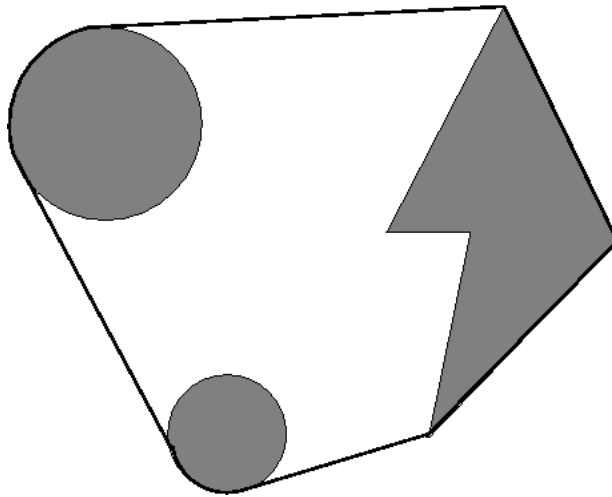| D	# Crime Scene
	Input: Standard Input
	Output: Standard Output

You have been appointed as a programmer in your local police station (they watch a lot of TV shows like CSI, Numb3rs etc., and though having a computer genius among them would be pretty cool). They have given you a desktop, Now, you can do all the "impossible" tasks, that are being done in the shows frequently (like magnifying an image infinite times with no pixelation, or building systems, where the only way to reboot is to shut down the main power of the building).

But due to the recent increase in crime and also the economic problems, the station is planning to make some budget cuts. One of the cut is in the use of crime scene tapes. So, at each crime scene, they want to use the minimal possible amount of tape.

Two circles and a polygon in the crime scene. The optimal taping is shown in dark.

You are given all the objects in the crime scene. The objects are all circular or in the shape of a polygon. Return the minimum amount of crime scene tapes to surround them.

Input

First line contains *T*, the number of test cases. Each test case starts with *N*, the number of objects in the crime scene. Each of the next *N* lines describe an object in the crime scene. The first character in the line will be either `'c'` or `'p'`, denoting whether the object is a circle, or a polygon. If it is a circle, then, it will be followed by 3 real numbers, `(cx, cy)` the center of the circle and radius of the circle.
In case of polygon, it will be followed by an integer *K*, the number of sides the polygon will have, followed by *K* pairs of real numbers, *(xi, yi)* the location of the *i*-th vertex of the polygon.
All polygons in this problem will be simple.

Output

For each test case, output the case number and the optimal length of the tape. Output 6 digits after the decimal point.

Constraints

- **N** <= 100
- **K** <= 10
- All radii will be greater than 0.
- All input values will be between -1000 and +1000.

Sample Input

Sample Input	Output for Sample Input
1 2 c 0 0 2 p 4 -1 2 1 2 1 -2 -1 -2	Case #1: 13.148009

Problem Setter: Manzurur Rahman Khan, Special Thanks: Samee Zahur