Computers often break during an Informatics competition. Therefore, there are people in the competition hall who are responsible for the technical support. For the purpose of the task, we will call them *runners*.

The hall can be represented as the 2D space, where at the beginning of the competition we can choose where to place each *runner* as a pair of . Every *runner* also has their own speed.

During the competition, exactly computers will break, each characterized by a pair . 1 *runner* must be selected to go from their position to in time and fix the computer, which happens in a negligible amount of time.

Their new position will now be .

Computers break in such a way that all *runners* are free at the time of computer break, i.e. the -th computer will break only after the -th computer is fixed.

We want to minimize the total amount of time a contestant sits with a broken computer, waiting for the respective *runner* to come and fix it.

**Input**

The first line of the **runners.in** file contains the numbers and . The next lines contain one number each – - the speed of the *runner*.

Each of the next lines contains two numbers for the position of the respective broken computer.

**Output**

On lines in the file **runners.out**, print 1 pair of numbers - the positions of the *runners* at the start of the competition. They must satisfy the constraints .

On the next lines, print 1 number each - the index of the *runner* that will repair the corresponding computer.

**Scoring**

For each test, let *minScore* be the smallest score among all participants' scores and *yourScore* be your score. You will be awarded multiplied by the amount of points for the test.

**Constraints**

*,* has at most 6 decimal digits.

**Time limit: 5 sec.**

**Memory limit: 256 MB.**

The tests are distributed as follows:

|  |  |
| --- | --- |
| **Percentage** |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Sample test**

|  |  |
| --- | --- |
| **Input (runners.in)** | **Output (runners.out)** |
| 5 2  1.300000  1.800000  3 8  6 7  9 4  10 2  1 5 | 3 8  6 7  1  2  2  2  1 |

**Example explanation**

The sample test is only for an explanation, in all real tests .

*Runner* 1 travels for 0 time to position .

*Runner* 2 travels for 0 time to position .

*Runner* 2 travels for time to position .

*Runner* 2 travels for time to position .

*Runner* 1 travels for time to position .

The total time in which a competitor is waiting with a broken computer is .

**Tests generation**

The numbers are randomly generated in the respective intervals that bound them (each number in the interval has an equal chance).