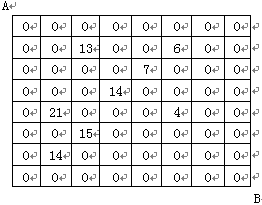
**方格取数**

(pane.cpp)

**描述**

设有N×N的方格图，我们在其中的某些方格中填入正整数，而其它的方格中则放入数字0。如下图所示：



某人从图中的左上角的A出发，可以向下行走，也可以向右行走，直到达右下角的B点。在走过的路上，他可以取走方格中的数(取走后的方格中将变为数字0)。

此人从A点到B点共走了两次，试找出两条这样的路径，使得取得的数字和为最大

**输入(pane.in)**

输入文件Pane.in第一行为一个整数N（N≤10），表示N×N的方格图。

接下来的每行有三个整数，第一个为行号数，第二个为列号数，第三个为在该行、该列上所放的数。一行0 0 0表示结束。

**输出(pane.out)**

输出文件pane.out包含一个整数，表示两条路径上取得的最大的和。

**样例输入**

8

2 3 13

2 6 6

3 5 7

4 4 14

5 2 21

5 6 4

6 3 15

7 2 14

0 0 0

**样例输出**

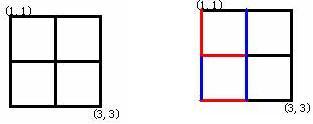
67

格子游戏

（game.cpp）

**描述**

Alice和Bob玩了一个古老的游戏：首先画一个n\*n的点阵（下图n=3），接着，他们两个轮流在相邻的点之间画上蓝边和红边：



直到围成一个封闭的圈（面积不必为1）为止，画“封圈”那一笔的人就是赢家。因为棋盘实在是太大了(n <= 200)，他们的游戏实在是太长了！他们甚至在游戏中都不知道谁赢得了游戏。于是请你写一个程序，帮助他们计算他们是否结束了游戏？

**输入(game.in)**

输入数据第一行为两个整数n和m。m表示一共画了m条线。以后m行，每行首先有两个数字(x, y)，代表了画线的起点坐标，接着用空格隔开一个字符，假如字符是"D "，则是向下连一条边，如果是"R "就是向右连一条边。输入数据不会有重复的边且保证正确。

**输出(game.out)**

输出一行：在第几步的时候结束。假如m步之后也没有结束，则输出一行“draw”。

**样例输入**

3 5

1 1 D

1 1 R

1 2 D

2 1 R

2 2 D

**样例输出**

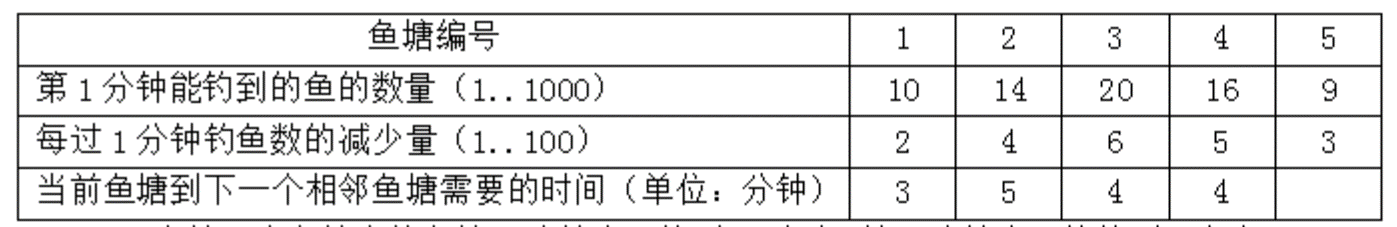
4

鱼塘钓鱼

（fishing.cpp）

**问题描述**

有N个鱼塘排成一排（N<100），每个鱼塘中有一定数量的鱼，例如：N=5时，如下表：



即：在第1个鱼塘中钓鱼第1分钟内可钓到10条鱼，第2分钟内只能钓到8条鱼，……，第5分钟以后再也钓不到鱼了。从第1个鱼塘到第2个鱼塘需要3分钟，从第2个鱼塘到第3个鱼塘需要5分钟，……

给出一个截止时间T(T<1000)，设计一个钓鱼方案，从第1个鱼塘出发，希望能钓到最多的鱼。

假设能钓到鱼的数量仅和已钓鱼的次数有关，且每次钓鱼的时间都是整数分钟

**输入格式(**fishing.in**)**

共5行，分别表示：

第1行为N；

第2行为第1分钟各个鱼塘能钓到的鱼的数量，每个数据之间用一空格隔开；

第3行为每过1分钟各个鱼塘钓鱼数的减少量，每个数据之间用一空格隔开；

第4行为当前鱼塘到下一个相邻鱼塘需要的时间；

第5行为截止时间T；

**输出格式(**fishing.out**)**

输出文件仅一个整数（不超过2^31-1），表示你的方案能钓到的最多的鱼。

**样例输入**

5

10 14 20 16 9

2 4 6 5 3

3 5 4 4

14

**样例输出**

76