of CSP-J 2020 Round 1

LIN Ronghui

(Fuzhou No.8 Middle School, Fuzhou, China, 350004)

1 试卷分析

CSP2020-J 组第一试试卷在内容上综合考察考生对计算机基础和常用算法的理解, 同时在数据结构和排列组合模块进行了小范围拓展; 在考点分布上, 整张试卷较为分散, 以第一部分单选题为例, 该部分试题既考察考生的语言理解水平, 也检测于考人员的编程技巧, 整体难度适中, 该部分细目表如表 1 所示:

表 1 CSP2020-J1 选择题细目表

单选序号	考察知识点	单选序号	考察知识点
1	计算机基础	2	编译器
3	逻辑运算	4	图像容量
5	冒泡排序	6	递归
7	链表特性	8	简单图论
9	进制转换	10	排列组合
11	栈的特征	12	数据结构
13	求余模拟	14	排列组合
15	排列组合		

第二部分一阅读程序则包含字符串处理、进制转换和数学运算, 该部分试题对考生的程序分析能力提出了进一步的要求。第三部分——完善程序部分在题面上提供了逻辑清晰的问题描述, 但在程序编写经验上为初学者设置了门槛。

2 单项选择题分析

(1) 在内存存储器中每个存储单元都被赋予一个唯一的序号, 称为(A)。

A. 地址 B. 序号 C. 下标 D. 编号

解析: 与街道地址相似, 内存存储器中的每个存储单元(即字节 Byte)都对应一个二进制地址且唯一, 可以将其近似看作连续的数组。每当地址总线中出现该地址的信息, 系统即选定这个二进制内存单元进行读写操作(即寻址操作)。

(2) 编译器的主要功能是(A)。

A. 将源程序翻译成机器指令代码

B. 将源程序重新组合

C. 将低级语言翻译成高级语言

D. 将一种高级语言翻译成另一种高级语言

解析: 编译器是一种用于代码转换的计算机程序, 它的主要功能是将一种语言编写的源代码转换为机器指令目标代码^[1]。

(3) 设 $x=true$, $y=true$, $z=false$, 以下逻辑运算表达式值为真的是(D)。

A. $(y \vee z) \wedge x \wedge z$ B. $x \wedge (z \vee y) \wedge z$

C. $(x \wedge y) \wedge z$ D. $(x \wedge y) \vee (z \vee x)$

解析: \vee “析取”是逻辑或的意思; \wedge “合取”是逻辑与的意思。考虑到 z 为 false, 所以任何与 z

有八运算的皆为假/false。

(4) 现有一张分辨率为 2048×1024 像素的 32 位真彩色图像。请问要存储这张图像, 需要多大的存储空间? (C)。

A. 16MB B. 4MB C. 8MB D. 32MB

解析: 图像存储空间=横向分辨率×纵向分辨率×颜色位数÷8, 此时得到的单位为 Byte, 需再换算为 MB 单位^[2]。本题完整计算公式:

$$\frac{2048 \times 1024 \times 32}{8 \times 1024 \times 1024} = \frac{2^{11} \times 2^{10} \times 2^5}{2^3 \times 2^{10} \times 2^{10}} = 2^3 = 8$$

(5) 冒泡排序算法的伪代码如下:

输入: 数组 L, $n \geq 1$ 。

输出: 按非递减顺序排序的 L。

算法 BubbleSort:

FLAG \leftarrow n //标记被交换的最后元素位置

while FLAG>1 do

k \leftarrow FLAG-1

FLAG \leftarrow 1

for j=1 to k do

if L(j) > L(j+1) then do

L(j) \leftrightarrow L(j+1)

FLAG \leftarrow j

对 n 个数用以上冒泡排序算法进行排序, 最少需要比较多少次? (C)。

A. n2 B. n-2 C. n-1 D. n

解析: 冒泡排序是一种交换排序, 它的思路是: 两两比较相邻记录的数值, 如果与序列不符则交换数值, 持续访问记录直到没有反序的情况为止。如果序列原本即呈有序状态, 则只需比较 n-1 次就好 (即一轮完成)。

(6) 设 A 是 n 个实数的数组, 考虑下面的递归算法^[1-4]:

XYZ (A[1..n])

if n=1 then return A[1]

else temp \leftarrow XYZ (A[1..n-1])

if temp < A[n]

then return temp

else return A[n]

请问算法 XYZ 的输出是什么? (B)。

A. A 数组的平均 B. A 数组的最小值
C. A 数组的中值 D. A 数组的中值

解析: 递归的特性可以总结为自己调用自己。递归必须包含两个条件: 递归出口(终止递归的条件, 即临界值)递归表达式(即调用自己的传导规

律)。本题递归出口为 1, 直接返回自己 (即 A[1]), 其他情况则不断维护 temp 与 A[n] 的比较序列, 始终保持最小值。

(7) 链表不具有的特点是(A)。

A. 可随机访问任一元素
B. 不必事先估计存储空间
C. 插入删除不需要移动元素
D. 所需空间与线性表长度成正比

解析: 链表是一种线性存储结构, 与顺序存储不同, 每个链表节点存放的是到下一个节点的指针(Pointer)。由于不必按顺序存储, 链表的插入和删除操作可以达到 O(1) 的复杂度。以图 1 为例, 单链表需并行判断结点的前驱与后继, 才能确定元素的位置。



图 1 单链表示例

(8) 有 10 个顶点的无向图至少应该有(A)条边才能确保是一个连通图。

A. 9 B. 10 C. 11 D. 12

解析: 图(graph)是一种描述多对多关系的数据结构^[1-4]。如果给图的每条边规定一个方向, 则我们将这样的图称为有向图, 反之则称为无向图。如果图中任意两点都是连通的, 则图被称作连通图。例如, 图 2 无向图有 5 个顶点, 则至少应该有 4 条边才能确保是一个连通图。

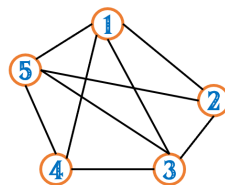


图 2 连通图

(9) 二进制 1011 转换成十进制数是(A)。

A. 11 B. 10 C. 13 D. 12

解析: 进制转换遵循按权相加原则, 这里的权就是源进制的位权, 所以本题计算方法为: $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$, 或凭借 8421 法 $8+2+1$ 也能得到结果。

(10) 五个小朋友并排站成一列, 其中有两个小朋友是双胞胎, 如果要求这两个双胞胎必须相邻, 则有(A)种不同排列方法?

A. 48 B. 36 C. 24 D. 72

解析: 先将双胞胎看作整体, 则总方案数为 A_4^4 ,

再观察双胞胎的左右状态, 这个状态也是有序的, 最后根据乘法原理答案为 $A_4^4 \times A_2^2 = 48$ 。

(11) 图 3 中所使用的数据结构是(A)。

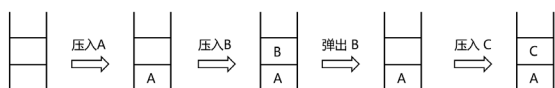


图 3 线性表操作

A. 栈 B. 队列 C. 二叉树 D. 哈希表

解析: 栈是一种线性表, 栈的特性是只在表尾(即栈顶)进行插入和删除操作。本题图呈现“先进后出”的顺序关系以及压入、弹出的操作特性, 这就是一个典型的栈^[3]。

(12) 独根树的高度为 1。具有 61 个结点的完全二叉树的高度为(D)。

A. 7 B. 8 C. 5 D. 6

解析: 高度为 n 的满二叉树的结点数为 $2^n - 1$, 完全二叉树的节点数小于等于满二叉树。本题 61 个结点处于 $2^5 - 1$ 和 $2^6 - 1$ 之间。

(13) 干支纪年法是中国传统的纪年方法, 由 10 个天干和 12 个地支组合成 60 个天干地支。由公历年份可以根据以下公式和表格换算出对应的天干地支。如图 4 所示。

天干 = (公历年份) 除以 10 所得余数

地支 = (公历年份) 除以 12 所得余数

天干	甲	乙	丙	丁	戊	己	庚	辛	壬	癸		
	4	5	6	7	8	9	0	1	2	3		
地支	子	丑	寅	卯	辰	巳	午	未	申	酉	戌	亥
	4	5	6	7	8	9	10	11	0	1	2	3

图 4 干支纪年与公历换算表

例如, 今年是 2020 年, 2020 除以 10 余数为 0, 查表为“庚”; 2020 除以 12 余数为 4, 查表为“子”, 所以今年是庚子年。

请问 1949 年的天干地支是 (C)

A. 己酉 B. 己亥 C. 己丑 D. 己卯

解析: 依题意模拟并与表对照即可。

$1949 \% 10 = 9$, 对应汉字己。

$1949 \% 12 = 5$, 对应汉字丑。

(14) 10 个三好学生名额分配到 7 个班级, 每个班级至少有一个名额, 一共有(A)种不同的分配方案。

A. 84 B. 72 C. 56 D. 504

解析: 本题适用插板法: 将 10 个三好生排为一列, 这样他们之间就产生了 9 个空位, 考虑到每

个班级至少要一个名额, 我们用假想的 6 个隔板插入 9 个空位中, 即 $C_9^6 = 84$ 。

(15) 有五副不同颜色的手套(共 10 只手套, 每副手套左右手各 1 只), 一次性从中取 6 只手套, 请问恰好能配成两副手套的不同取法有(A)种。

A. 120 B. 180 C. 150 D. 30

解析: 从五套任意取出两副手套方案为 C_5^2 , 剩余的手套中任取两支的方案为 C_6^2 , 扣除剩余 3 套当中取 1 套其中正好取到同一副的情况为 C_3^1 , 整合方案为 $C_5^2 \times (C_6^2 - C_3^1) = 120$ 。

3 阅读程序分析

(1)

```
#include <cstdlib>
#include <iostream>
using namespace std;
char encoder[26]={'C','S','P',0};
char decoder[26];
string st;
int main(){
    int k=0;
    for (int i = 0; i < 26; ++i)
        if (encoder[i] != 0) ++k;
    for (char x = 'A'; x <= 'Z'; ++x){
        bool flag = true;
        for (int i = 0; i < 26; ++i)
            if (encoder[i] == x){
                flag = false;
                break;
            }
        if (flag) {
            encoder[k] = x;
            ++k;
        }
    }
    for (int i = 0; i < 26; ++i)
        decoder[encoder[i] - 'A'] = i + 'A';
    cin >> st;
    for (int i = 0; i < st.length(); ++i)
        st[i] = decoder[st[i] - 'A'];
    cout << st;
    return 0;
}
```

● 判断题

输入的字符串应当只由大写字母组成，否则在访问数组时可能越界。(√)

若输入的字符串不是空串，则输入的字符串与输出的字符串一定不一样。(×)

将第 12 行的 “i < 26” 改为 “i < 16”，程序运行结果不会改变。(√)

将第 26 行的 “i < 26” 改为 “i < 16”，程序运行结果不会改变。(×)

● 单选题

若输出的字符串为 “ABCABCABCA”，则下列说法正确的是 (A)。

- A. 输入的字符串中既有 S 又有 P
- B. 输入的字符串中既有 S 又有 B
- C. 输入的字符串中既有 A 又有 P
- D. 输入的字符串中既有 A 又有 B

若输出的字符串为 “CSPCSPCSPCSP”，则下列说法正确的是 (D)。

- A. 输入的字符串中既有 P 又有 K
- B. 输入的字符串中既有 J 又有 R
- C. 输入的字符串中既有 J 又有 K
- D. 输入的字符串中既有 P 又有 R

解析：本题主要实现字符串加密，代码中的两个字符数组 encoder 与 decoder 分别对应编码与解码。主要功能是用字符 ‘C’、‘S’、‘P’ 替换字母表中的 ‘A’、‘B’、‘C’，其余的字符向后移动。如表 2 所示，encodr 列即为加密状态;根据 “decoder [encoder [i] - ‘A’] = i + ‘A’” 的规则解密为 decoder 状态。当输入的字符串 st 经过加密，从 decoder 相应得到解密后的 st 字符串。反之从 decoder 的字母，也能比照出 cin 的字母。

于是：

判断 1：如果字符对应的 ASCII 码比大写字母 A 小（即减去 ‘A’），则下标可能会出现负值，最终导致越界，所以下标应限制在大写字母范围内，故本描述正确。

判断 2：参照表 2，加密与解密序列中大写字母 T 之后的字母完全一致，即输入 “UVWXYZ” 输出也是 “UVWXYZ”，故本描述有误。

判断 3：初始状态加密只对应 3 个有效字母，条件只需满足 i 不大于 3 即可，最终结果不受影响，故本描述正确。

判断 4：遍历（循环）范围需包含所有的大写字母，故本描述有误。

单选 1：依据表 2，输出 “ABC” 对应输入 “CSP”，

满足既有 S 又有 P，故本题选 A。

单选 2：同上表，输出 “CSP” 对应输入 “PRN”，满足既有 P 又有 R，故本题选 D。

表 2 加解密字母对照表

i	cin	decoder	encoder
0	A	D	C
1	B	E	S
2	C	A	P
3	D	F	A
4	E	G	B
5	F	H	D
6	G	I	E
7	H	J	F
8	I	K	G
9	J	L	H
10	K	M	I
11	L	N	J
12	M	O	K
13	N	P	L
14	O	Q	M
15	P	C	N
16	Q	R	O
17	R	S	Q
18	S	B	R
19	T	T	T
20	U	U	U
21	V	V	V
22	W	W	W
23	X	X	X
24	Y	Y	Y
25	Z	Z	Z

(2)

```
#include <iostream>
using namespace std;
long long n, ans;
int k, len;
long long d[1000000];
int main() {
    cin >>n >>k;
    d[0] = 0;
    len = 1;
    ans = 0;
    for ( long long i = 0; i < n; ++i ) {
        ++d [ 0 ];
        for (int j = 0 ; j + 1 < len ; ++j ) {
            if( d [ j ] == k ){
```

```

        d[j] = 0;
        d[j+1] += 1;
        ++ans;
    }

    if (d[len-1] == k) {
        d[len-1] = 0;
        d[len] = 1;
        ++len;
        ++ans;
    }
    cout << ans << endl;
    return 0;
}

```

假设输入的 n 是不超过 2^{62} 的正整数, k 都是不超过 10000 的正整数, 完成下面的判断题和单选题:

● 判断题

若 $k=1$, 则输出 ans 时, $len=n$ 。(F)

若 $k>1$, 则输出 ans 时, len 一定小于 n 。(F)

若 $k>1$, 则输出 ans 时, k^{len} 一定大于 n 。(T)

● 单选题

若输入的 n 等于 10^{15} , 输入的 k 为 1, 则输出等于(D)。

- A. 1 B. $(10^{30}-10^{15})/2$
 C. $(10^{30}+10^{15})/2$ D. 10^{15}

若输入的 n 等于 205, 891, 132, 094, 649(即 3^{30}), 输入的 k 为 3, 则输出等于(B)。

- A. 3^{30} B. $(3^{30}-1)/2$
 C. $3^{30}-1$ D. $(3^{30}+1)/2$

若输入的 n 等于 100, 010, 002, 000, 090, 输入的 k 为 10, 则输出等于(D)。

- A. 11,112,222,444,543 B. 11,122,222,444,453
 C. 11,122,222,444,543 D. 11, 112, 222, 444,453

解析: 本题是进制转换^[4]问题, 模拟 k 进制的第 n 个数。即询问将某数字进行进制转换后的进位计数。代码中 len 表示当前进位数, ans 计数器记录发生了几次进位。

判断 1: 应该为第 2 位, 即 len 为 2, 故本描述有误。

判断 2: 题意为判断 k 进制的位数是否一定小于 n , 以 2 为例, 2 的二进制是 10, 它的位数是 2, $len(2 \text{ 位})$ 并不小于 n , 故本题不正确。

判断 3: 一个 k 进制数, 如果有 len 位, 每一位有 k 种变化, 一共能表示 k^{len} 种数值, 数值是从 0 到 $k^{len}-1$ 的。 n 就是在 0 到 $k^{len}-1$ 之间的, 故本描

述正确。

单选 1: 输入数为 1 的情况, 每个 i 会发生一次进位, 共循环 n 次, 因此共产生 n 次进位, 故本题选 D。

单选 2: 进位按照从左到右的顺序正好构成一个等比数列, 输出为 $(3^i-1)/2$, 即等比级数求和公式, 选 B。

单选 3: 个位总共产生 $n/10^1$ 次进位, 百位总共产生 $n/10^2$ 次进位……以此类推并相加, 总进位数为 11112222444453 次进位, 故选 D。

(3)

```

#include <algorithm>
#include <iostream>
using namespace std;
int n;
int d[50][2];
int ans;
void dfs(int n, int sum) {
    if (n == 1) {
        ans = max(sum, ans);
        return;
    }
    for (int i = 1; i < n; ++i) {
        int a = d[i-1][0], b = d[i-1][1];
        int x = d[i][0], y = d[i][1];
        d[i-1][0] = a + x;
        d[i-1][1] = b + y;
        for (int j = i; j < n-1; ++j)
            d[j][0] = d[j+1][0], d[j][1] = d[j+1][1];
        int s = a + x + abs(b - y);
        dfs(n-1, sum + s);
        for (int j = n-1; j > i; --j)
            d[j][0] = d[j-1][0], d[j][1] = d[j-1][1];
        d[i-1][0] = a, d[i-1][1] = b;
        d[i][0] = x, d[i][1] = y;
    }
}
int main() {
    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> d[i][0];
    for (int i = 0; i < n; ++i)
        cin >> d[i][1];
    ans = 0;
    dfs(n, 0);
    cout << ans << endl;
    return 0;
}

```

假设输入的 n 是不超过 50 的正整数, $d[i][0]$ 、 $d[i][1]$ 都是不超过 10000 的正整数, 完成下面的判断题和单选题:

● 判断题

若输入 n 为 0, 此程序可能会死循环或发生运行错误。(F)

若输入 n 为 20, 接下来的输入全为 0, 则输出为 0。(T)

输出的数一定不小于输入的 $d[i][0]$ 和 $d[i][1]$ 的任意一个。(F)

● 单选题

若输入的 n 为 20, 接下来的输入是 20 个 9 和 20 个 0, 则输出为 (B)。

A. 1890 B. 1881 C. 1908 D. 1917

若输入的 n 为 30, 接下来的输入是 30 个 0 和 30 个 5, 则输出为 (C)。

A. 2000 B. 2010 C. 2030 D. 2020

若输入的 n 为 15, 接下来的输入是 15 到 1, 以及 15 到 1, 则输出为 (C)。

A. 2440 B. 2220 C. 2240 D. 2420

解析: 本题类似“石子合并”, 针对一个 2 列的二维数组, 不断选出相邻的两行进行合并, 直到合并为 1 行为止。结果为每次合并时相邻两行第 0 列值之和与第 1 列之差的绝对值, 最后合并求最值。其中, ans 为和最大的方案, 本题主要考察考生的搜索功底。

判断 1: 若 n 为 0 程序直接返回主函数结束, 不会报错, 本描述有误。

判断 2: 数组元素都为 0 的情况, 和 sum 也是 0, 本描述正确。

判断 3: 可以举个反例 2 1 1 10000 10000, 故本描述有误。

单选 1: 近似贪心思想, 忽略第 1 列, 简化分析, 每次让刚刚合并过的行继续合并, 会得到更大的 sum , 即 $(9+9)+(9+9+9)+(9+9+9+9)+\dots$, 答案为最后一项为 20 个 9 相加, 故选 B。

单选 2: 带入 n 较小的情况可以发现规律, 如 $n=4$ 时加到 $2*5$, 可以推出 $n=30$ 时加到 $28*5$, 即 $0+5+10+\dots+28*5=(28*29/2)=2030$, 故选 C。单选 1, 增加一个差的计算。

单选 3: 计算量很大, 从左往右贪心, 得出 $2*14*15*16/3=2240$, 故选 C。

4 完善程序分析

(1) (质因数分解) 给出正整数 n , 请输出将 n 质因数分解的结果, 结果从小到大输出。

例如: 输入 $n=120$, 程序应该输出 2 2 2 3 5, 表示 $120=2\times 2\times 2\times 3\times 5$ 。

输入保证 $2\leq n\leq 10^9$ 。

提示: 先从小到大枚举变量 i , 然后用 i 不停试除 n 来寻找所有的质因子。试补全程序。

```
#include <stdio>
using namespace std;
int n, i;
int main() {
    scanf("%d", &n);
    for(i = ①; ② <= n; i++) {
        ③{
            printf("%d", i);
            n = n / i; }
        }
    if( ④ )
        printf("%d", ⑤ );
    return 0; }
```

①处应填(C)

A. 1 B. n-1 C. 2 D. 0

②处应填(C)

A. n/i B. $n/(i*i)$ C. $i*i$ D. $i*i*i$

③处应填(C)

A. $\text{if}(n \% i == 0)$ B. $\text{if}(i * i \leq n)$
C. $\text{while}(n \% i == 0)$ D. $\text{while}(i * i \leq n)$

④处应填(A)

A. $n > 1$ B. $n \leq 1$ C. $i < n/i$ D. $i + i \leq n$

⑤处应填(C)

A. 2 B. n/i C. n D. i

解析: 本题依靠枚举, 输出质因数分解的结果。
选择 1: 枚举质因子最小值以 2 为起点, 故选 C。
选择 2: 质因子范围访问至 \sqrt{n} 即可, 无需完整遍历, 故选 C。
选择 3: 考虑因子相同的情况, 不断迭代直到不能被 i 整除为止, 故选 C。
选择 4: 如果 $n > 1$, 就表示因子未被完全除尽, 故选 A。
选择 5: 最后还剩下 n 未输出, 所以输出 n , 选 C。

(2) (最小区间覆盖) 给出 n 个区间, 第 i 个区间的左右端点是 $[a_i, b_i]$ 。现在要在这些区间中选出若干个, 使得区间 $[0, m]$ 被所选区间的并覆盖 (即每一个 $0 \leq i \leq m$ 都在某个所选的区间中)。保证答案存在, 求所选区间个数的最小值。

输入第一行包含两个整数 n 和 m ($1 \leq n \leq 5000$,

$1 \leq m \leq 10^9$ 。

接下来 n 行, 每行两个整数 a_i, b_i ($0 \leq a_i, b_i \leq m$)。

提示: 使用贪心法解决这个问题。先用 $O(n^2)$ 的时间复杂度排序, 然后贪心选择这些区间。试补全程序。

```
#include <iostream>
using namespace std;
const int MAXN = 5000;
int n, m;
struct segment { int a, b; } A[MAXN];
void sort() // 排序
{
    for (int i = 0; i < n; i++)
        for (int j = 1; j < n; j++)
            if ( ① ) {
                segment t = A[j];
                ②
            }
}
int main() {
    cin >> n >> m;
    for (int i = 0; i < n; i++)
        cin >> A[i].a >> A[i].b;
    sort();
    int p = 1;
    for (int i = 1; i < n; i++)
        if ( ③ )
            A[p++] = A[i];
    n = p;
    int ans = 0, r = 0;
    int q = 0;
    while (r < m) {
        while ( ④ )
            q++;
        ⑤;
        ans++;
    }
    cout << ans << endl;
    return 0;
}
```

①处应填(B)

- A. $A[j].b > A[j-1].b$
- B. $A[j].a < A[j-1].a$
- C. $A[j].a > A[j-1].a$
- D. $A[j].b < A[j-1].b$

②处应填(D)

- A. $A[j+1] = A[j]; A[j] = t;$
- B. $A[j-1] = A[j]; A[j] = t;$
- C. $A[j] = A[j+1]; A[j+1] = t;$
- D. $A[j] = A[j-1]; A[j-1] = t;$

③处应填(A)

- A. $A[i].b > A[p-1].b$
- B. $A[i].b < A[i-1].b$
- C. $A[i].b > A[i-1].b$
- D. $A[i].b < A[p-1].b$

④处应填(A)

- A. $q+1 < n \ \&\& \ A[q+1].a \leq r$
- B. $q+1 < n \ \&\& \ A[q+1].b \leq r$
- C. $q < n \ \&\& \ A[q].a \leq r$
- D. $q < n \ \&\& \ A[q].b \leq r$

⑤处应填(B)

- A. $r = \max(r, A[q+1].b)$
- B. $r = \max(r, A[q].b)$
- C. $r = \max(r, A[q+1].a)$
- D. $q++$

解析:

选择 1: 对区间内的第一个元素进行冒泡排序, 依据题意进行升序排序, 如果右边的起点小于左边的起点则交换, 故选 B。

选择 2: 交换顺序, 通过第三方变量中继传递, 即将 $A[j]$ 传递给 t , 然后将 $A[j-1]$ 赋值给 $A[j]$, 最后 t 的值再传递给 $A[j-1]$ 选 D。

选择 3: 依据贪心策略, 当目前区间的第二个值比第 p 个区间的第二个值大的情况才保留记录, 故选 A。

选择 4: 变量 q 从 0 开始, 当 $q+1$ 小于 n 时, 才能确保第 $q+1$ 个区间有值, 同时只有在 $q+1$ 个区间的第一个数小等于 r 时, 这个区间才起用, 选 A。

选择 5: 当前存储的最右边的区间和第 q 个区间的第二个元素取最大值, 故选 B。

参考文献

- [1] 王晓东. 计算机算法设计与分析. 第 5 版. 北京: 电子工业出版社, 2018
- [2] 刘汝佳. 算法竞赛入门经典. 第 2 版. 北京: 清华大学出版社, 2014
- [3] 王晓东. 数据结构: C 语言描述. 第 3 版. 北京: 电子工业出版社, 2019
- [4] 林厚从. 信息学奥赛课课通 (C++) . 北京: 高等教育出版社, 2018