South African Computer Olympiad
Third Round 2009
Day 1

Overview

Problem	**pieces**	**guardian**	**stacker**
Source	pieces.java pieces.py pieces.c pieces.cpp	N/A	stacker.java stacker.py stacker.c stacker.cpp
Input file	stdin	guardian.in	stdin
Output file	stdout	guardian.out	stdout
Time limit	0.2 seconds	N/A	0.2 seconds
Memory limit	128MiB	N/A	128MiB
Number of tests	10	10	10
Points per test	10	10	10
Detailed feedback	No	No	Yes
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

Artefact Pieces

Introduction

You are an avid relic hunter and have recently uncovered a mystical map leading to an ancient artefact, which is divided into two separate pieces. The two pieces are buried at different locations along a straight road, which runs for ten kilometres from west to east. The pieces of the artefact are located at the unknown positions A and B, measured in centimetres along the road from the western end.

You have a metal detector which will help you find the pieces, but it is not very powerful. It can only be turned on 50 times before the battery runs out. Each time it is switched on it will only tell you whether it is between the two pieces, to the west of both pieces, or to the east of both pieces; it then immediately turns off again. If the metal detector is exactly above one of the pieces, it will report that it is between the two.

Thankfully, each piece of the artefact is buried with the exact location of the other piece. So you only need to locate one of the pieces.

Task

Write a program that locates one of the artefact pieces with no more than 50 uses of the metal detector. The metal detector can only be used at whole-numbered positions (in centimetres).

Example

Your program probes at positions 0, 1, 2, 3 and 4. The metal detector reports "west", "between", "between", "between" and "east", respectively. Then the pieces are positions 1 and 3 and your program can report either of them.

Interaction

To probe at a position p ($0 \leq p \leq 1\,000\,000$ — a probe out of range may cause catastrophic failure, resulting in undefined behaviour!), your program must output "`P` p" followed by a newline, and then flush the output stream. It can then read in a single line containing a single character: "`W`" if the probe is to the west of the two pieces, "`E`" if it is to the east and "`B`" if it is between. If you attempt to probe more than 50 times, the metal detector your program will be terminated and score zero.

Output	Input
P 0	
	W
P 1	
	B
P 2	
	B
P 3	
	B
P 4	
	E
F 3	

Figure 1: *Example interaction*

When it has found one of the pieces at position f, your program must output "`F` f" followed by a newline, flush the output stream, and exit.

To flush the output stream in Python, import the module `sys` and then call `sys.stdout.flush()`. In Java, call `System.out.flush()`.

Constraints

- $0 \leq A < B \leq 1\,000\,000$

Additionally, in 40% of cases:

- $A \leq 49$

Time limit

0.2 seconds. Python multiplier: 10.

Testing

To test your program, you can provide a test file to the online testing facility in the following format: a single line containing two space-separated integers, A and B satisfying the above constraints.

Scoring

If your program interacts correctly, uses no more than 50 probes, and finds one of the pieces of the artefact, you will score 100% for that run. Otherwise you will score 0%.

Guardian Knights

Introduction

In the middle of the night, several UFOs landed in Fredville. Although they do not seem to be dangerous, the townspeople have decided to hire some knights to keep watch over them just in case.

They want to position the knights so that they can attack any one of the UFOs in a single move should the need arise. A knight moves by taking two steps in one direction followed by one at right angles. They cannot move any other way, but can jump over UFOs and other knights. A knight can guard several UFOs at once, as long as they are all a single move away.

The townspeople are not wealthy, so they want your help in placing the knights so that they can hire as few knights as possible.

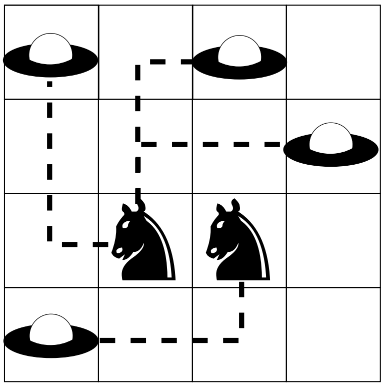
Task

Given the positions of all the UFOs, place as few knights as possible, ensuring that each UFO is a single move away from at least one knight.

A knight cannot be placed at the same position as a UFO or another knight.

Example

If the UFOs are at $(1,1),(3,1),(1,4)$ and $(4,2)$, they can be guarded by knights at $(2,3)$ and $(3,3)$. In the figure, the y coordinates are shown increasing *downward*.



Input (guardian.in)

The first line of input contains a single positive integer, N, the number of UFOs. The next N lines each contain two integers, x_{u} and y_{u}, the coordinates of the UFOs.

Sample input

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

Output (guardian.out)

The first line of output should contain a single positive integer, K ($K \leq N$): the number of knights. The next K lines should each contain two integers, x_{k} and y_{k} ($-1\,000\,000\,010 \leq x_{\mathrm{k}}, y_{\mathrm{k}} \leq 1\,000\,000\,010$): the coordinates of the knights.

Sample output

```
2
2 3
3 3
```

Constraints

- $-1\,000\,000\,000 \leq x_{\mathrm{u}}, y_{\mathrm{u}} \leq 1\,000\,000\,000$
- $1 \leq N \leq 10\,000$

It is guaranteed to be possible to guard every UFO.

Submission

Each submitted output file will be checked to ensure the file is formatted correctly and that each UFO is guarded by at least one knight.

Scoring

Invalid submissions will score 0.

Valid submissions for each case will be compared to the best submission for that case by any contestant. If your submission is the best you will score 10. Otherwise, if the best submission uses B knights, then your submission will score $\lfloor \frac{9B}{K} \rfloor$.

Stacker

Introduction

Stacker is a two player game. There is a shared stack of N numbers and each player takes it in turns to remove between 1 and M values (inclusive) from the top of the stack. A player's score is the sum of every value they remove from the stack.

The game is over when every value has been removed from the stack. The objective of the game is to maximise your score.

Task

The opposing player plays such that his final score is as large as possible. Assuming you start and both players play perfectly (they can plan ahead through the whole game, and never play sub-optimally), find the maximum score you can get in a game of Stacker.

Example

In the sample input you can remove $1, 2$ or 3 values per turn and your stack initially looks like this:

$$5\ 2\ 0\ 1\ 4\ 3\ 5\ 2\ 0\ 0$$

At each turn you remove values from the right. The best you can score is 12, causing the opponent to score 10. This is done by:

Player	Removed	Stack	Score
		5 2 0 1 4 3 5 2 0 0	0 - 0
You	0	5 2 0 1 4 3 5 2 0	0 - 0
Opponent	0, 2, 5	5 2 0 1 4 3	0 - 7
You	3, 4	5 2 0 1	7 - 7
Opponent	1, 0, 2	5	7 - 10
You	5		12 - 10

Input (stdin)

The first line of the input contains two space-separated integers: N and M. The next N lines each contain a single integer, x_i. These are the values on the stack at the start of the game. The first value is at the bottom of the stack (so will be removed *last*). The last value is at the top of the stack (so will be removed *first*).

Sample input

```
10 3
5
2
0
1
4
3
5
2
0
0
```

Output (stdout)

The output contains a single integer T. T is the largest value you can score assuming the opposing player maximises his score.

Sample output

```
12
```

Constraints

- $1 \leq N \leq 220\,000 = 22 \times 10^4$
- $1 \leq M \leq N$
- $0 \leq x_i \leq 5000$

Additionally, in 50% of the test cases:

- $1 \leq N \leq 5\,000 = 5 \times 10^3$
- $1 \leq M \leq 100$

Additionally, in 30% of the test cases:

- $1 \leq N \leq 10$
- $1 \leq M \leq 5$

Time limit

0.2 seconds. Python multiplier: 10. Java multiplier: 3.

Detailed feedback

Detailed feedback is enabled for this problem.

Scoring

A correct solution will score 100% while an incorrect solution will score 0%.