算法设计与分析实验

1、共有两个单元 4 道题，每个单元至少完成 1 道题。

2、编程语言：任选

3、5 月 23 日之前交报告。

4、报告内容：

（1）源程序

（2）测试截图（对结果附一定的文字说明）

（3）其它需要说明的文字

5、纪律要求：**独立完成，不得抄袭**。

实验一 最大子数组问题/平面上的最近点对问题

1、分别实现 4.1 节所讲的求最大子数组问题的分治算法 FIND-MAXIMUM-SUBARRAY 和习题 4.1-5 所提示的非递归线性时间算法。

要求：

（1） 分别实现两个算法的功能函数；

（2）将（1）中实现的函数，集成在一个程序中，通过菜单选择功能编号的方式调用两个算法：

1：求最大子数组问题的分治算法

2：求最大子数组问题的线性时间算法

（3）测试数据集从文件读入。每个文件一个数据集，文件名任意。数据集文件是文本文件，格式如下：

文件中包含 1+n 个整数：第一整数是 n 的值，表示后面又 n 个整数；其后跟 n 个整数。所有整数用空格分隔。

含义：读入第一个整数，确定测试数据集的大小，然后开辟程序的数据空间，读入这 n 个整数，并求这 n 个整数的最大子数组。

0≤n≤1000000。算法必须具有良好的空间效率并能处理 n 的各种取值情况。

（4）输出

找出这 n 个整数的最大子数组，输出一行结果：先输出子数组的和值，然后输出子数组在整个数据集下标区间[0:n-1]中的起始下标和终止下标，最后输出该子数组包含的 k 个整数。

输出时所有整数用空格隔开。如果输出长度超出一行，自然换行即可，不用加换行符

如果 n=0，则输出 0。

（5）对于（3）和（4），如果一个数据集中包含多个和值相同的最大子数组，先实现找出其中之一即返回结果的程序，输出该结果。

（6）在（5）的基础上，改造程序，对数据集包含多个和值相同的最大子数组的情况，能找出所有这样的子数组并输出结果（每个结果一行）。

（7）自行构造数据集定量对比两个算法计算时间上的差异。

2、编程实现求平面上的最近点对问题的程序

要求：

（1）编程实现求最近点对问题的程序；

（2）测试数据集从文件读入。数据集文件是文本文件，文件名任意，格式如下：

文件包含 1+2n 个整数：第一整数是 n 的值，表示后面有 n 个点的坐标；其后跟 2n 个整数，两个整数一组，表示一个点的 x 坐标和 y 坐标。所有整数用空格分隔。

含义：读入第一个整数，确定测试数据集的大小，然后开辟程序的数据空间，读入这 n 个点的坐标，并求这 n 个点之间最近的点对。

0≤n≤10000。算法必须具有良好的空间效率并能处理 n 的各种取值情况，包括数据不完整的情况。

（3）输出

找出这 n 点中的最近点对，然后输出一行结果：先输出这两个点的坐标（四个整数），再输出两者之间的距离（浮点数）。输出的所有数据之间用空格隔开，处于一行。

（4）对以上（2）和（3），若存在几个相同距离的最近点对，先实现找出其中之一即返回结果的程序，输出结果。

（5）在（4）的基础上，改造程序，对 n 个结点之间存在几个距离相同的最近点对的情况，找出所有这样的点对并输出结果（每个点对输出一行）。

（6）n=0 时输出：没有结点；n=1 时输出：至少需要两个结点。坐标不配对时，输出：有异常坐标。

按照各题的测试数据文件，根据格式要求自行构造。

以上两题至少完成一题。

实验二 最短路径算法/N 皇后问题

1、编程实现求单源点最短路径的 Bellman-Ford 算法和 Dijkstra 算法，以及求每对结点之间最短路径的 Floyd-Warshall 算法和 Johnson 算法。

要求：

（1）分别实现 Bellman-Ford 算法、Dijkstra 算法、Floyd-Warshall 算法及 Johnson 算法的功能函数；

（2）将（1）中实现的函数，集成在一个程序中，通过菜单选择功能编号的方式调用这些算法：

1：Bellman-Ford 算法

2：Dijkstra 算法

3：Floyd-Warshall 算法

4：Johnson 算法

（3）测试数据集从文件读入。每个文件一个数据集，文件名任意。数据集文件是文本文件，格式如下：

文件中第一行是 2 个整数:第一整数是有向图中的结点数 n,图中结点用编号 1~n 代表；第二个整数是询问的问题数 m。

第二行起的 n 行，每行 n 个整数，这 n×n 个数对应图的成本邻接矩阵 A，A[i,j]是边（i，j）的权值，A[i,j]=10000 表示结点 i 到结点 j 没有边。

第 n+2 行起的 m 行是询问。分单源点测试和每对结点之间的最短路径测试。对单源点测试，每行一个整数 s,是图中一个结点的编号，表示求以 s 为源点的单源点最短路径。对每对结点之间的最短路径测试，每行两个整数（i，j），表示询问从结点 i 到结点 j 的最短路径是什么。

程序运行时，打开测试数据文件，首先读入第一行的 2 个整数，确定图的成本邻接矩阵的大小，然后读入图的成本邻接矩阵以及询问的问题。再后，根据（2）选择的算法进行最短路径计算。

0≤n≤100。算法必须具有良好的空间效率并能处理 n 的各种取值情况。

（4）输出

（a）对于单源点最短路径算法

读入图后，根据询问（指定源点），计算指定源点到其它各个结点的最短路径。输出时，先输出成本邻接矩阵,然后按照路径从短到长的顺序依次输出源点到其它结点的各条最短路径。

路径的输出形式是：先输出路径的权值，然后输出该路径的结点序列，一条路径输出一行，每行中的数之间用空格隔开。

如果执行的是 Dijkstra 算法，但发现图中含有负权值的边，则输出“图中含有负权值

的边，算法终止”；

如果执行的是 Bellman-Ford 算法，但计算后发现图中含有负权值的环路，则输出“图中含有负权值的环路，无法计算最短路径”；

（b）对于每对结点的最短路径算法

读入图后，先整体计算图中每对结点之间的最短路径，得最短路径权重矩阵 D 和前驱结点矩阵Π。然后输出图的邻接成本矩阵 A、计算所得的最短路径权重矩阵 D 和前驱结点矩阵Π。

再之后，依次输出每个询问的结果，即：询问 i 到 j 的最短路径时，根据 D 查询 i 到 j 的最短路径权重，根据Π求 i 到 j 最短路径的结点序列，然后输出结果。询问结果的输出形式是：先输出指定的两点的编号，然后输出两点间最短路径的权重，再接着输出路径上从起点到终点的阶点序列。每个询问输出一行，如果一行内容太长，显示不下，自然换行即可，程序不用控制换行。以上输出的所有数之间用空格隔开。

如果图中含有负权值的环路，则计算后输出“图中含有负权值的环路，无法计算最短路径”；

以下是每对结点之间最短路径的一个测试数据样例（设输入文件名为 in.dat，将以下红色文字复制到一个文本文件里，保存为 in.dat 即可）。

```
5 2
0 3 8 10000 -4
10000 0 10000 1 7
10000 4 0 10000 10000
2 10000 -5 0 10000
10000 10000 10000 6 0
1 2
2 5
```

设运行 Floyd-Warshall 算法。则输出为：

```
图的成本邻接矩阵：
0 3 8 10000 -4
10000 0 10000 1 7
10000 4 0 10000 10000
2 10000 -5 0 10000
10000 10000 10000 6 0
图的最短路径权重矩阵：
0 1 -3 2 -4
3 0 -4 1 -1
7 4 0 5 3
2 -1 -5 0 -2
```

8 5 1 6 0
图的前驱结点矩阵：
-1 3 4 5 1
4 -1 4 2 1
4 3 -1 2 1
4 3 4 -1 1
4 3 4 5 -1
询问结果：
1 2 1 1 5 4 3 2
2 5 -1 2 4 1 5

其它测试数据自行构造。

2、N 皇后问题

要求

（1）编程实现求解 N 皇后问题的程序。

（2）测试方式：

运行程序，首先输入代表皇后数的 n，4≤n≤64；然后输出该 n 皇后问题的所有解。

每个解以 n×n 的阵列形式输出，阵列元素[i,j]=1 表示在 i 行 j 列放置了一个皇后，[i,j]=0 表示在 i 行 j 列没有放置任何皇后。阵列中的 0 和 1 之间以空格隔开，排列整齐，以方便于看出棋盘上皇后的布局。如果有多个解，两个解之间加一空行。

如：

输入 8

输出：

0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0

0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0

思考题：8 皇后问题是一道经典的排列题。虽然上面给出 4≤n≤64，但随着 n 的增大，算法的执行时间会越来越长。你可以测试一下自己的程序能够有效执行到 n 等于多少（计时看一下各种 n 情况下计算时间的长短）。