Mutable operations of set

add(*elem*)
Add element *elem* to the set.

Example:

```
>>> A=set()
>>> print(A)
set()
>>> A.add(10)
>>> A.add(20)
>>> A.add(30)
>>> A.add(40)
>>> A.add(50)
>>> print(A)
{40, 10, 50, 20, 30}
>>> A.add(60)
>>> print(A)
{40, 10, 50, 20, 60, 30}
>>> A.add(70)
>>> print(A)
{70, 40, 10, 50, 20, 60, 30}
```

https://www.hackerrank.com/challenges/py-set-add/problem?isFullScreen=false

```
N=int(input())
A=set()
for i in range(N):
    stamp=input()
    A.add(stamp)

print(len(A))
```

Example:

```
list1=[1,2,3,3,4,5,2,1,5,2,3,4,1,2,3]
A=set(list1) # {1,2,3,4,5}
```

```
for value in A:
    c=list1.count(value)
    print(f'{value}-->{c}')
```

Output:

```
1-->3
2-->4
3-->4
4-->2
5-->2
```

Example:

```
# input --> "abcaabdddc"
# ouput --> 3a2b2c3d

str1=input("Enter any string ") # abcab
A=set(str1) # {'a','b','c'}
str2=""
list2=list(A)
list2.sort()
for s in list2:
    c=str1.count(s)
    str2=str2+str(c)+s

print(str1)
print(str2)
```

Output:

```
Enter any string abcabc
abcabc
2a2b2c
```

remove(*elem*)
Remove element *elem* from the set. Raises KeyError if *elem* is not contained in the set.

Example:

```
>>> A={10,20,30,40,50}
>>> print(A)
{50, 20, 40, 10, 30}
>>> A.remove(10)
>>> print(A)
{50, 20, 40, 30}
>>> A.remove(20)
>>> print(A)
{50, 40, 30}
>>> A.remove(10)
Traceback (most recent call last):
  File "<pyshell#25>", line 1, in <module>
    A.remove(10)
KeyError: 10
```

discard(*elem*)
Remove element *elem* from the set if it is present.

```
>>> A={10,20,30,40,50}
>>> print(A)
{50, 20, 40, 10, 30}
>>> A.discard(10)
>>> print(A)
{50, 20, 40, 30}
>>> A.discard(20)
>>> print(A)
{50, 40, 30}
>>> A.discard(10)
```

pop()
Remove and return an arbitrary element from the set. Raises KeyError if the set is empty.

```
>>> A={10,20,30,40,50,60,70,80,90,100}
>>> print(A)
{100, 70, 40, 10, 80, 50, 20, 90, 60, 30}
>>> value1=A.pop()
>>> print(value1)
100
>>> value2=A.pop()
>>> print(value2)
70
>>> print(A)
{40, 10, 80, 50, 20, 90, 60, 30}
>>>
```

clear()
Remove all elements from the set.

```
>>> A=set(range(10,110,10))
>>> print(A)
{100, 70, 40, 10, 80, 50, 20, 90, 60, 30}
>>> A.clear()
>>> print(A)
set()
```

https://www.hackerrank.com/challenges/py-set-discard-remove-pop/problem?isFullScreen=false

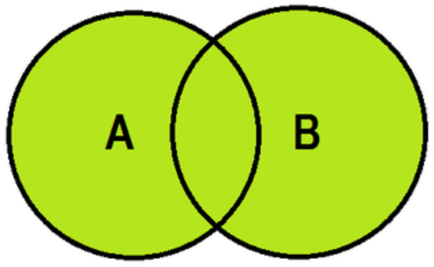
```
n=int(input())
A=set(map(int,input().split()))
N=int(input())
for i in range(N):
    cmd=input().split()
    if cmd[0]=="pop":
        A.pop()
    elif cmd[0]=="remove":
        A.remove(int(cmd[1]))
    elif cmd[0]=="discard":
        A.discard(int(cmd[1]))


print(sum(A))
```

Set Operations

union(**others*)

set | other | ...



A.union(B) or A|B

Return a new set with elements from the set and all others.

Example:

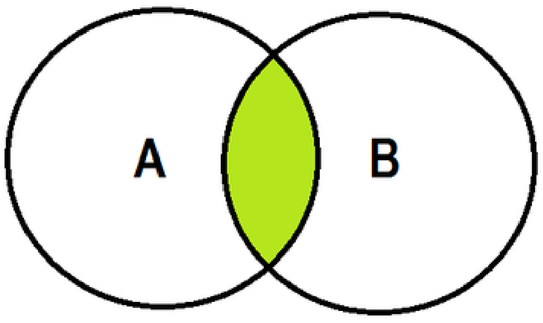
```
>>> A={1,2,3,4,5}
>>> B={4,5,6,7,8,9,10}
>>> C=A.union(B)
>>> print(A)
{1, 2, 3, 4, 5}
>>> print(B)
{4, 5, 6, 7, 8, 9, 10}
>>> print(C)
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
>>> A={1,2,3}
>>> B={3,4,5,6}
>>> C={4,5,6,7,8,9}
>>> D=A|B|C
>>> print(A)
{1, 2, 3}
>>> print(B)
{3, 4, 5, 6}
>>> print(C)
{4, 5, 6, 7, 8, 9}
>>> print(D)
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

https://www.hackerrank.com/challenges/py-set-union/problem?isFullScreen=false

```
n=int(input())
```

```
E=set(map(int,input().split()))
b=int(input())
F=set(map(int,input().split()))
EF=E.union(F)
print(len(EF))
```

intersection(others*)**
set & other & ...
Return a new set with elements common to the set and all others.



A.intersection(B) or A&B

Example:

```
>>> A={1,2,3,4,5}
>>> B={1,3,5,6,7,8}
>>> C=A.intersection(B)
>>> print(A)
{1, 2, 3, 4, 5}
>>> print(B)
{1, 3, 5, 6, 7, 8}
>>> print(C)
{1, 3, 5}
>>> D=A&B
>>> print(D)
{1, 3, 5}
```