South African Computer Olympiad
Training Camp 1, 2005
Day 1

Overview

Author	**Bruce Merry (from CCC 1996)**	**Indiana Jones**	**Carl Hultquist**
Problem	**pieswap**	**pyramid**	**walls**
Source	pieswap.pas pieswap.c pieswap.cpp pieswap.java	pyramid.pas pyramid.c pyramid.cpp pyramid.java	walls.pas walls.c walls.cpp walls.java
Input file	pieswap.in	pyramid.in	walls.in
Output file	pieswap.out	pyramid.out	walls.out
Time limit	1 second	1 second	5 seconds
Number of tests	10	10	10
Points per test	10	10	10
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

Pie Swapping

Author

Bruce Merry (from CCC 1996)

Introduction

Fred the manic store-keeper wants to rearrange the pies in his shop window. His display has N pies in a single row. It also has a rather awkward design, which means that he can only rearrange pies by repeatly swapping two pies that are next to each other in the row.

Task

Given the current order of the pies, help Fred find the minimum number of swaps that he needs to make to put them in the order that he wants.

Example

Suppose there are four pies. Let them be numbered 1–4 in the order that Fred wants them. The current order is 2 4 1 3. He can arrange the pies using the following sequence:

```
2 1 4 3
1 2 4 3
1 2 3 4
```

This takes three swaps.

Input (pieswap.in)

The first line of input contains N, the number of pies. The following N lines each contains one integer, listing the current order of the pies from left to right. Each integer from 1 to N will appear exactly once in these lines.

Sample input

```
4
2
4
1
3
```

Output (pieswap.out)

The first line of output must contain S, the minimum number of swaps required. The following S lines each contains a single integer, describing a swap to perform. The pie indicated by the integer is to be swapped with the pie on its right (note: the integer indicates a pie, not a position). After applying these swaps, the pies must be in order from 1 to N from left to right.

Sample output

```
3
4
4
2
```

Constraints

$1 \leq N \leq 200$.

50% constraints

$1 \leq N \leq 10$.

Time limit

1 second.

Scoring

You will score 0%

- if the output is formatted incorrectly;
- if a pie indicated in the output is already the rightmost pie;
- if more than double the minimum number of swaps is used;
- if the output swaps fail to correctly order the pies.

Otherwise you will score 100% for an optimal solution, or 20% for a sub-optimal solution.

Egyptian Pyramid

Author

Indiana Jones

Introduction

Indiana Jones has found a previously undiscovered pyramid in Egypt, buried in sand. It has a very unique design. It can only be entered at the top and left at one of a number of exits at the bottom. For every chamber of the pyramid there are two exits: one goes down and to the left, the other down and to the right. Some cunning trapdoor mechanisms make it impossible to go upwards.

```
            X
          /   \
        X       X
      /   \   /   \
    X       X       X
  /   \   /   \   /   \
X       X       X       X
```

The resulting network looks like the above, with X's representing chambers and slashes representing passages.

Fortunately Indiana knows (from some ancient documents) exactly what is in each chamber and how much it is worth. Unfortunately the evil Nazi's are approaching and he will only have time to move through the pyramid once (going from top to bottom).

Task

Your task is to help Indiana to determine the route that will get him the greatest value of treasure.

Example

Suppose the pyramid looks like this (where the values indicate the treasure in each chamber):

```
      5
     / \
    4   2
   / \ / \
  2   5   1
 / \ / \ / \
1   3   2   4
```

Then the most treasure that Indiana can get would be if he collected the treasures with values 5, 4, 5 and 3, in that order, for a total of 17.

Input (pyramid.in)

The first line of input will contain a single integer, N, which is the number of rows in the pyramid (and also the number of exits). The next N lines each describe one row of the pyramid, from top to bottom. Line $i+1$ of the input will contain i space-separated integers, which are the values of the chambers in row i, from left to right.

Sample input

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

Output (pyramid.out)

You must output only a single integer, the maximum value of the treasures Indiana can collect.

Sample output

```
17
```

Constraints

- $1 \leq N \leq 100$
- $1 \leq$ any treasure's value ≤ 1000

50% constraints

- $1 \leq N \leq 20$

Time limit

1 second.

Scoring

Your program will score 100% for a test-case if you output the correct answer. Otherwise, your program will score 0% for that test-case.

Blow-torching Walls

Author

Carl Hultquist

Introduction

After Farmer John's cows tried to escape the farm recently, Farmer John decided that he has had enough: these cows must be sent to the abbatoir! The cows, who have been spying on Farmer John, have discovered this. . . and are very worried. They **really** need to escape now, otherwise they'll end up as roast beef on Farmer John's dinner table!

There is only one chance of escaping: and that is through the underground maze that runs beneath the farm. The cows raided Farmer John's office during the night and found a map of the maze. The only problem is that Farmer John has blocked up many parts of the maze with concrete wall. . . But luckily for the cows, they have also raided his toolshed and have stolen a blow-torch!

Blow-torching an entire piece of wall takes a **long** time: so the cows need to work out the smallest number of walls that they'll have to blow-torch away in order to escape the maze, and also find the shortest route that needs this number of walls blow-torched away.

Task

Your task is to help the cows plan their escape route. The maze consists of a number of square areas which are either open or which are a wall. The cows can move from any one of these square areas to any of the adjacent square areas (first blow-torching any wall that might be in the way). Moving from one square to another counts as moving a distance of 1 unit. You must:

1. Determine the smallest number of walls that need to be blow-torched for the cows to travel from the given starting point in the maze to the given ending point.
2. Determine the shortest route from the starting point to the ending point that requires Z walls to be blow-torched, where Z is the number determined in part 1. There may be more than one such route: you only need to determine one of them.

Example

Suppose the maze looks like the diagram below, where a '.' indicates an open part of the maze, and a 'W' indicates a wall, and that the cows need to get from the top-left hand corner to the bottom-right hand corner.

```
.W...
...WW
.W.W.
.WWW.
.WW..
```

The smallest number of walls that the cows need to blow-torch is 1, and they can get through the maze most quickly by moving: down, right, right, down, right (blow-torching this wall), right, down and down, which is a total of 8 units of movement.

Input (walls.in)

The first line of input will contain two integers, W and H, which indicate the width and height of the maze respectively. The second line of input will contain two integers, SX and SY, indicating the X and Y co-ordinates respectively of where the cows enter the maze. The third line of input will contain two integers, EX and EY, indicating the X and Y co-ordinates respectively of where the cows must exit the maze. The next H lines will each contain W integers, separated by spaces, that describe the maze. Each of these integers will either be 0 or 1: 0 indicates that there is no wall at that point in the maze, 1 indicates that there **is** a wall.

The upper-left co-ordinate of the maze is $(1, 1)$ and the lower-right co-ordinate is (W, H). This is illustrated below:

```
(1,1)..........
...............
..........(W,H)
```

Sample input

```
5 5
1 1
5 5
0 1 0 0 0
0 0 0 1 1
0 1 0 1 0
0 1 1 1 0
0 1 1 0 0
```

Output (walls.out)

The first line of output must contain two integers, Z and L. Z is the smallest number of walls that need to be blow-torched (determined in part 1 above), and L is the length of the shortest path that requires Z walls to be blow-torched. The next $L+1$ lines should each contain two integers, X and Y, which indicate the co-ordinates of the route that the cows must take.

Sample output

```
1 8
1 1
1 2
2 2
3 2
3 3
4 3
5 3
5 4
5 5
```

Constraints

- $1 \leq W, H \leq 1000$
- The starting location, (SX, SY), will never have a wall on it.

50% constraints

- $1 \leq W, H \leq 250$

Time limit

5 seconds.

Scoring

If your output for a test-case differs in any way from the format prescribed above, then your program will score 0% for that test-case. If your path fails to lead the cows from the starting point to the ending point, then you will score 0%. If the number of walls you need to blow-torch in the path is different from Z, then you will score 0%.

Otherwise, let Q be the optimal number of walls that needed to be blow-torched. Let M be the optimal route length for a route requiring Q walls to be blow-torched. Put $K = \max(M, L)$. Your score is then given by $S = [100 - 30 \times (Z - Q)] \times (\frac{M}{K})^{Z-Q+1}\%$.

Thus if you find the optimal number of walls, and the optimal route for that number of walls, your solution scores 100% :-)