Foraging for Food

You've just arrived at RPI to start your career as a graduate student! Congratulations! You've immediately applied yourself to research and TA work, churning out proofs and undocumented research code by day and grading undergraduates' assignments by night. All of that work makes you VERY hungry; luckily there are frequently events around campus that offer free food. However, you don't want to spend too long away from work getting all that free pizza. You decide to create a program that lets you know which events to go to in order to get as much food during your lunch break as possible.

Input Format

Input: The first line contains the integer length of your lunch break, measured in *Arbitrary Time Units (TM)*. You need to be back to work before time runs out! The next lines contain geographical coordinates of each location (floating point numbers). Once you have been to an event, you cannot go again! The time it takes you to travel between events is equal to 1 + the number of events you have visited for that run, times the distance between your current location and your goal. Since you are eating the food at the events, returning to your office in the middle of a trip wouldn't do you any good to reduce your movement cost. Your office is always located at 0,0. Example:

```
20
0.00 3.00
0.5 2.5
-1.0 0.75
13.0 5.00
```

Constraints

- You can't visit an event more than once

- You must complete your round trip (to and from your office) within the time span of your lunch break

- Returning to your office between locations doesn't reset the number of places you have visited -- i.e it doesn't reset your cost

Output Format

Output: The output should be all paths that cost you the least time for the greatest amount of food. In other words, prioritize more food over saving time. These should be given as each point you travel to, formatted with two decimal points of precision per coordinate. The last line of each path should give the time cost of the path (float, two decimals) and the number of locations visited (integer) Example:

```
0 0
0.0 3.0
0.5 2.5
-1.0 0.75
0 0
16.33 3
```

Sample Input 0

```
20
0 3
0.5 2.5
-1 .75
13 5
```

Sample Output 0

```
0.00 0.00
0.00 3.00
```

0.50 2.50
-1.00 0.75
0.00 0.00
16.33 3