Fantastic Beasts and Paths To Find Them

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 positions, but can be oriented however he wants. If two islands **share a common point or touch** then Newt can hop 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.

Input Format

Each test case corresponds to a single setup of islands

The first line of each test case contains n, which represents the number of islands.

The next n lines each represent one of the following types of islands:

1. $\texttt{RECTANGLE}\ \langle\texttt{IslandID}\rangle\ x_1\ y_1\ x_2\ y_2\ x_3\ y_3 x_4\ y_4$
2. $\texttt{TRIANGLE}\ \langle\texttt{IslandID}\rangle\ x_1\ y_1\ x_2\ y_2\ x_3\ y_3$
3. $\texttt{CIRCLE}\ \langle\texttt{IslandID}\rangle\ x_{\text{center}}\ y_{\text{center}}\ \text{radius}$

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.

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 shapes in the path.

Please refer to sample test cases to understand output format.

Sample Input 0

```
3
TRIANGLE T1 0 0 3 0 0 3
RECTANGLE R1 1 1 1 3 3 1 3 3
CIRCLE C1 0 0 2
```

Sample Output 0

```
YES
T1 R1 C1
```

Explanation 0

Here, all islands overlap. Thus, Newt can reach anny island from any other island. Thus, a required path exists.

Sample Input 1

```
4
RECTANGLE R1 0 0 2 0 2 2 0 2
TRIANGLE T1 3 3 6 3 3 6
CIRCLE C1 5 5 1
RECTANGLE R2 5 1 7 1 7 3 5 3
```

Sample Output 1

```
NO
3
C1 T1 R2
```

Explanation 1

There are four islands: a rectangle at the origin (R1), a triangle (T1), a small circle at $(5, 5)$ (C1), and a rectangle (R2).

R1 is too far from all other islands and can't connect to any of them.

However, C1 is close enough to T1, and T1 is close enough to R2, so they form a connected path of length 3.

Since the longest path only visits 3 out of 4 islands, the output is **NO** and the path is C1 T1 R2.

Sample Input 2

```
5
TRIANGLE T1 0 0 3 6 6 0
RECTANGLE R1 -2 -2 2 -2 2 2 -2 2
CIRCLE C1 10 -10 5
RECTANGLE R2 -4 4 0 4 0 0 -4 0
TRIANGLE T2 1 1 4 4 7 -2
```

Sample Output 2

```
NO
4
T1 R1 R2 T2
```

Explanation 2

There are five islands: a triangle (T1), a rectangle around the origin (R1), a far away circle at $(10, -10)$ (C1), a rectangle in the top left quadrant (R2), and another triangle (T2).

C1 is too far from all other islands and cannot connect to any of them.

The other four islands are close enough to form a connected path. So, the longest path visits 4 shapes and the output is **NO** with the path T1 R1 R2 T2.