Set Mutations

Problem Statement

We have seen the applications of union, intersection, difference and symmetric difference operations, but these operations did not make any changes or mutation to the set.

We can use the following operations to create mutations to set :

.update() or **|=**
Update the set, adding elements from an iterable/another set.

```
>>> H = set("Hacker")
>>> R = set("Rank")
>>> H.update(R)
>>> print H
set(['a', 'c', 'e', 'H', 'k', 'n', 'r', 'R'])
```

.intersection_update() or **&=**
Update the set, keeping only elements found in it and an iterable/another set.

```
>>> H = set("Hacker")
>>> R = set("Rank")
>>> H.intersection_update(R)
>>> print H
set(['a', 'k'])
```

.difference_update() or **-=**
Update the set, removing elements found in an iterable/another set.

```
>>> H = set("Hacker")
>>> R = set("Rank")
>>> H.difference_update(R)
>>> print H
set(['c', 'e', 'H', 'r'])
```

.symmetric_difference_update() or **^=**
Update the set, keeping only elements found in either set, but not in both.

```
>>> H = set("Hacker")
>>> R = set("Rank")
>>> H.symmetric_difference_update(R)
>>> print H
set(['c', 'e', 'H', 'n', 'r', 'R'])
```

TASK
You are given a set **A** and **N** numbers of other sets. These **N** sets have to perform some specific mutation operations to set **A**.

Your task is to execute those operations and print the sum of elements of set **A**.

Input Format

First line contains, number of elements in set **A**.
Second line contains, space separated list of elements of set **A**.

Third line contains, **N**, number of other sets.
Next *2*N* lines are divided into **N** parts of two lines each.
First line of each part contains, space separated entries of *operation name* and *length of other set*.
Second line of each part contains, space separated list of elements of other set.

0 < len(set(**A**)) < 1000
0 < len(otherSets) < 100
0 < N < 100

Output Format

Output the sum of elements of set **A**.

Sample Input

```
16
1 2 3 4 5 6 7 8 9 10 11 12 13 14 24 52
4
intersection_update 10
2 3 5 6 8 9 1 4 7 11
update 2
55 66
symmetric_difference_update 5
22 7 35 62 58
difference_update 7
11 22 35 55 58 62 66
```

Sample Output

```
38
```

Explanation

After first operation, i.e., *intersection_update operation*, we get :
set **A** = set([1, 2, 3, 4, 5, 6, 7, 8, 9, 11])

After second operation, i.e., *update operation*, we get :
set **A** = set([1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 55, 66])

After third operation, i.e., *symmetric_difference_update operation*, we get :
set **A** = set([1, 2, 3, 4, 5, 6, 8, 9, 11, 22, 35, 55, 58, 62, 66])

After fourth operation, i.e., *difference_update operation*, we get :
set **A** = set([1, 2, 3, 4, 5, 6, 8, 9])

Sum of elements of set **A** after these operations is **38**.