CSP-J 初赛模拟卷（第 17 套）

入门级 C++ 语言试题

一、单项选择题（共 **15** 题，每题 **2** 分，共计 **30** 分）

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，以此来推，用 Z 表示 26。对于27以上的数，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 (nlogn)　　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)，然后将这b_i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．(rand()%(b-a))+a　　B．(rand()%(b-a+1))+a
C．(rand()%(b-a))+a+1　D．(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 **×**；共 **18** 小题，共计 **40** 分，各小题分值见题目）

(1)

```
01 #include<iostream>
02 using namespace std;
03 #define MAXN 20
04 int gu[MAXN][MAXN];
05 int luo(int n, int m) {
06     if(n <= 1 || m < 2)
07         return 1;
08     if(gu[n][m] != -1)
09         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

16. (1分) luo函数中，m的值不可能是奇数。（ ）

17. (1分) 若将第11行的“<”改为“<=”，程序的输出结果可能会改变。（ ）

18. (2分) 若将第8、9、13行删除，程序的运行的结果不变。（ ）

19. (2分) 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。（ ）

单选题

20. (4分) 若输入数据为 4 8，则输出为（ ）。

A. 7　　B. 8　　C. 15　　D. 16

21. (3分) 最坏情况下，此程序的时间复杂度是（ ）。

A. $O(m^2n)$　　B. $O(mn!)$　　C. $O(n^2)$　　D. $O(n^2m)$

(2)

```
01 #include<cstdio>
02 #include<cstring>
03 using namespace std;
04 int n, m;
05 int f[101][101];
06 int F[101][101];
07 int main(){
08     scanf("%d%d",&n,&m);//n的值在1到100之间
09     memset(f,-1,sizeof(f));
10     for(int i = 1; i <=  m; i++) {
11         int u,v,w;//w的值在0到10000之间
12         scanf("%d%d%d",&u,&v,&w);
13         f[u][v]=f[v][u]=w;
14     }
15     for(int k = 1; k <= n; k++)
16         for(int i=1;i <=n; i++)
17             for(int j=1;j<= n; j++)
18                 if(f[i][k]!=-1&&f[k][j]!=-1)
19                     if(f[i][j]==-1||f[i][j]>f[k][j]+f[i][k])
20                         f[i][j]=f[i][k]+f[k][j];
21     int ans=2147483647;
22     for(int i=1;i<=n;i++)
23         for(int j=1;j<=n; j++){
24             for(int x=1;x<=n; x++)
25                 for(int y=1;y<=n; y++)
26                     F[x][y]=f[x][y];
27             F[i][j]=F[j][i]=0;
28             for(int x=1;x<= n; x++)
29                 for(int y=1;y<=n; y++)
30                     if(F[x][y]==-1||F[x][y]>F[x][i]+F[i][y])
31                         F[x][y]=F[x][i]+F[i][y];
32             for(int x=1;x<=n; x++)
33                 for(int y=1;y<=n; y++)
```

```
34                      if(F[x][y]==-1||F[x][y]>F[x][j]+F[j][y])
35                          F[x][y]=F[x][j]+F[j][y];
36              int res=0;
37              for(int x=1;x<=n; x++)
38                  for(int y=1;y<x; y++)
39                      res +=F[x][y];
40              ans = res < ans ? res : ans;
41          }
42      printf("%d\n",ans);
43      return 0;
44  }
```

判断题

22. (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。（ ）

23. (2分) 这个程序的时间复杂度和m无关。（ ）

24. (2分) 20行的ans如果初始化为10^7时，可能无法得到正确结果。（ ）

25. (2分) 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。（ ）

单选题

26. (3分) 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

27. (3分) 这个程序使用了（ ）算法。

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

(3)

```
01  #include<bits/stdc++.h>
02  using namespace std;
03  #define MOD 19260817
04  #define MAXN 1005
05  long long A[MAXN][MAXN]={0},sum[MAXN][MAXN]={0};
06  int n, m, q;
07  int main(){
08      A[1][1]=A[1][0]=1;
```

```
09      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

28. (1分) 当i<=j时，A[i][i]的值是0。（ ）

29. (2分) 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。（ ）

30. (2分) sum[i][j]的值(1<j<i≤1000)不小于sum[i-1][j-1]的值。（ ）

31. (2分) 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。（ ）

单选题

32. (4分) A[i][j]（1≤i≤10，1≤j≤10)的所有元素中，最大值是（ ）。

A. 126　　B. 276　　C. 252　　D. 210

33. (3分) 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（程序输入不超过数组或字符串定义的范围；共 **10** 小题，共计 **30** 分，各小题分值见题目）

(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。 输出一行，一个整数表示答案。

```
01 #include<cstdio>
02 using namespace std;
03 #define MAXN 300005
04 int n, ans=0, a[MAXN], in[MAXN]={0};
05 bool vis[MAXN]={0};
06 void dfs(int cur, int w){
07     if(vis[cur])
08         return;
09     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 }
```

34. (3分) (1)处应填

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

35. (3分) (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

36. (3分) (3)处应填

A. 0　B. 1　C. w　D. 1-w

37. (3分) (4)处应填

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

38. (3分) (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≤N≤10000)个题目，第i题对应的得分是a。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<K<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 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。

```
01 #include<cstdio>
02 #include<cmath>
03 #define min(a,b)(a<b?a:b)
04 #define MAXN 100002
05 using namespace std;
06 int n, k[MAXN], cnt=0;
07 int s[MAXN], minScore, sum;
08 double maxAverage=0,nowAverage;
09 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){
```

```
            __(4)__
                maxAverage=nowAverage;
        } else if(fabs(nowAverage-maxAverage)<1e-6)
            __(5)__;
    }
    for(int i=cnt; i >= 1; i--)
        printf("%d\n",k[i]);
    return 0;
}
```

39. (3分) (1)处应填

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

40. (3分) (2)处应填

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

41. (3分) (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)

42. (3分) (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;

43. (3分) (5)处应填

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