爱码柿编程

基础练习（1）

- 授课人：樊高雁

目录
CONTENTS

01 Promotion Counting

01 Promotion Counting

母牛Bessie在帮助农民John维护USA Cow Olympiad（USACO），这是一个在线竞赛平台，选手回答具有挑战性的问题，以展示他们对牛的了解。

为了区分不同的选手水平，John最近将竞赛分成了4个不同的难度赛区：青铜，银，金，白金。所有新参赛者均从青铜赛区开始，每当他们在比赛中表现优异，他们就会晋升到下一个更高级别的赛区。选手甚至可以在一次比赛中多次晋升。John记录了所有比赛参赛者及其当前分区的列表，这样他就能在比赛中管理选手的晋升情况。

在统计最近一次比赛的分区情况时，John想提供从青铜到银，从银到金，从金到白金的晋升人数的信息。然而在比赛中他忘了统计。作为一个聪明的牛，Bessie意识到John可以仅凭赛前赛后的各分区选手人数推断出晋升情况。请帮她完成这个推断。

输入格式 (file promote.in):

输入包括4层，每一层包含2个范围在0...1000000的整型数，第一行数字分别表示赛前赛后青铜赛区选手的数量。第二行数字表示赛前赛后银赛区选手的数量，第三，第四行则分别为金，白金赛区赛前赛后选手的数量。

输出格式(file promote.out):

请输出3行，每行包含一个整型数，第一行应该包含从青铜赛区晋升到银赛区的选手数量，第二行包含从银赛区晋升到金赛区的选手数量，最后一行包含从金赛区晋升到白金赛区的选手数量。

01 Promotion Counting

输入样例:

1 2

1 1

1 1

1 2

输出样例:

1

1

1

说明：

在这个例子中，赛前每个赛区均有一个选手。在赛后，青铜和白金赛区的选手数量变为2人。可能发生的一种方式是在比赛期间加入了两名新选手，一个选手一直晋升到铂金，另一个则一直在青铜赛区。

02 Angry Cows

02 Angry Cows

母牛Bessie设计了一款名为“Angry Cows”的电子游戏，玩家拉动弹弓将母牛射入一个一维场景，该场景由一组位于数轴上不同点的干草捆组成，母牛落在干草捆上，会使干草捆爆炸，同时会产生连锁反应，使附近的干草捆爆炸。游戏目的是发射一头母牛，引爆尽可能多的干草捆。

假设数轴上有N捆干草捆，坐标分别为x_1,x_2,...x_N，如果一头母牛落到位于x位置的干草捆上，则这捆干草捆会产生1爆炸半径的爆炸。这意味着1个单位距离内的任何其他干草捆也会爆炸，经过一个单位时间，这些相邻的干草捆自身爆炸（同时爆炸），每个爆炸半径为2，这些爆炸又可能会引爆在2个爆炸半径内的另外尚未爆炸的干草捆，在下一个单位时间，这些干草捆爆炸（同时爆炸），产生3个单位距离的爆炸。即在t单位时间时爆炸的干草捆，其爆炸半径为t。而被这些爆炸引爆的干草捆会在t+1单位时间时爆炸，爆炸半径为t+1，以此类推。

请决定发射母牛的目的位置，使爆炸的干草捆数量最多，并求出爆炸的最大数量。

输入格式(file angry.in):

输入的第一行包含干草捆的数量N，剩下N行包含每个干草捆在数轴上的位置，分别为x_1,x_2...x_N（每个均为在0...1000000000的整型数）

输出格式(file angry.out):

请输出发射一头母牛能造成干草捆爆炸的最大数量。

02 Angry Cows

输入样例:

6

8

5

6

13

3

4

输出样例:

5

说明：

在这个例子中，将母牛发射到数轴上5位置可以导致4，6上的干草捆的爆炸，爆炸半径为2，继而导致3，8上的干草捆爆炸，爆炸半径为3，但半径内已无干草捆，不能再进一步导致爆炸。

爱码桔编程

03 Mowing the Field

03 Mowing the Field

农民John在管理农场方面十分可靠，除了一点：他不能及时且美观地修剪他的草坪。

农场是由一个个方形单元格组成的大型2D网格。在时间t=0时，John从其中的一个单元格开始，在该单元格内割草。使其成为初始唯一一个被修剪过的单元格。之后John的割草过程可由N个语句描述。例如，如果第一个语句是“W 10“，那么对于时间t=1到t=10（即接下来的10个单位时间），John会从初始单元格向西移动，并修剪一路上经过的单元格。在t=10时，他能够完成初始单元格西边的10个单元格的修剪。

由于John的割草进度实在太慢，以至于他在割完整个2D网格前，其中一些割过的单元格的草已经重新长出来。在时间t割的草将在时间t+x时重新出现。

John的修剪过程可能会让他多次经过同一个单元格，但他说他从未遇到过已经完成修剪的单元格。也就是说，每次John经过一个已经修剪过的单元格，必定是在上次完成修剪的x个单位时间后，以使草能重新长出来。

请确定x的最大可能值，以使John的话成立。

03 Mowing the Field

输入格式(file mowing.in):

第一行输入包含整数N(1≤N≤100)，剩下的N行，每行输入代表一次John的行动，格式为‘D S’其中D为描述方向的字符（N=北，E=东，S=南，W=西），S是向该方向前进的步数(1≤S≤10)。

输出格式(file mowing.out):

请决定x的最大值，使得John从未经过一个已修剪完毕的单元格，如果John本来就从未访问过一个单元格2次，请输出-1。

输入样例:

```
6
N 10
E 2
S 3
W 4
S 5
E 8
```

输出样例:

```
10
```

说明：

在这个例子中，John在第17个单位时间时经过他已在第7个单位时间修剪过的单元格，因此，x必须小于10，否则单元格中的草还未长出。在第26个单位时间他经过了在第2个单位时间时他修剪过的单元格，因此x必须也小于24，比较两个约束，我们可以得出x最大可以取10。

04 Milking Order

04 MILKING ORDER

Farmer John的N头奶牛（2≤N≤100），方便起见仍然编号为1...N。奶牛们正好闲得发慌。因此，她们发展了一个与Farmer John每天早上为她们挤牛奶的时候的排队顺序相关的复杂的社会结构。经过若干周的研究，Farmer John发现这个结构基于两个关键特性。

首先，由于奶牛们的社会阶层，某些奶牛会基于她们的社会地位等级，坚持要在其他奶牛之前挤奶。比方说，如果奶牛3有最高的地位，奶牛2位于中等地位，奶牛5是低地位，那么奶牛3会最早挤奶，然后是奶牛2，最后是奶牛5。

然后，有些奶牛只允许她们在排队顺序中一个特定的位置挤奶。比方说，奶牛4可能坚持要在所有奶牛中的第二位挤奶。

幸运的是，Farmer John总是能够以一种满足所有这些情况的顺序给他的奶牛们挤奶。

不幸的是，奶牛1最近生病了，所以Farmer John想要尽早给这头奶牛挤奶，使得她可以回到牛棚获得急需的休息。请帮助Farmer John求出奶牛1可以在挤奶顺序中出现的最早位置。

MILKING ORDER

输入格式:

第一行包含N，M(1≤M<N)，和K (1≤K<N)，表示Farmer John有N头奶牛，其中M头形成了社会阶层，K头需要在挤奶顺序中处于一个特定的位置。下一行包含M个不同的整数m_i($1 \le m_i \le N$)。在这一行出现的奶牛必须以与她们在这行出现的顺序相同的顺序进行挤奶。下面K行，每行包含两个整数ci($1 \le c_i \le N$)和qi($1 \le q_i \le N$)，表示奶牛c_i一定要在第p_i位进行挤奶。

输入数据保证在这些限制之下，Farmer John能够建立一个符合要求的挤奶顺序。

输出格式:

输出奶牛1可以在挤奶顺序中出现的最早位置。

输入样例:

```
6 3 2
4 5 6
5 3
3 1
```

输出样例:

```
4
```