difference(*others*)
set - other - ...
Return a new set with elements in the set that are not in the others.

```
>>> A={1,2,3,4,5}
>>> B={1,2,3,6,7,8}
>>> C=A.difference(B)
>>> print(A,B,C,sep="\n")
{1, 2, 3, 4, 5}
{1, 2, 3, 6, 7, 8}
{4, 5}
>>> X={1,2,3,4,5}
>>> Y={1,2,3,7,8,9}
>>> Z=X-Y
print(X,Y,Z,sep="\n")
{1, 2, 3, 4, 5}
{1, 2, 3, 7, 8, 9}
{4, 5}
>>> python_students={"naresh","suresh","ramesh","kishore"}
>>> java_students={"kishore","rajesh","kiran","naresh"}
>>> only_python=python_students.difference(java_students)
>>> print(python_students)
{'naresh', 'suresh', 'ramesh', 'kishore'}
>>> print(java_students)
{'naresh', 'rajesh', 'kiran', 'kishore'}
>>> print(only_python)
{'suresh', 'ramesh'}
>>> only_java=java_students-python_students
>>> print(only_java)
{'rajesh', 'kiran'}
```

symmetric_difference(*other*)
set ^ other
Return a new set with elements in either the set or *other* but not both.

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

```
{1, 2, 3, 4, 5}
>>> print(B)
{1, 2, 3, 6, 7}
>>> print(C)
{4, 5, 6, 7}
```

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

```
m=int(input())
E=set(map(int,input().split()))
n=int(input())
F=set(map(int,input().split()))
a=E.symmetric_difference(F)
print(len(a))
```

Example:

```
# Login/Logout Application

users=set()

while True:
    print("1. Login")
    print("2. Logout")
    print("3. View Users List")
    print("4. Exit")
    opt=int(input("Enter Your Option "))
    if opt==1:
        username=input("UserName ")
        password=input("Password ")
        users.add((username,password))
        print("Login Completed")
    elif opt==2:
        username=input("UserName ")
        password=input("Password ")
        users.remove((username,password))
        print("User Logout")
    elif opt==3:
        for a in users:
            print(a)
    elif opt==4:
        break
    else:
        print("Invalid Option")
```

Output

```
Enter Your Option 2
UserName nit
Password nit321
User Logout
1. Login
2. Logout
3. View Users List
4. Exit
Enter Your Option 3
1. Login
2. Logout
3. View Users List
4. Exit
Enter Your Option 4
```

Example:

```
# Develop A email List, where duplicate email-id's
# are not allowed

n=int(input("How many emailid's"))
email=set()

for i in range(n):
    emailid=input("EmailID :")
    email.add(emailid)


print(email)
```

Mutable set operations

update(**others*)
set |= other | ...

Update the set, adding elements from all others.

```
>>> A={1,2,3,4,5}
>>> B={1,2,3,5,6,7,8}
>>> C=A.union(B)
>>> print(A)
{1, 2, 3, 4, 5}
>>> print(B)
{1, 2, 3, 5, 6, 7, 8}
>>> print(C)
{1, 2, 3, 4, 5, 6, 7, 8}
>>> A.update(B)
>>> print(A)
{1, 2, 3, 4, 5, 6, 7, 8}
A=[1,2,3]
B=[5,6,7]
C={10,20,30}
C.update(A,B)
>>> print(C)
{1, 2, 3, 5, 6, 7, 10, 20, 30}
>>> S1={1,2,3}
>>> A=[10,20,30]
>>> S1|=A
Traceback (most recent call last):
  File "<pyshell#42>", line 1, in <module>
    S1|=A
TypeError: unsupported operand type(s) for |=: 'set' and 'list'
```

intersection_update(*others*)
set &= other & ...
Update the set, keeping only elements found in it and all others.

```
>>> students={"naresh","ramesh","kishore","rajesh","kiran"}
>>> students_present={"naresh","kishore","rajesh"}
>>> students.intersection_update(students_present)
>>> print(students)
{'naresh', 'rajesh', 'kishore'}
>>> S1={1,2,3,4,5}
>>> S2={1,2,3}
>>> S1&=S2
>>> print(S1)
{1, 2, 3}
>>> print(S2)
{1, 2, 3}
>>> deptno10={"naresh","ramesh","kishore","rajesh"}
>>> deptno20={"kiran","ramesh","rajesh","rakesh"}
>>> deptno1020=deptno10.intersection(deptno20)
>>> print(deptno10)
{'naresh', 'rajesh', 'kishore', 'ramesh'}
>>> print(deptno20)
{'kiran', 'rajesh', 'ramesh', 'rakesh'}
>>> print(deptno1020)
{'rajesh', 'ramesh'}
```

difference_update(*others*)
set -= other | ...
Update the set, removing elements found in others.

```
>>> A={1,2,3,4,5}
>>> B=[1,2,4,5]
>>> A.difference_update(B)
>>> print(A)
{3}
>>> S1={1,2,3,4,5,6}
>>> S2={1,2,6,7,8,9}
>>> S1-=S2
>>> print(S1)
{3, 4, 5}
```

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

```
>>> A={10,20,30}
>>> B={10,20,40}
>>> A.symmetric_difference_update(B)
>>> print(A)
{40, 30}
```