# **Sets-STL**



Sets are a part of the C++ STL. Sets are containers that store unique elements following a specific order. Here are some of the frequently used member functions of sets:

• Declaration:

```
set<int>s; //Creates a set of integers.
```

• Size:

```
int length=s.size(); //Gives the size of the set.
```

• Insert:

```
s.insert(x); //Inserts an integer x into the set s.
```

• Erasing an element:

```
s.erase(val); //Erases an integer val from the set s.
```

• Finding an element:

```
set<int>::iterator itr=s.find(val); //Gives the iterator to the element val if it is found otherwise
returns s.end() .
Ex: set<int>::iterator itr=s.find(100); //If 100 is not present then it==s.end().
```

To know more about sets click Here. Coming to the problem, you will be given Q queries. Each query is of one of the following three types:

1 x: Add an element x to the set.

2x: Delete an element x from the set. (If the number x is not present in the set, then do nothing).

 $\mathbf{3} \ x$ : If the number x is present in the set, then print "Yes"(without quotes) else print "No"(without quotes).

#### **Input Format**

The first line of the input contains Q where Q is the number of queries. The next Q lines contain 1 query each. Each query consists of two integers y and x where y is the type of the query and x is an integer.

#### **Constraints**

$$1 <= Q <= 10^5$$

$$1 <= y <= 3$$

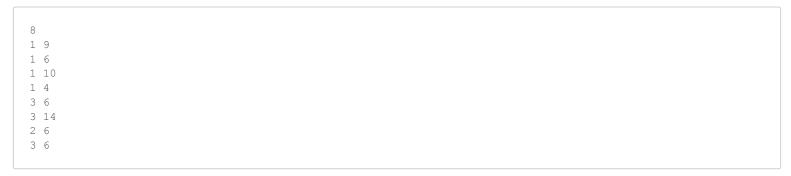
$$1 <= x <= 10^9$$

### **Output Format**

For queries of type 3 print "Yes"(without quotes) if the number  $\boldsymbol{x}$  is present in the set and if the number is not present, then print "No"(without quotes).

Each query of type  ${\bf 3}$  should be printed in a new line.

### **Sample Input**



## **Sample Output**

