全国入门组 CSP-J 初赛模拟试题（5）

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．文件型病毒传染的主要对象是（　）

A.文本文件　B.系统文件　C.可执行文件　D.EXE 和.COM 文件

2．24 针打印机的分辨率约为 180dpi. Dpi 数越大，打印精度越高。其中单位 dpi 是指(　)。

A.印点/厘米　B.印点/毫米　C.印点/英寸　D.印点寸

3．内存地址的最重要特点是（　）。

A.随机性　B.唯一性　C.顺序性　D.连续性

4．多媒体计算机是指（　）

A.具有多种功能的计算机　B.具有多种外设的计算机

C.能处理多种媒体的计算机　D.能借助多种媒体操作的计算机

5．最早的计算机的用途是用于（　）。

A.科学计算　B.自动控制　C.系统仿真　D.辅助设计

6．CPU 中（　）机构相当于运算器中的一个存储单元，它的存取速度比存储器要快得多。

A.存放器　B.辅存　C.主存　D.寄存器

7．计算机软件我们一般指的是（　）。

A.系统软件和实用软件　B.实用软件和自由软件

C.培训软件和管理软　D.编辑软件和科学计算软件

8．操作系统在第几代计算机开始应用（　）。

A.第一代　B.第二代　C.第三代　D.第四代

9.计算机中的数有浮点与定点两种，其中用浮点表示的数，通常由(　)这两部分组成(　)。

A.指数与基数　B.尾数与小数　C.阶码与尾数　D.整数与小数

10．如果用一个字节来表示整数，最高位用作符号位，其他位表示数值。例如：0000001 表示+1，1000001 表示-1，试问这样表示法的整数 A 的范围应该是（　）。

A.-127<=A<=127　B.-128<=A<-128　C.-128<=A<128　D.-128

11．下列叙述中，正确的是（　）。

A 线性表的线性存储结构优于链表存储结构

B.队列的操作方式是先进后出

C.栈的操作方式是先进先出

D.二维数组是指它的每个数据元素为一个线性表的线性表

12．用某种排序方法对线性表 25 , 84,21,47,15,27,68,35,20 进行排序，结点变化如下:

(1)25,84, 21,47,15, 27, 68,35,28;

(2)20,15, 21,25,47, 27, 68,35,84;

(3)15,20, 21,25,35, 27, 47,68,84;

(4)15,20, 21,25,27, 35, 47,68,84；那么，排序方法是（　）。

A.选择排序　B.希尔排序　C.合并排序　D.快速排序

13．如果某二叉树的前序为 STUWV， 中序为 UWTVS，那么该二叉树的后序是（　）。

A.WUVTS　B.UWVTS　C.VWUTS　D.WUTSV

14．下面关于数据结构的叙述中，正确的叙述是（　）。

A.顺序存储方式的优点是存储密度大，且插入、删除运算效率高

B.链表中的每一个结点都包含一个指针

C.包含 n 个结点的二叉排序树的最大检索长度为 log -2n

D.将一棵树转换为二叉树后，根结点没有右子树

15．表达式(1+34)*5-56/7 的后缀表达式为（　）。

A.1 34+5 56 7-*/ B.-*+1 34 5/56 7 C.1 34 +5*56 7/- D.1 34 5* +56 7/

二、阅读程序（程序输入不超过数组或字符串定义的范围；除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分）

1、

```
1    #include<iostream>
2    using namespace std;
3    void hanoi(int n,char a, char b,char c) {
4        if(n==1)
5            cout<<n<<" "<<a<<" "<<c<<endl;
6        else {
7            hanoi(n-1,a,c,b);
8            cout<<n<<" "<<a<<" "<<c<<endl;
9            hanoi(n-1,b,a,c);
10       }
11   }
12   int main() {
13       int n;
14       cin>>n;
15       hanoi(n,'A','B','C');
16       return 0;
17   }
```

判断题

1) (1 分)当 n≥0 时，程序不会出现死循环。( )

2) (1 分)输出共有 2^n 行。( )

3) 当 n>0 时，将第 4 行的“==” 改为“<=”，程序输出结果必定不变。( )

4) 将第 5 行的“n”改为“1”，程序输出结果必定不变。( )

选择题

5) (3 分)此程序的时间复杂度是（ ）。

A.O(n) B.O(n^2) C.O(n^3) D.O(2^n)

6) 若要求输出不超过 15 行，则下列哪个 n 的值是合法的（ ）。

A.0 B. 4 C. 5 D.6

2、

```
1    #include <cstdio>
2    #define N 1005
3
4    using namespace std;
5    int num[N];
6
7    int main() {
8        int a1=1,n,x;
9        scanf("%d", &n);
10       num[1] = 1;
```

```
11          for(int i=1; i<=n; ++i) {
12              x=0;
13              for(int j=1; j<=a1; ++j) {
14                  num[j] = num[j] * 5 + x;
15                  x = num[j] / 10;
16                  num[j] %= 10;
17              }
18              if (x > 0) num[++a1] = x;
19          }
20          printf("0.");
21          for(int i=a1; i<n; ++i) {
22              putchar('0');
23          }
24          for(int i=a1; i>=1; i--) {
25              printf("%d", num[i]);
26          }
27          putchar('\n');
28          return 0;
29      }
```

判断题

1) (1 分)程序输出的是 5^n 的值。(　　)

2) (1 分)程序执行到第 27 行时，i 的值为 1。(　　)

3) 对于任意 1≤i≤al，都有 0≤num≤9。(　　)

4) 程序输出的是一个小数，且小数末尾可能有多余的 0。(　　)

选择题

5) 此程序的时间复杂度是（　　）。

A.O(n)　　B.O(n^2)　　C.O(n^3)　　D.O(nlogn)

6) 若 n=3，则输出为（　　）。

A.8　　B.0.125　　C.0.8　　D.125

3、在起点和终点之间，有 N 块岩石(不含起点和终点的岩石)。在比赛过程中，选手们将从起点出发，每一步跳向相邻的岩石，直至到达终点。

为了提高比赛难度，组委会计划移走一些岩石， 使得选手们在比赛过程中的最短跳跃距离尽可能长。由于预算限制，组委会至多从起点和终点之间移走 M 块岩石(不能移走起点和终点的岩石)。

输入文件第一行包含三个正整数 L, N, M，分别表示起点到终点的距离，起点和终点之间的岩石数，以及组委会至多移走的岩石数。

接下来 N 行，每行一个整数，第 i 行的整数 a[i] (0 < a[i] < L)表示第 i 块岩石与起点的距离。这些岩石按与起点距离从小到大的顺序给出，且不会有两个岩石出现在同一个位置。

输出文件只包含一个整数，即最短跳跃距离的最大值。

```
1   #include <iostream>
2   using namespace std;
3   int l,n,m,a[50005], ans;
4   bool check(int dis)
```

```
5    {
6        int count=0,last=0;
7        for(int i=1; i<=n; i++)
8            if(a[i]-last<dis)count++;
9            else last=a[i];
10       if(count>m)return 0;return 1;
11   }
12   int main()
13   {
14       ios::sync_with_stdio(0);
15       cin>>l>>n>>m;
16       for(int i=1; i<=n; i++)
17           cin>>a[i];
18       a[n+1]=l;
19       int fl=0,fr=l;
20       while(fl<=fr)
21       {
22           int mid=(fl+fr)/2;
23           if(check(mid))fl=mid+1, ans=mid;
24           else fr=mid-1;
25       }
26       cout<<ans ;
27       return 0;
28   }
```

判断题

1) (1 分)将第 19 行的“fl=0”改为“f1=l”，程序输出结果必定不变。(　　)

2) (2 分)程序执行到第 26 行时，必有 fl>fr。 (　　)

3) (2 分)若第 23 行执行的 check(mid)==l, 则最终的 ans≤此时的 mid。(　　)

4) (2 分)程序执行到第 10 行时，count 的值表示:如果最短跳跃距离恰好为 dis，那么最少需要移走几块岩石。(　　)

选择题

5) 此程序的时间复杂度是（　　）。

A.O(n^2)　　B.O(nl)　　C.O(n log l)　　D.O(nlogn)

6) 若输入为:25 5 2 2 11 14 17 21 则输出为（　　）。

A.3　　B. 4　　C.5　　D.6

三、完善程序（单选题，每题 3 分，共计 30 分）

1、迪杰斯特拉算法是由荷兰计算机科学家狄克斯特拉于 1959 年提出的，因此又叫狄克斯特拉算法。是从一个顶点到其余各顶点的最短路径算法，解决的是有向图中最短路径问题。迪杰斯特拉算法主要特点是以起始点为中心向外层层扩展，直到扩展到终点为止。

```
#include<iostream>
using namespace std;
int main() {
    int edgs;
```

```
int points ;
int dis[10];
int flag[10];
int infinity=999999;
cin>>points>>edgs ;
int edg[10][10];
for (int i=1; i<=points; i++) { // 初始化
    for (int j=1; j<=points; j++) {
        if (i==j) {
            edg[i][j]=___(1)___;
        } else {
            edg[i][j]=___(2)___;
        }
    }
}
int point1 , point2 , quanzhi ;
for (i=1; i<=edgs; i++) {
    cin>>point1>>point2>>quanzhi ;
    edg[point1][point2]= ___(3)___ ;
}
for (i=1; i<=points; i++) {
    dis[i]=edg[1][i];
}
for (i=1; i<=points; i++) {
    flag[i]=0;
}
flag[1]=1;
int min,u;
for (i=1; i<=points-1; i++) {
    //源点到源点不用比较，因次总的次数少一次
    min=infinity;
    for (int j=1; j<=points; j++) {
        if (f1ag[j]==0&&dis[j]<min) {
            //核心思想:依次比较出离源点最近的点
            min=___(4)___;
            u=j;
        }
    }
    flag[u]=1;
    for (int v=1; v<=points; v++) {
        //找出离源点最近的点后更新 dis 里面的源点到各个点的值是否最小
        if (edg[u][v]<infinity) {
            if (dis[v]>dis[u]+edg[u][v]) {
                dis[v]=___(5)___;
```

```
                    }
                }
            }
        }
        for (i=1; i<=points; i++) {
            cout<<dis[i]<<" ";
        }
        cout<<endl;
}
```

选择题

1）①处应填（）

A.infinity　　B.dis[j]　　C.0　　D.1

2）②处应填（）

A.infinity　　B.dis[j]　　C.0　　D.1

3）③处应填（）

A.quanzhi　　B.0　　C.inf　　D.1

4）④处应填（）

A.j　　B.dis[j]　　C.flag[j]　　D.i

5）⑤处应填（）

A.dis[u]　　B.edg[u][v]　　C.dis[u]+edg[u][v]　　D.infinity

2、完全背包问题

容量为 10 的背包，有 5 种物品，每种物品数量无限，其重量分别为 5, 4, 3, 2，1，其价值分别为 1，2，3，4，5。

设计算法，实现背包内物品价值最大。代码如下(输出 50)

```
#include<iostream>
#include<algorithm>
using namespace std;
int main() {
    int total_weight = 10;
    int w[6] = { 0,5,4,3,2,1};
    int v[6] = { 0,1,2,3,4,5};
    int dp[11]= { ___(1)___ };

    for(int i=1; i<=___(2)___; i++)
        for(int j=w[i]; j<=___(3)___; j++)
            dp[j] =___(4)___;
    cout <<___(5)___<< endl;
    return 0;
}
```

选择题

1) ①处应填( )

A.0　　B.5　　C.10　　D.15

2) ②处应填( )

A.5　　B.6　　C.10　　D.15

3) ③处应填( )

A.5　　B.6　　C.10　　D.15

4) ④处应填( )

A.dp[j]+v[i]　B.dp[j - w[i]] + v[i]　C.min(dp[j],dp[j-w[i]]+v[i])　D.max(dp[j], dp[j-w[i]]+v[i])

5) ⑤处应填( )

A.v[10]　　B.dp[10]　　C.w[10]　　D.total_weight