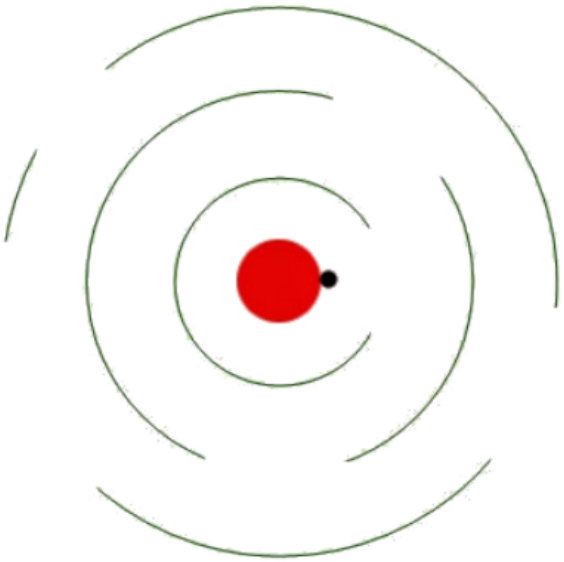
UVa Online Judge

12728 Super Circumference

Skyrk has developed a game called Super Circumference. Its simplicity is only equaled by its monumental difficulty. The goal of Super Circumference is to control a point which circles around a central circumference attempting to avoid contact with incoming circular walls.

The point circles around at the speed of one full revolution per second. A level has several sets of incoming circular walls which the point has to avoid contact. An incoming wall can be regarded as a circumference sector. The set of walls come at the speed of one set every P seconds. The player wins the game if he successfully evade all walls.



The red circle is the center which the black point circles around. The incoming green walls approach the red circle. A brand new level has been created with N sets of walls, to adjust its difficulty properly, you are to find the minimum possible P that still makes the level possible to complete.

Input

The first line contains T ($T \leq 100$) — the number of test cases, after this line T test cases follows. The first line of a test case contains one integer N ($1 \leq N \leq 10^4$) — the number of sets of walls. Next N lines contain the description of the set of walls in the following form: First the number K ($0 \leq K \leq 10$) — the number of walls this set has. Next, there'll be K pairs of floating point numbers X and Y ($0 \leq X, Y < 2\pi$) — the wall starts at X and extends along the circumference in clockwise direction until Y. Walls of the same set do not intersect. The circumference sector covered by a wall will not be empty and will not be a full circumference. When the game starts the first wall will hit the center after P seconds and the black point can start at any position. The sets of walls are given in the order they approach the center.

Output

For each test case print a line containing '`Case #`X`:  `Y', where X is the case number, starting at 1, and Y is the minimum possible P that makes the level possible to complete. Y should be rounded up to 4 digits after the decimal point. The input will be in a way that errors up to 10^{-5} will still give the correct answer.

Sample Input

```
2
3
1 1 5.28
2 1.5 4.28 5 1
3 0 2 2.5 3 3.8 4.5
2
```

```
2 0 1 2 3
2 0.9 2.1 2.9 0.1
```

Sample Output

```
Case #1: 0.0446
Case #2: 0.0159
```