Marathon Match

Problem Statement

Contest: Round 1

Normal view

Problem: SmallPolygons

Problem Statement

You are given a set of points with integer coordinates on a two-dimensional plane and an integer **N**. You have to construct at most **N** polygons which have these points as vertices. More specifically,

- Each vertex of each polygon must be a point from the given set.
- Each point from the given set must belong to exactly one of the constructed polygons.
- Each polygon must be simple.
- Edges of different polygons can't intersect (but one polygon can lie completely within another).

Your task is to minimize the sum of areas of polygons that you constructed.

Implementation Details

Your code should implement one method `choosePolygons(int[] points, int N)`. **points** gives you the set of points: point i (0-based) has coordinates (points[2*i], points[2*i+1]). **N** is the number of polygons. You must return a set of polygons you've constructed from these points as a `String[]`. Each element of your return must describe one polygon and should be formatted as a space-separated list of its vertices in clockwise or counterclockwise order. Each vertex is given as its 0-based index in **points**.

Scoring

Your score for an individual test case will be the sum of areas of polygons you've constructed. If your return has invalid format or specifies any invalid polygons, your score for the test case will be 0. Your overall score will be calculated in the following way: for each test case where your score is not 0, you get 1 point for each competitor you beat on this test case (i.e., your score on a test case is smaller than this competitor's score) and 0.5 points for each competitor you tie with (a tie with yourself is not counted); finally, the sum of points is divided by (the number of competitors - 1).

Test Case Generation

Each test case is generated as follows:

- The number of points in the set is chosen either between 20 and 99, between 100 and 499, or between 500 and 1500, all inclusive (either interval is chosen with equal probability).
- The points coordinates are chosen between 0 and 699, inclusive, so that all points are distinct.
- The number of polygons allowed is chosen between 2 and 20, inclusive.
- All values are chosen uniformly and independently, at random.

Note that example 0 (seed 1) does not conform to these constraints.

Tools

An offline tester/visualizer is available here. You can use it to test/debug your solution locally. You can also check its source code for exact implementation of test case generation and score calculation. It also allows manual play.

Definition

Class:	SmallPolygons
Method:	choosePolygons
Parameters:	int[], int
Returns:	String[]
Method signature:	String[] choosePolygons(int[] points, int N)
(be sure your method is public)	

Notes

- The number of points will be between 20 and 1500, inclusive.
- The maximal allowed number of polygons N will be between 2 and 20, inclusive.
- The coordinates of points will be chosen between 0 and 699, inclusive. Each coordinate is chosen uniformly and independently. All points will be distinct.
- The time limit is 10 seconds per test case (this includes only the time spent in your code). The memory limit is 1024 megabytes.
- There is no explicit code size limit. The implicit source code size limit is around 1 MB (it is not advisable to submit codes of size close to that or larger). Once your code is compiled, the binary size should not exceed 1 MB.
- The compilation time limit is 30 seconds. You can find information about compilers that we use and compilation options here.
- There are 10 example test cases and 100 full submission (provisional) test cases.

Examples

0)

```
seed = 1
NP = 10
N = 3
```

1)

```
seed = 2
NP = 90
N = 17
```

2)

```
seed = 3
NP = 166
N = 16
```

3)

```
seed = 4
NP = 22
N = 13
```

4)

```
seed = 5
NP = 1368
N = 6
```

5)

```
seed = 6
NP = 61
N = 11
```

6)

```
seed = 7
NP = 32
N = 20
```

7)

```
seed = 8
NP = 254
N = 20
```

8)

```
seed = 9
NP = 205
N = 4
```

9)

```
seed = 10
NP = 490
N = 19
```
