Sets

- **set items are enclosed in curly braces**
- **set items is not ordered and not indexed**
- **set indexing and slicing can't be done**
- **set has only immutable data types (numbers, string, tuples) lists and dictionary are not**
- **set items can't be repeat, if repeat will ignore the repeat**

```
mySet = {1, 2, 3}
print(mySet)    # {1, 2, 3}
```

Sets Methods

- **clear()**
  - **used to remove all elements from the set**

```
admins = {"osama", "khaled"}
admins.clear()
print(admins)       # # set()
```

- **union()**
  - **used to union more than one set**

```
admins = {"osama", "khaled"}
users = {"mona", 'asmaa'}
print(admins.union(users))      # {'mona', 'osama', 'khaled', 'asmaa'}
```

- **add()**
  - **used to add new element to the set**
  - **that take one argument not more**

```
numbers = {1, 3, 4, 45}
numbers.add(10)
print(numbers)      # {1, 3, 4, 10, 45}
```

- **copy()**
  - **used to copy the set element to new set**

```
numbers = {1, 3, 4, 45}
nums = numbers.copy()
print(nums)     # {1, 3, 4, 45}
```

- **remove()**
  - **used to remove a specific element from the set**
  - **if the element not exist will make error**

```
numbers = {1, 3, 4, 45}
numbers.remove(3)
print(numbers)      # {1, 4, 45}
```

- **discard()**
  - **used to remove element from the set**
  - **if the element is not exist that will pass (not make error)**

```
numbers = {1, 3, 4, 45}
numbers.discard(3)
numbers.discard(130)
print(numbers)      # {1, 4, 45}
print(numbers.discard(130))     # None
```

- **pop()**
  - **used to remove random element from the set**

```
numbers = {1, 3, 4, 45}
numbers.pop()
print(numbers)      # {3, 4, 45}
```

- **update()**
  - **used to update a set with the union of itself and others**

```
numbers = {1, 3, 4, 45}
nums = {10, 1, 3, 11}
numbers.update(nums)
print(numbers)      # {1, 3, 4, 10, 11, 45}
```

```
# different()
# - used to differ between two sets
# - that print the elements that exist on first set and not exist on the second set
nums1 = {1, 2, 3, 4, 5}
nums2 = {1, 2, 6 ,7 ,8}
print(nums1)    # {1, 2, 3, 4, 5}
print(nums1.difference(nums2))  # {3, 4, 5}

# difference_update()
# - used to get the difference then update the original set with the new elements
print(nums1)    # {1, 2, 3, 4, 5}
nums1.difference_update(nums2)
print(nums1)    # {3, 4, 5}
```

```python
# intersection()
# - print the elements that exist on the first set and exist on the second set
x = {1, 2, 3, 4, 5, 6}
y = {1, 2, 3, 66, 77, 88}
print(x.intersection(y))        # {1, 2, 3}

# intersection_update()
# - print the elements that exist on the first, second set
# - after print elements update the original set with the new elements
print(x)    # {1, 2, 3, 4, 5, 6}
x.intersection_update(y)
print(x)        # {1, 2, 3}

# symmetric_difference()
# - get the elements that not exists on the first set and second set
set1 = {1, 2, 3, 4}
set2 = {1, 2, 22, 55}
print(set1.symmetric_difference(set2))   # {3, 4, 22, 55}
print(set2.symmetric_difference(set1))   # {3, 4, 22, 55}

# symmetric_difference_update()
# - get the elements that not exists on the first set and second set
# - then update the original set with the elements
print(set1)     # {1, 2, 3, 4}
set1.symmetric_difference_update(set2)
print(set1)     # {3, 4, 22, 55}

# issuperset()
# - check if the second set exists on the first set
mySet1 = {1, 2, 3, 4, 5}
mySet2 = {1, 2, 3}
print(mySet1.issuperset(mySet2))    # True
print(mySet2.issuperset(mySet1))    # False

# issubset()
# - check if the  elements in the first set is exist on the second set
print(mySet1.issubset(mySet2))    # False
print(mySet2.issubset(mySet1))    # True

# isdisjoint()
# - check if the elements on both sets are disjoint
# - means first set contains elements that does not exist on the second set
# return False if the first set contain elements that exist on second set
a = {1, 2, 3}
b = {4, 5, 6}
print(a.isdisjoint(b))      # True
```