1. Python: Multiset Implementation

A *multiset* is the same as a set except that an element might occur more than once in a multiset. Implement a multiset data structure in Python. Given a template for the *Multiset* class, implement 4 methods:

- *add(self, val)*: adds *val* to the multiset
- *remove(self, val)*: if *val* is in the multiset, removes *val* from the multiset; otherwise, do nothing
- __contains__(self, val): returns True if *val* is in the multiset; otherwise, it returns False
- __len__(self): returns the number of elements in the multiset

Additional methods are allowed as necessary.

The implementations of the 4 required methods will be tested by a provided code stub on several input files. Each input file contains *several* operations, each of one of the below types. Values returned by *query* and *size* operations are appended to a *result* list, which is printed as the output by the provided code stub.

- add val: calls add(val) on the Multiset instance
- remove val: calls remove(val) on the Multiset instance
- query val: appends the result of expression *val* in *m*, where *m* is an instance of Multiset, and appends the value of that expression to the *result* list
- size: calls len(m), where *m* is an instance of Multiset, and appends the returned value to the *result* list

Complete the class Multiset in the editor below with the 4 methods given above (add, remove, __contains__, and __len__).

Constraints

- $1 \le$ number of operations in one test file $\le 10^5$
- if *val* is a parameter of operation, then *val* is an integer and $1 \le val \le 10^9$

▼ Input Format Format for Custom Testing

In the first line, there is a single integer, *q*, denoting the number of queries.
Then, *q* lines follow. In the i^{th} of them, there is a string denoting an operation and optionally an integer denoting the parameter of the operation.

▼ Sample Case 0

Sample Input

```
STDIN       Function
-----       --------
12      →   number of queries, q = 12
query 1 →   operations = ["query 1", "add 1", ..., "query 2", "size"]
add 1
query 1
remove 1
query 1
add 2
add 2
size
query 2
remove 2
query 2
size
```

Sample Output

```
False
True
False
2
True
True
1
```

Explanation

There are 12 operations to be performed. Start with an empty multiset: *multiset = []*.

1. The first operation asks if 1 is in the multiset. It is not, so False is appended to the result: *result = [False]*.
2. The second operation adds 1 to the multiset: *multiset = [1]*.
3. The third operation asks if 1 is in the multiset. It is now, so True is appended to the result: *result = [False, True]*.
4. The fourth operation removes 1 from the multiset: *multiset = []*.
5. The fifth operation asks if 1 is in the multiset. It is not, so False is appended to the result: *result = [False, True, False]*.
6. The sixth operation adds 2 to the multiset: *multiset = [2]*.
7. The seventh operation adds 2 to the multiset: *multiset = [2, 2]*.
8. The next operation asks what is the size of the multiset: *result = [False, True, False, 2]*.
9. The next operation asks if 2 is in the multiset. It is, so True is appended to the result: *result = [False, True, False, 2, True]*.
10. The next operation removes 2 from the multiset: *multiset = [2]*
11. The next operation asks if 2 is in the multiset. It is, so True is appended to the result: *result = [False, True, False, 2, True, True]*.
12. Finally, the last operation asks for the size of the multiset and the length, 1, is appended to the result. *result = [False, True, False, 2, True, True, 1]*

▼ Sample Case 1

Sample Input

```
STDIN       Function
-----       --------
3       →   number of queries, q = 3
size    →   operations = ["size", "add 17", "size"]
add 17
size
```

Sample Output

```
0
1
```

Explanation

There are 3 operations to be performed. Start with the empty multiset: *multiset = []*.

1. The first asks what is the size of the multiset. Since the multiset is empty, 0 is appended to the result: *result = [0]*.
2. The second operation adds 17 to the multiset: *multiset = [17]*.
3. The third operation asks what is the size of the multiset. 1 is appended to the result: *result = [0, 1]*.