CS2810 - OOAIA Lab 14
Fantastic Beasts and Paths To Find Them

Date: 22 April 2025

Link: HackerRank Link

Problem Description

Newt Scamander, our favourite magizoologist, is up to an adventure again.

He is on a covert mission across the **Shattered Archipelagos**, a hidden chain of magical islands scattered across the sea — each shaped like a **triangle, rectangle, or circle**. Each island harbors a rare fantastic beast, and Newt must rescue them before a violent magical storm sweeps through.

But there's a problem.

Newt is still under an international travel ban imposed by the British Ministry of Magic, which means he's operating *completely off the books*. To leave no trace of his journey, he must **destroy every island he lands on** — vanishing it into thin air with a Disillusionment Charm and a blast of spellwork.

Newt is a master of magical manipulation. Before every hop from one island to another, he may cast a spell to **rotate any of the islands about their center** by any amount as he pleases. The islands remain rooted at their centers, but can be oriented however he wants. If two islands **share a common point or touch** then Newt can jump between them.

Your task is to help Newt determine whether it is possible for him to visit **every island exactly once**, starting and ending wherever he likes, destroying each one after visiting it, and never revisiting any. If possible, you need to print such a path. Else, you need to print the longest path such that every vertex in that path is visited exactly once using that path.

Design Specifications

Implement an abstract base class `Island`. Create derived classes for differently shaped-islands and override the appropriate methods. Overload the stream insertion and extraction operators for the `Island` class. The design decisions for `Island` are left up to you. However, design decisions will be evaluated based on good programming practices.

Input and Output Format

Input Format:

1. Each test case corresponds to a single setup of islands
2. The first line of each test case contains **n**, which represents the number of islands.
3. The next `n` lines each represent one of the following types of islands:
    1. `RECTANGLE <IslandID> x1 y1 x2 y2 x3 y3 x4 y4`
    2. `TRIANGLE <IslandID> x1 y1 x2 y2 x3 y3`
    3. `CIRCLE <IslandID> x_center y_center radius`

Here, IslandID is a string.

Output Format:

Output `NO` if it is not possible for Newt to visit every island exactly once under the given conditions. Following NO, print the length of the longest path such that every vertex in that path is visited exactly once using that path. Also print such a path.

Otherwise, print `YES` followed by any valid visit order.

To print a path, you must print the IDs of the islands in the path.

Please refer to sample test cases to understand output format.

Constraints

$1 \leq n \leq 17$

For all coordinate values x_i, y_i belonging to any island, $-100 \leq x_i, y_i \leq 100$

All input coordinates are integer values.

Notes

It is guaranteed that the centroid of every shape will be at integer coordinates.

You are free to use the `sqrtl` function for computing square roots.

Center of a shape stands for its centroid. For Triangles, please use its centroid (and not circumcentre).

The setters and tester's solution for this problem involves a dynamic programming technique known as Dynamic Programming over Subsets. This requires knowledge of representing sets using binary numbers. You can learn more about representation of sets here.
Here are some slides for your reference.