Overview

Author	Timothy Stranex	Migael Strydom	Allen Jansen van Nieuwen-huizen	Keegan Carruthers-Smith
Problem	**bridger**	**taxi**	**mine**	**climber**
Source	bridger.pas bridger.c bridger.cpp	taxi.pas taxi.c taxi.cpp	mine.pas mine.c mine.cpp	climber.pas climber.c climber.cpp
Input file	bridger.in	taxi.in	mine.in	climber.in
Output file	bridger.out	taxi.out	mine.out	climber.out
Time limit	5 seconds	3 seconds	2 seconds	3 seconds
Number of tests	10	10	10	10
Points per test	10	10	10	10
Total points	**100**	**100**	**100**	**100**

The maximum total score is 400 points.

Bridgecrafter Bill

Author

Timothy Stranex

Introduction

After many years studying under Bridgemaster Yukari (9 dan), Bill has learned the ancient art of Bridgecraft. Now, he wants to move to the city to profit from his new skills. Unfortunately, Bill is unknown outside of his village so nobody in the city will hire him. Sad but resolute, he decides to build a bridge for free to demonstrate his ability. Help Bill to find the minimum cost of such a bridge.

Task

There are a number of bridge endpoints throughout the city. Bill can build a bridge between any two of these end-points. The cost of building a bridge is directly proportional to the Manhattan distance[1] between the bridge end-points. This distance is given by,

$$D = |x_2 - x_1| + |y_2 - y_1|,$$

where (x_1, y_1) and (x_2, y_2) are two points. You must find the minimum value of D for a set of N bridge end-points.

Example

Suppose the following bridge end-points are given:

1. $(0, 0)$
2. $(100, 100)$
3. $(20, 5)$
4. $(5, 5)$

The minimum value of D is 10 which occurs when building a bridge between the first and fourth bridge end-points.

Input (bridger.in)

The first line of the input contains a single integer N. The next N lines each contain two space-separated integers X and Y which indicate that there is a bridge end-point at (X, Y).

[1] In a city, you must travel along the horizontal or vertical roads between buildings; you cannot move between two points in a straight line.

Sample input

```
4
0 0
100 100
20 5
5 5
```

Output (bridger.out)

The program must output a line containing only a single integer D, the minimum Manhattan distance between any two of the given points.

Sample output

```
10
```

Constraints

- $1 \leq N \leq 100000 = 10^5$
- $0 \leq X, Y \leq 1000000000 = 10^9$

50% constraints

- $1 \leq N \leq 10000 = 10^4$
- $0 \leq X, Y \leq 1000000000 = 10^9$

Time limit

5 seconds.

Scoring

If your program outputs the correct answer for the test-case, you will score 100% for that test-case. Otherwise, you will score 0% for that test-case.

Taxi

Author

Migael Strydom

Introduction

Eric is a taxi driver. His job is to collect passengers at the airport at the start of a long road and drop each off at the block he or she wishes to go to along the road. The blocks are numbered according to the distance they are from the airport. A passenger would therefore specify a whole number to indicate where she wants to go, so 3 would mean she wants to be taken 3 blocks away from the airport.

Eric has noticed, however, that stopping at every passenger's destination takes a lot of time, and in order to maximise the use of his time he will limit the number of stops he makes. That means some passengers will have to walk, and in order to maintain a good relationship with them, Eric needs to minimise the total number of blocks walked by his customers.

Task

Your task is to write a program that will minimise the total number of blocks walked by Eric's passengers, given the maximum number of stops he is prepared to make. The total number of blocks walked is the sum of the number of blocks walked by each passenger. Note that it is possible for Eric to tell passengers to walk from the beginning without a taxi ride if necessary. Also, it is not necessary for the taxi to drop passengers off at one of the specified stops — if two passengers specified blocks 13 and 15, Eric could choose to drop them off at block 14.

Example

Suppose Eric allows 2 stops and 4 passengers specify their destinations as 1, 5, 6, 8. To minimise the total number of blocks walked, the taxi could stop at 5 and 8. This means one passenger would walk to block 1, two would get off at 5 with one walking to 6, and the other would get off at 8. The total number of blocks walked is therefore 2.

Input (taxi.in)

The first line of input contains S, the number of stops Eric is prepared to make. The next line contains N, the number of passengers to be dropped off. The following N lines each contain a single positive integer specifying the block the passenger wants to get off at, from smallest to largest.

Sample input

```
2
4
1
5
6
8
```

Output (taxi.out)

You must output a line containing only a single integer, the minimum total number of blocks the passengers have to walk.

Sample output

```
2
```

Constraints

- $1 \leq S \leq 10$
- $1 \leq N \leq 100$
- $1 \leq$ largest block number ≤ 1000

50% constraints

- $1 \leq S \leq 10$
- $1 \leq N \leq 20$
- $1 \leq$ largest block number ≤ 20

Time limit

3 seconds.

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.

Collapsing mine

Author

Allen Jansen van Nieuwenhuizen

Introduction

Barry the mole has dug himself a very snug mine near Stilfontein. His tunneling is three-dimensional as he is able to dig along the three axes. He also likes chaos and thus his mine resembles a maze. One day while digging in the mine the earthquake prediction algorithm on his laptop goes haywire and tells him that an earthquake is about to strike, and that his whole mine will collapse.

Task

Your task is to write an algorithm that finds the route out of his mine, using the given starting and ending points. The maze is represented as a 3D grid, with each cell being either solid or open. The escape route must not visit any cell twice. There is guaranteed to be a unique solution that meets this requirement.

Example

In the example he has a $5 \times 5 \times 5$ maze. The input shows the start and end location along with the map. The output shows the route from the start to the end.

Input (mine.in)

The first line of the input contains three space-seperated integers, A B and C, where

- A is the number of cells along the X axis;
- B is the number of cells along the Y axis;
- C is the number of cells along the Z axis.

The second line contains another three space-separated integers representing the starting coordinates s_X, s_Y and s_Z in that order. The third line contains three integers representing the ending coordinates e_X,e_Y and e_Z in that order. The coordinates range from $(1, 1, 1)$ to (A, B, C).

The maze is given as B blocks. The first block describes the part of the maze with $y = 1$, the second describes the part with $y = 2$ and so on. Each block consists of C lines, with the first line corresponding to $z = 1$, the second to $z = 2$ and so on. Each line consists of A characters, with x running from 1 to A along the line. Each character is either a '`.`' for a solid cell or a '`0`' (number zero) for an open cell.

Sample input

```
5 5 5
2 1 2
2 5 2
.....
.0...
.....
.....
.....
.....
.0.0.
.0...
.000.
.....
.....
...0.
.....
...0.
.....
.....
.000.
.0.0.
.0.0.
.....
.....
.0...
.....
.....
.....
```

Output (mine.out)

The output must show the maze with the escape route marked in. The format is the same as for the input file, except that '`0`' (zero) indicates cells on the escape route while all other characters must be '`.`'.

Sample output

```
.....
.O...
.....
.....
.....
.....
.O...
.O...
.OOO.
.....
.....
.....
.....
...O.
.....
.....
.OOO.
...O.
...O.
.....
.....
.O...
.....
.....
.....
```

Constraints

- $3 \leq A, B, C \leq 65$
- $1 \leq s_X, e_X \leq A$
- $1 \leq s_Y, e_Y \leq B$
- $1 \leq s_Z, e_Z \leq C$

50% constraints

- $3 \leq A, B, C \leq 33$

Time limit

2 seconds.

Scoring

A correct answer scores 10 points, an incorrect answer 0.

Climber

Author

Keegan Carruthers-Smith

Introduction

While on your flight to the IOI, your plane fails and you crash into the middle of nowhere. Unfortunately you are the only survivor.

Fortunately you have your laptop with you, and a map indicating heights of the terrain and the closest village. You also use your map reading skills to figure out where on the map you are.

You also have extensive mountaineering skills, so you can navigate the landscape quite efficiently. Your mountaineering theory taught you that for every metre you climb up, you lose one unit of energy. You also know that for every 2 metres you descend, you gain one unit of energy: unfortunately the energy gained must be rounded down, so for example if you descend 5m, you only gain 2 units of energy. Also, this round down happens after every descent step: if you descend 5m, and **then** descend another 3m, you gain $2 + 1 = 3$ units of energy.

Task

Your task is to find a route from the start location to the end location so that your energy remains above zero at all times.

Example

Suppose you have an initial energy of 6 and you end up in an area with a width of 4 and a height of 3 with the following heights above sea level:

```
9 3 20 1
11 5 9 2
13 8 5 2
```

Say you start in block $(2, 1)$ (2^{nd} value, 1^{st} row, height of 3) and you end in block $(4, 2)$ (4^{th} value, 2^{nd} row, height of 2). If you travel South, South, East, East and North then you will reach the end location. Your energy will change from 6 to 4 to 1 to 2 to 3 to 3, and so remains above 0 at all times.

Input (climber.in)

The first line contains one integer, E, which indicates your initial energy. The second line contains two space seperated integers, S_X and S_Y, indicating the starting location. The third line contains two space seperated integers, E_X and E_Y, indicating the ending location. The fourth line contains two space seperated integers, W and H, indicating the width and the height of the map respectively. The next H lines each contain W space-separated integers, $h_{x,y}$. Each integer indicates the height above sea level in metres.

The upper-left co-ordinate is $(1, 1)$ and the lower right is (W, H).

Sample input

```
6
2 1
4 2
4 3
9 3 20 1
11 5 9 2
13 8 5 2
```

Output (climber.out)

The first line contains a single integer, N, indicating the number of times you will have to move from one block to another. The next N lines each contain a single uppercase ASCII character indicating the direction of movement. The possible characters are:

- N — North (Upwards)
- S — South (Downwards)
- W — West (Left)
- E — East (Right).

Sample output

```
5
S
S
E
E
N
```

Constraints

- $1 \leq E \leq 40000$
- $2 \leq W, H \leq 750$

- $0 \leq h_{x,y} \leq 10000$
- The starting location will not be the same as the ending location

50% constraints

- $1 \leq E \leq 40000$
- $2 \leq W, H \leq 250$
- $0 \leq h_{x,y} \leq 100$

Time limit

3 seconds.

Scoring

You will score 0% if:

- Your output format differs from the above prescribed format
- Your path does not finish at the ending location
- If at any time your energy is ≤ 0
- If your path goes outside of the bounds of the map.

Otherwise, you will score 100%.