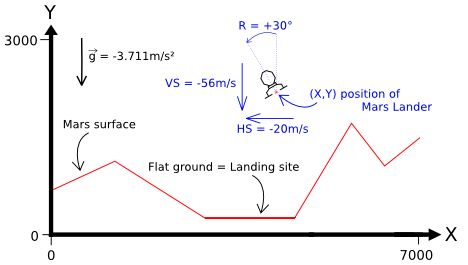
## The Goal

The goal for your program is to safely land the "Mars Lander" shuttle, the landing ship which contains the Opportunity rover. Mars Lander is guided by a program, and right now the failure rate for landing on the NASA simulator is unacceptable.  
  
This puzzle is the second level of the "Mars Lander" trilogy. The controls are the same as the previous level but you must now control the angle in order to succeed.

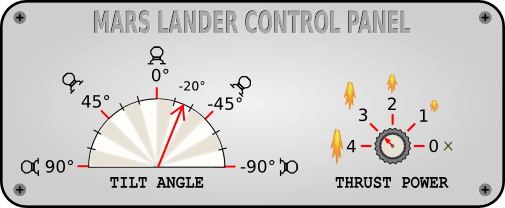
## Rules

Built as a game, the simulator puts Mars Lander on a limited zone of Mars sky.



The zone is wide and high.  
  
There is a unique area of flat ground on the surface of Mars, which is at least meters wide.

Every second, depending on the current flight parameters (location, speed, fuel ...), the program must provide the new desired tilt angle and thrust power of Mars Lander:



Angle goes from to . Thrust power goes from to .

The game simulates a free fall without atmosphere. Gravity on Mars is . For a thrust power of X, a push force equivalent to X m/s² is generated and X liters of fuel are consumed. As such, a thrust power of in an almost vertical position is needed to compensate for the gravity on Mars.  
  
For a landing to be successful, the ship must:

* land on flat ground
* land in a vertical position (tilt angle = 0°)
* vertical speed must be limited ( ≤ 40m/s in absolute value)
* horizontal speed must be limited ( ≤ 20m/s in absolute value)

## Note

Tests and validators are only slightly different. A program that passes a given test will pass the corresponding validator without any problem.

## Game Input

The program must first read the initialization data from standard input. Then, within an infinite loop, the program must read the data from the standard input related to Mars Lander's current state and provide to the standard output the instructions to move Mars Lander.

Initialization input

Line 1: the number surfaceN of points used to draw the surface of Mars.  
Next surfaceN lines: a couple of integers landX landY providing the coordinates of a ground point. By linking all the points together in a sequential fashion, you form the surface of Mars which is composed of several segments. For the first point, landX = and for the last point, landX =

Input for one game turn

A single line with 7 integers: X Y hSpeed vSpeed fuel rotate power

* X,Y are the coordinates of Mars Lander (in meters).
* hSpeed and vSpeed are the horizontal and vertical speed of Mars Lander (in m/s). These can be negative depending on the direction of Mars Lander.
* fuel is the remaining quantity of fuel in liters. When there is no more fuel, the power of thrusters falls to zero.
* rotate is the angle of rotation of Mars Lander expressed in degrees.
* power is the thrust power of the landing ship.

Output for one game turn

A single line with 2 integers: rotate power :

* rotate is the desired rotation angle for Mars Lander. Please note that for each turn the actual value of the angle is limited to the value of the previous turn .
* power is the desired thrust power. = off. = maximum power. Please note that for each turn the value of the actual power is limited to the value of the previous turn .

Constraints

2 ≤ surfaceN < 30  
0 ≤ X < 7000  
0 ≤ Y < 3000  
-500 < hSpeed, vSpeed < 500  
0 ≤ fuel ≤ 2000  
-90 ≤ rotate ≤ 90  
0 ≤ power ≤ 4  
Response time per turn ≤ 100ms

Example

Initialization input

6 (surfaceN) Surface made of 6 points

0 1500 (landX landY)

1000 2000 (landX landY)

2000 500 (landX landY) Start of flat ground

3500 500 (landX landY) End of flat ground

5000 1500 (landX landY)

6999 1000 (landX landY)

No output expected

You can ignore this but you need to read the values.

Input for turn 1

5000 2500 -50 0 1000 90 0 (X Y hSpeed vSpeed fuel rotate power)

Output for turn 1

-45 4 (rotate power)

Requested rotation to the right, maximum thrust power

Input for turn 2

4950 2498 -51 -3 999 75 1 (X Y hSpeed vSpeed fuel rotate power)

Tilt angle changed only by 15° and thrust power only by 1

Output for turn 2

-45 4 (rotate power)

Same request as previous turn

Input for turn 3

4898 2493 -53 -6 997 60 2 (X Y hSpeed vSpeed fuel rotate power)

Output for turn 3

-45 4 (rotate power)

Same request as previous turn