

2020 CCF 非专业级别软件能力认证第一轮

(LGR--10) 洛谷模拟试题试卷

认证时间：2020 年 10 月 8 日 09:30-11:30

考生注意事项：

- 试题纸共有 9 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. 十进制数 114 的相反数的 8 位二进制补码是：
A. 10001110 B. 10001101 C. 01110010 D. 01110011
2. 以下哪个网站不是 Online Judge（在线程序判题系统）？Online Judge 可以查看算法题目，提交自己编写的程序，然后可以获得评测机反馈的结果。
A. Luogu B. Gitee C. LeetCode D. Codeforces
3. 小 A 用字母 A 表示 1，B 表示 2，以此类推，用 Z 表示 26。对于 27 以上的数字，可以用两位或者更长的字符串来对应，例如 AA 对应 27，AB 对应 28，AZ 对应 52，AAA 对应 703……那么 BYT 字符串对应的数字是什么？
A. 2018 B. 2020 C. 2022 D. 2024
4. UIM 拍摄了一张照片，其分辨率是 4096×2160 ，每一个像素都是 24 位真彩色。在没有压缩的情况下，这张图片占用空间接近以下哪个值？
A. 8MB B. 25MB C. 200MB D. 200KB
5. 在一个长度为 n 的数组中找到第 k 大的数字，平均的算法时间复杂度最低的是：
A. $O(n)$ B. $O(nk)$ C. $O(n \log n)$ D. $O(n^2)$
6. 对于“树”这种数据结构，正确的有：
①一个有 n 个顶点、 $n-1$ 条边的图是树
②一个树中的两个顶点之间有且只有一条简单路径
③树中一定存在度数不大于 1 的顶点
④树可能存在环
A. ①②④ B. ①②③ C. ②③ D. ①②

7. 博艾中学进行了一次信息学会考测试，其优、良、及格、不及格的试卷数量分别为 10、13、14、5 张。现在这些卷子混在一起，要将这些卷子按照等级分为 4 叠。分卷子的方法是，每次将一叠有不同等级答卷的卷子分为两堆，使得这两堆中没有相同等级的卷子，然后可以再分，直到分为 4 叠。要分完这些卷子，至少需要多少次“分卷子”的操作？将一堆数量为 n 的卷子分成两堆，就会产生 n 次分卷子的操作。
- A. 84 B. 93 C. 78 D. 85
8. 一个二叉树的前序遍历是 HGBDAFEC，中序遍历是 BGHFAEDC，同时采用顺序存储结构，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. 7 B. 13 C. 15 D. 12
9. 在 C++ 语言中，如果 $a=1; b=0; c=1$ ；那么以下表达式中为 1 的是：
- A. $a\&\&b || b\&\&c$ B. $a+b>c || b$ C. $!(c\&\&(!a || b))$ D. $a+b+c$
10. 在一个初始长度为 n 的链表中连续进行 k 次操作，每次操作是读入两个数字 a_i 和 b_i ，在链表中找到元素为 a_i 的结点（假设一定可以找到），然后将 b_i 这个元素插入到这个结点前面。在最理想的情况下，链表访问的结点数量最少可能是多少（不算将要插入的结点）？
- A. n 次 B. k 次 C. nk 次 D. $n+k$ 次
11. A 班有 5 名风纪委员，B 班有 4 名风纪委员，C 班有 3 名风纪委员。现在需要这些同学中选取 6 名风纪委员巡逻，如果只关注各班派出的风纪委员人数，有几种不同的方案？
- A. 9 B. 12 C. 15 D. 18
12. 以下哪种排序算法的时间复杂度是 $O(n^2)$ ？
- A. 计数排序 B. 插入排序 C. 希尔排序 D. 归并排序
13. 已知 $\text{rand}()$ 可以生成一个 0 到 32767 的随机整数，如果希望得到一个范围在 $[a, b)$ 的随机整数， a 和 b 均是不超过 100 的正整数且 $a < b$ ，那么可行的表达式是什么？
- A. $(\text{rand}() \% (b-a)) + a$ B. $(\text{rand}() \% (b-a+1)) + a$
C. $(\text{rand}() \% (b-a)) + a+1$ D. $(\text{rand}() \% (b-a+1)) + a+1$
14. 一个 7 个顶点的完全图需要至少删掉多少条边才能变为森林？
- A. 16 B. 21 C. 15 D. 6
15. 2020 年 8 月，第（）届全国青少年信息学奥林匹克竞赛在（）举行？
- A. 26，广州 B. 26，长沙 C. 37，广州 D. 37，长沙

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.

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

● 判断题

- 1) (1分) luo 函数中，m 的值不可能是奇数。()
- 2) (1分) 若将第 11 行的“<”改为“<=”，程序的输出结果可能会改变。()
- 3) 若将第 8、9、13 行删除，程序的运行的结果不变。()
- 4) 在添加合适的头文件后，将第 19 到 21 行替换为“memset(gu, 255, sizeof(gu));”可以起到相同的作用。()

● 选择题

- 5) (4分) 若输入数据为 4 8，则输出为 ()。
A. 7 B. 8 C. 15 D. 16
- 6) 最坏情况下，此程序的时间复杂度是 ()。
A. $O(m^2n)$ B. $O(nm!)$ C. $O(n^2)$ D. $O(n^2m)$

2.

```
1  #include<cstdio>
2  using namespace std;
3  int n, m;
4  int f[101][101];
5  int F[101][101];
6  int main() {
7      scanf("%d%d", &n, &m); // n 的值在 1 到 100 之间
8      memset(f, -1, sizeof(f));
9      for(int i = 1; i <= m; i++) {
10         int u, v, w; // w 的值在 0 到 10000 之间
11         scanf("%d%d%d", &u, &v, &w);
12         f[u][v] = f[v][u] = w;
13     }
14     for(int k = 1; k <= n; k++)
15         for(int i = 1; i <= n; i++)
16             for(int j = 1; j <= n; j++)
17                 if(f[i][k] != -1 && f[k][j] != -1)
18                     if(f[i][j] == -1 || f[i][j] > f[k][j] + f[i][k])
19                         f[i][j] = f[i][k] + f[k][j];
20     int ans = 2147483647;
21     for(int i = 1; i <= n; i++)
22         for(int j = 1; j <= n; j++) {
23             for(int x = 1; x <= n; x++)
24                 for(int y = 1; y <= n; y++)
25                     F[x][y] = f[x][y];
26             F[i][j] = F[j][i] = 0;
27             for(int x = 1; x <= n; x++)
28                 for(int y = 1; y <= n; y++)
29                     if(F[x][y] == -1 || F[x][y] > F[x][i] + F[i][y])
30                         F[x][y] = F[x][i] + F[i][y];
31             for(int x = 1; x <= n; x++)
32                 for(int y = 1; y <= n; y++)
33                     if(F[x][y] == -1 || F[x][y] > F[x][j] + F[j][y])
34                         F[x][y] = F[x][j] + F[j][y];
35             int res = 0;
36             for(int x = 1; x <= n; x++)
37                 for(int y = 1; y < x; y++)
38                     res += F[x][y];
39             ans = min(res, ans);
40     }
```

```

41     printf("%d\n", ans);
42     return 0;
43 }

```

● 判断题

- 1) (1 分) 14 到 16 行，将外层到内层的循环变量依次调整为 i、j、k，程序的运行的结果不变。()
- 2) 这个程序的时间复杂度和 m 无关。()
- 3) 20 行的 ans 如果初始化为 10^7 时，可能无法得到正确结果。()
- 4) 若将第 27 到 30 行的部分和 31 到 34 行的两个部分互换，程序的运行的结果不变。()

● 选择题

5) 若输入数据为 4 5/1 2 3/1 3 6/2 3 4/2 4 7/3 4 2 (其中“/”为换行符)，则输出为 ()。

- A. 14 B. 18 C. 21 D. 28

6) 这个程序使用了 () 算法。

- A. Floyd B. Dijkstra C. Prim D. Kruskal

3.

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define MOD 19260817
4  #define MAXN 1005
5  long long A[MAXN][MAXN] = {0}, sum[MAXN][MAXN] = {0};
6  int n, m, q;
7  int main() {
8      A[1][1] = A[1][0] = 1;
9      for(int i = 2; i <= 1000; i++) {
10         A[i][0] = 1;
11         for(int j = 1; j <= i; j++)
12             A[i][j] = (A[i - 1][j] + A[i - 1][j - 1]) % MOD;
13     }
14     for(int i = 1; i <= 1000; i++)
15         for(int j = 1; j <= 1000; j++)
16             sum[i][j] = (sum[i - 1][j] + sum[i][j - 1]
17                 - sum[i - 1][j - 1] + A[i][j] + MOD) % MOD;
18     int q;
19     cin >> q;
20     while(q--) {
21         int n, m;
22         cin >> n >> m;

```

```

23         cout << sum[n][m] << endl;
24     }
25     return 0;
26 }

```

● 判断题

- 1) (1分) 当 $i \leq j$ 时, $A[i][j]$ 的值是 0。 ()
- 2) 当 $i > j$ 时, $A[i][j]$ 的值相当于从 i 个不同元素中取出 j 个元素的排列数。 ()
- 3) $sum[i][j]$ 的值 ($1 < j < i \leq 1000$) 不小于 $sum[i-1][j-1]$ 的值。 ()
- 4) 若将第 12 行改为 “ $A[i][j] = (A[i-1][j] + A[i-1][j-1] + MOD) \% MOD$;”, 程序的运行的结果不变。 ()

● 选择题

- 5) (4分) $A[i][j]$ ($1 \leq i \leq 10, 1 \leq j \leq 10$) 的所有元素中, 最大值是 ()。
 A. 126 B. 276 C. 252 D. 210
- 6) 若输入数据为 1/5 3 (其中 “/” 为换行符), 则输出为 ()。
 A. 10 B. 35 C. 50 D. 24

三、完善程序 (单选题, 每小题 3 分, 共计 30 分)

1. (封禁 xxs) 现有 n 个 xxs (编号为 1 到 n), 每个 xxs 都有一个关注者, 第 i 个 xxs 的关注者是 a_i 。现在管理员要将其中的一些 xxs 的账号封禁, 但需要注意的是如果封禁了第 i 个人, 那么为了不打草惊蛇, 就不能封禁他的关注者 a_i 。现在想知道最多可以封禁多少个 xxs。

输入第一行是一个不超过 300000 的整数 n , 第二行是 n 个 1 到 n 的整数表示 a_i 。

输出一行, 一个整数表示答案。

```

1  #include <cstdio>
2  using namespace std;
3  #define MAXN 300005
4  int n, ans = 0, a[MAXN], in[MAXN] = {0};
5  bool vis[MAXN] = {0};
6  void dfs(int cur, int w) {
7      if(vis[cur])
8          return;
9      vis[cur] = true;
10     if(w == 1) ans++;
11     ①

```

```

12         if(②)
13             dfs(a[cur], ③);
14     }
15     int main() {
16         scanf("%d", &n);
17         for(int i = 1; i <= n; i++) {
18             scanf("%d", &a[i]);
19             in[a[i]]++;
20         }
21         for(int i = 1; i <= n; i++)
22             if(!in[i]) ④;
23         for(int i = 1; i <= n; i++)
24             if(⑤) dfs(i, 0);
25         printf("%d\n", ans);
26         return 0;
27     }

```

1) ①处应填 ()

- A. a[cur] = cur;
- B. in[a[cur]] = 0;
- C. in[a[cur]]--;
- D. in[cur]--;

2) ②处应填 ()

- A. in[a[cur]] != 0 || w == 1
- B. in[a[cur]] == 0 || w == 0
- C. in[a[cur]] != 0 || w == 0
- D. in[a[cur]] == 0 || w == 1

3) ③处应填 ()

- A. 0
- B. 1
- C. w
- D. 1 - w

4) ④处应填 ()

- A. dfs(i, 1)
- B. dfs(i, 0)
- C. dfs(a[i], 1)
- D. dfs(a[i], 0)

5) ⑤处应填 ()

- A. !in[i]

- B. in[i]
- C. !vis[i]
- D. vis[i]

2. (烧作业) 某课作业布置了 N ($3 \leq N \leq 100000$) 个题目, 第 i 题对应的得分是 a_i 。作业的总得分的计算方式为去掉作业中得分最小的一个题, 剩下其它所有题目得分的平均值。但很不幸小 A 遇到了一场火灾, 前 K ($1 \leq K \leq N-2$) 个题目被烧了, 无法记录得分。小 A 想知道, K 是多少时, 可以得到最高的作业得分? 作业被烧了前 K 页, 这时的得分是从第 $K+1$ 页到最后一页中, 去除最小得分后取平均值。

输入第一行是整数 N , 第二行是 n 个不超过 10000 的非负整数表示 a_i 。

输出一行, 若干个整数表示答案。如果有多个 K , 请依次升序输出。

```
1  #include <cstdio>
2  #include <cmath>
3  #define min(a,b) (a<b?a:b)
4  #define MAXN 100002
5  using namespace std;
6  int n, k[MAXN], cnt = 0;
7  int s[MAXN], minScore, sum;
8  double maxAverage = 0, nowAverage;
9  int main() {
10     scanf("%d", &n);
11     for(int i = 1; i <= n; i++)
12         scanf("%d", &s[i]);
13     minScore = s[n];
14     ①;
15     for(int i = n - 1; i >= 2; i--) {
16         minScore = min(minScore, s[i]);
17         ②;
18         nowAverage = ③;
19         if(nowAverage > maxAverage) {
20             ④
21             maxAverage = nowAverage;
22         } else if(fabs(nowAverage - maxAverage) < 1e-6)
23             ⑤;
24     }
25     for(int i = cnt; i >= 1; i--)
26         printf("%d\n", k[i]);
27     return 0;
28 }
```


1) ①处应填 ()

- A. `sum = n`
- B. `sum = s[1]`
- C. `sum = s[n]`
- D. `sum = 0`

2) ②处应填 ()

- A. `sum = maxAverage * (n-i)`
- B. `sum += s[i]`
- C. `sum += s[n - i]`
- D. `sum = s[i] + minScore`

3) ③处应填 ()

- A. `(double)(sum + minScore)/(n - i)`
- B. `sum * 1.0 / (n - i)`
- C. `(int)(sum - minScore)/(n - i)`
- D. `(double)(sum - minScore)/(n - i)`

4) ④处应填 ()

- A. `k[++cnt] = i;`
- B. `k[cnt++] = i - 1`
- C. `cnt = 1; k[cnt] = i - 1;`
- D. `cnt = 0; k[cnt] = i;`

5) ⑤处应填 ()

- A. `k[cnt++] = i;`
- B. `k[++cnt] = i - 1;`
- C. `k[cnt++] = n - i;`
- D. `k[cnt] = i;`