TOTAL Python

sets

Sets are another type of data structures. They differ from lists, tuples, and dictionaries because they are a mutable collection of immutable elements, unordered, and without repeated data.

mutable ✓ ordered ✗ allow duplicates ✗

```
my_set_a = {1, 2, "three"}   my_set_b = {3, "three"}
```

methods

add(item) add an element to the set

```
my_set_a.add(5)
print(my_set_a)
>> {1, 2, "three", 5}
```

clear( ) remove all elements from a set

```
my_set_a.clear()
print(my_set_a)
>> set()
```

copy( ) return a copy of the set

```
my_set_c = my_set_a.copy()
print(my_set_c)
>> {1, 2, "three"}
```

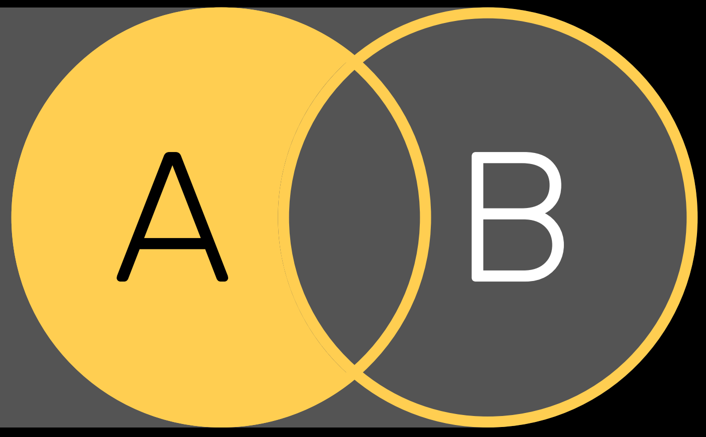
TOTAL Python

sets

```
my_set_a = {1, 2, "three"}   my_set_b = {3, "three"}
```

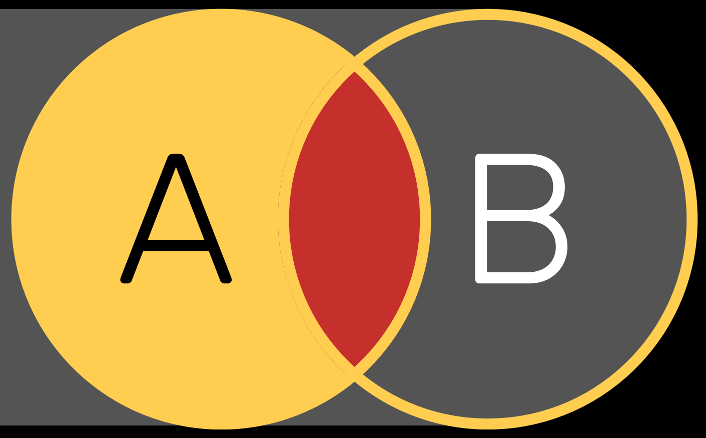
difference(set) returns the set formed by all the elements that only exist in set A

```
my_set_c = my_set_a.difference(my_set_b)
print(my_set_c)
>> {1, 2}
```



difference_update(set) removes from A all elements common to A and B

```
my_set_a.difference_update(my_set_b)
print(my_set_a)
>> {1, 2}
```

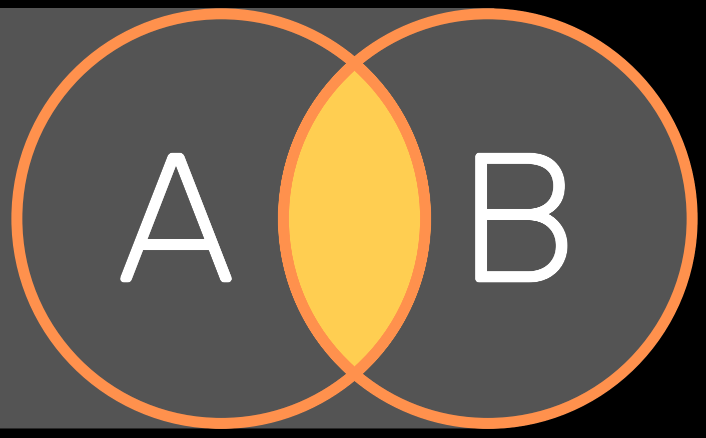


discard(item) remove an element from the set

```
my_set_a.discard("three")
print(my_set_a)
>> {1, 2}
```

intersection(set) returns the set formed by all the elements that exist in A and B simultaneously.

```
my_set_c = my_set_a.intersection(my_set_b)
print(my_set_c)
>> {'three'}
```



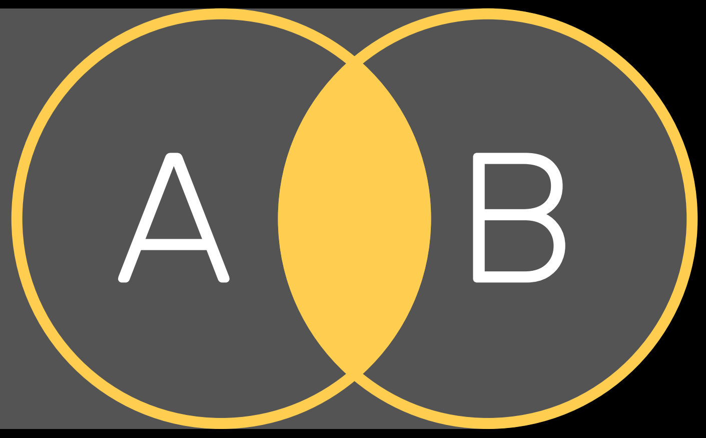
TOTAL Python

sets

```
my_set_a = {1, 2, "three"}   my_set_b = {3, "three"}
```

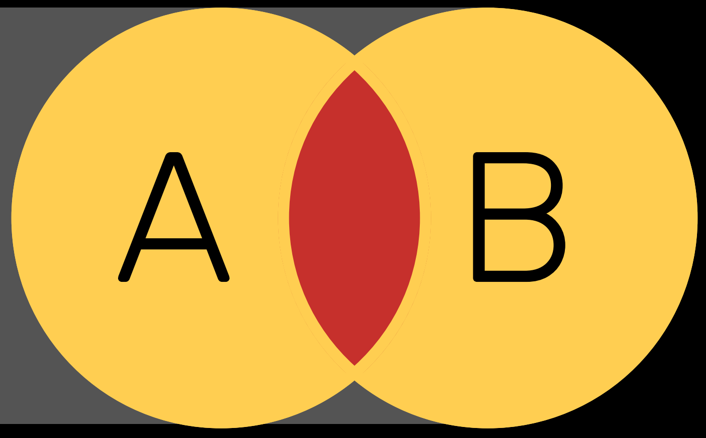
intersection_update(set) keeps only the elements common to A and B

```
my_set_b.intersection_update(my_set_a)
print(my_set_b)
>> {"three"}
```



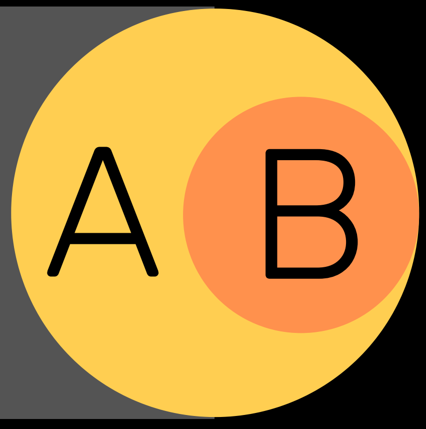
isdisjoint(set) returns True if A and B have no elements in common

```
disjoint_set = my_set_a.isdisjoint(my_set_b)
print(disjoint_set)
>> False
```



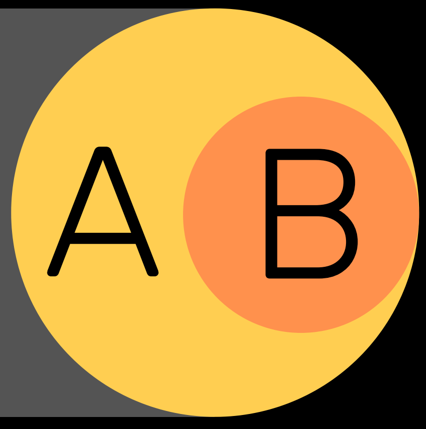
issubset(set) returns True if all elements of B are present in A

```
a_subset = my_set_b.issubset(my_set_a)
print(a_subset)
>> False
```



issuperset(set) returns True if A contains all elements of B

```
a_superset = my_set_a.issuperset(my_set_b)
print(a_superset)
>> False
```



sets

```
my_set_a = {1, 2, "three"}   my_set_b = {3, "three"}
```

pop( ) removes and returns a random element from the set

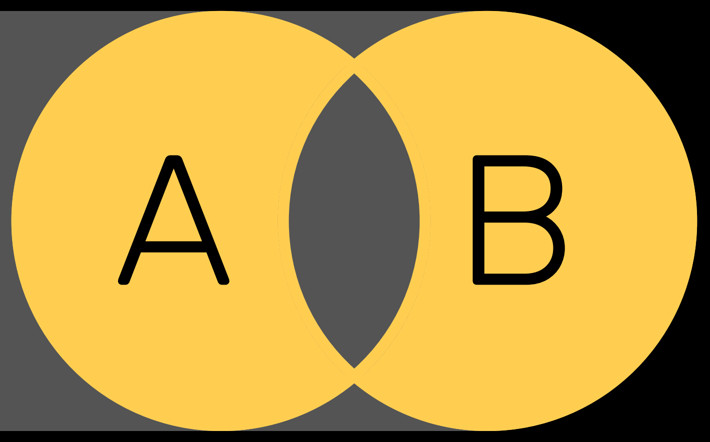
```
random = my_set_a.pop()
print(random)
>> {2}
```

remove(item) removes an item, or throws an error if it doesn't exist

```
my_set_a.remove("three")
print(my_set_a)
>> {1, 2}
```

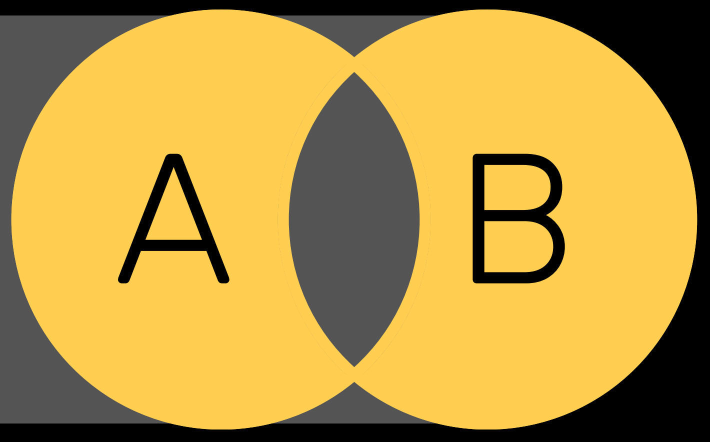
symmetric_difference(set) returns all elements of A and B, except those that are common to both

```
my_set_c = my_set_b.symmetric_difference(my_set_a)
print(my_set_c)
>> {1, 2, 3}
```



symmetric_difference_update(set) removes the elements common to A and B, keeping those that are not present in both at the same time

```
my_set_b.symmetric_difference_update(my_set_a)
print(my_set_b)
>> {1, 2, 3}
```



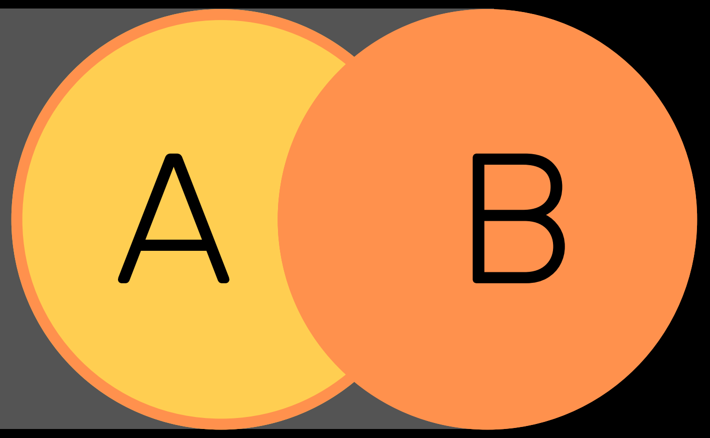
sets

TOTAL Python

```
my_set_a = {1, 2, "three"}   my_set_b = {3, "three"}
```

union(set) returns a set result of combining A and B (duplicates are removed)

```
my_set_c = my_set_a.union(my_set_b)
print(my_set_c)
>> {1, 2, 3, 'three'}
```



update(set) insert into A the elements of B

```
my_set_a.update(my_set_b)
print(my_set_a)
>> {1, 2, 3, 'three'}
```