Problem 136: Anti-Asteroid Weapon

Difficulty: Easy

Author: Luis Rivera, Aguadilla, Puerto Rico, United States

Originally Published: Code Quest 2021

Problem Background

Earth is under attack… and you're our only hope!

A cluster of asteroids is quickly approaching Earth's atmosphere, and each of them is large enough to cause severe damage to all life on the planet. To protect us from this threat, world governments have worked with Lockheed Martin to develop a top-secret asteroid-destroying weapon to obliterate the space rocks without a trace.

The bad news is that the weapon takes about 20 minutes to charge between each shot… but the good news is that the asteroids are conveniently arriving in 20 minute intervals, so we should be able to shoot each one before it can impact us.

Problem Description

Your team needs to design a targeting algorithm to determine which of the asteroids needs to be shot first. The recharge delay on the weapon means that it's critically important that asteroids be destroyed in the correct order. Shooting an asteroid in the wrong order might allow another one to hit the Earth while the weapon is recharging.

NASA and the European Space Agency have been able to map the locations of each of the asteroids on a two-dimensional coordinate grid, with the Earth at the origin (0,0). You'll need to print out a list of the incoming asteroids in order of increasing distance from Earth. To calculate the distance from the origin point, you can use this formula:

$$d = \sqrt{x^2 + y^2}$$

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include:

- A line containing a positive integer, A, representing the number of asteroids
- A lines containing two integers separated by spaces, representing the X and Y coordinates of each incoming asteroid, respectively.

```
2
3
2 1
1 1
5 5
4
2 2
1 7
-1 0
1 1
```

Sample Output

For each test case, your program must print one line for each incoming asteroid, containing the coordinates of the asteroid as presented in the input. Asteroids must be printed in order by increasing distance from Earth.

```
1 1
2 1
5 5
-1 0
1 1
2 2
1 7
```