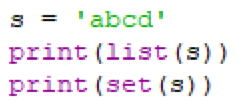
Sets in Python

A set is a group of elements that are not in any specific sequence. A Python set is comparable to this mathematical description, but with the additional requirements listed below.

- The set's elements cannot be duplicated.
- The set's elements are immutable (cannot be changed), but the set as a whole is mutable.
- A python set has no indexes associated with its elements. As a result, no indexing or slicing operations are supported.

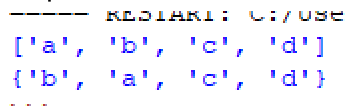
Defining a Set

A set can be created in two ways. First, you can define a set with the built-in set() function:
x = set(<iter>)



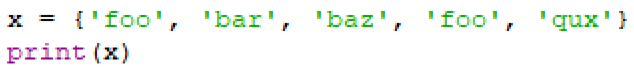
```
s = 'abcd'
print(list(s))
print(set(s))
```

Output:



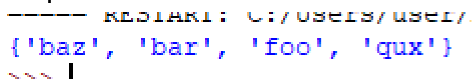
```
----- RESTART: C:/Use
['a', 'b', 'c', 'd']
{'b', 'a', 'c', 'd'}
>>>
```

Alternately, a set can be defined with curly braces ({}):
x = {<obj>, <obj>, ..., <obj>}



```
x = {'foo', 'bar', 'baz', 'foo', 'qux'}
print(x)
```

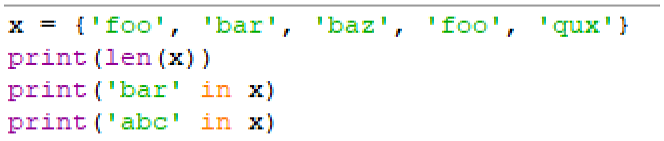
Output:



```
----- RESTART: C:/Users/user/
{'baz', 'bar', 'foo', 'qux'}
>>>
```

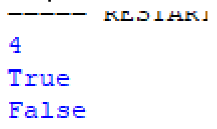
Set Size and Membership

The len() function returns the number of elements in a set, and the in and not in operators can



```
x = {'foo', 'bar', 'baz', 'foo', 'qux'}
print(len(x))
print('bar' in x)
print('abc' in x)
```

Output:



```
----- RESTART
4
True
False
```

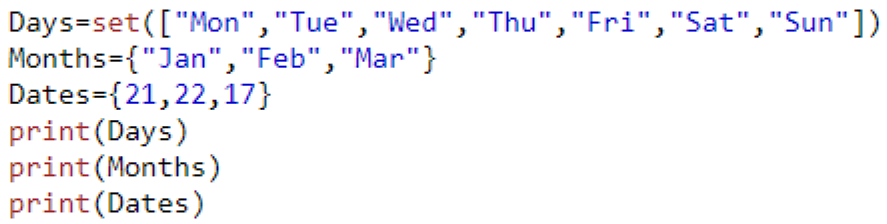
Set Operations

Python sets are commonly used for mathematical operations such as union, intersection, difference, and complement, among others. We can make a set, access its elements, and perform the following mathematical operations, as indicated below.

Creating a set

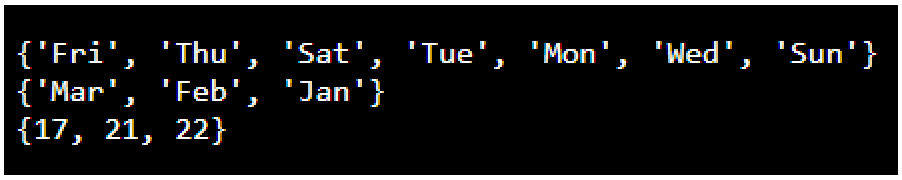
A set is formed by using the set() function or by enclosing all elements in a pair of curly braces.

Ex



```
Days=set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])
Months={"Jan","Feb","Mar"}
Dates={21,22,17}
print(Days)
print(Months)
print(Dates)
```

Output:
When the above code is run, it yields the following result. Please take note of how the order of the elements has altered in the final output.



```
{'Fri', 'Thu', 'Sat', 'Tue', 'Mon', 'Wed', 'Sun'}
{'Mar', 'Feb', 'Jan'}
{17, 21, 22}
```

Getting Values from a Set

Individual values in a set cannot be accessed. We can only access all of the elements at the same time, as illustrated above. However, by looping through the set, we may obtain a list of individual elements.
Example

```
Days=set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])

for d in Days:
   print(d)
```

Output:
When the above code is run, it yields the following result:

```
Sat
Mon
Wed
Tue
Thu
Fri
Sun
```

Adding Items to a Set

The add() method is used to add elements to a set. As previously stated, there is no specific index associated with the newly inserted element.
Ex:

```
Days=set(["Mon","Tue","Wed","Thu","Fri","Sat"])
Days.add("Sun")
print(Days)
```

Output:
When the above code is run, it yields the following result:

```
{'Sun', 'Thu', 'Mon', 'Sat', 'Fri', 'Tue', 'Wed'}
```

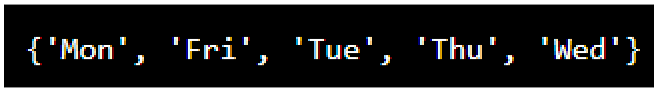
Removing Item from a Set

Using the discard() method, we can remove elements from a set. As previously stated, there is no specific index associated with the newly inserted element.

Ex:

```
Days=set(["Mon","Tue","Wed","Thu","Fri","Sat"])
Days.discard("Sat")
print(Days)
```

Output:
When the above code is run, it yields the following result.



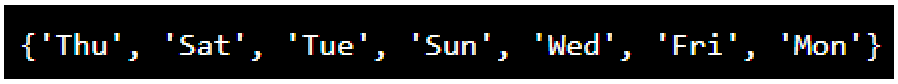
```
{'Mon', 'Fri', 'Tue', 'Thu', 'Wed'}
```

Union of Sets

The union operation on two sets yields a new set that contains all of the different elements from both sets. In the following example, the element "Wed" appears in both collections.
Ex:

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Wed","Thu","Fri","Sat","Sun"])
AllDays=DaysA|DaysB
print(AllDays)
```

Output:
When the above code is run, it yields the following result. Please keep in mind that the outcome contains only one "wed."



```
{'Thu', 'Sat', 'Tue', 'Sun', 'Wed', 'Fri', 'Mon'}
```

Intersection of Sets

When two sets are intersected, a new set is created that contains just the elements that are shared by both sets. In the following example, the element "Wed" appears in both collections.
Ex:

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set (["Wed","Thu","Fri","Sat","Sun"])
AllDays = DaysA & DaysB
print(AllDays)
```

Output:
When the above code is run, it yields the following result. Please keep in mind that the outcome contains only one "wed."

```
{'Wed'}
```

Difference of Sets

When you perform a difference operation on two sets, you get a new set that has only the items from the first set and none from the second. The element "Wed" is present in both sets in the example below, hence it will not be found in the result set.
Ex:

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Wed","Thu","Fri","Sat","Sun"])
AllDays = DaysA - DaysB
print(AllDays)
```

Output:
When the above code is run, it yields the following result. Please keep in mind that the outcome contains only one "wed."

```
{'Mon', 'Tue'}
```

Compare Sets

We can determine whether a given set is a subset or superset of another. Depending on the elements in the sets, the result is True or False.
Ex:

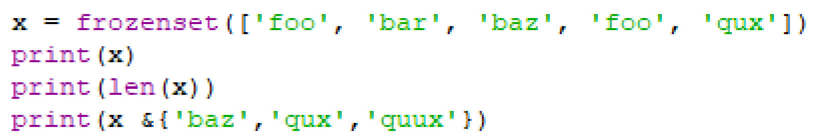
```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])
SubsetRes= DaysA <= DaysB
SupersetRes=DaysB >= DaysA
print(SubsetRes)
print(SupersetRes)
```

Output:
When the above code is run, it yields the following result:

```
True
True
```

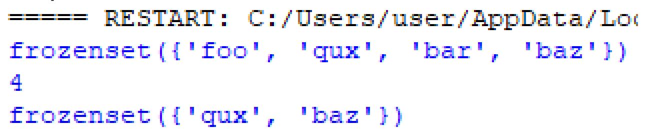
Frozen Sets

Python has an additional built-in type called a frozenset, which is identical to a set except that it is immutable. On a frozenset, you can do the following non-modifying operations:



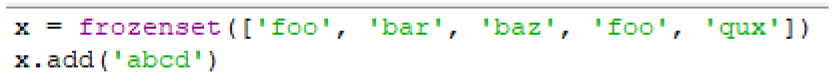
```python
x = frozenset(['foo', 'bar', 'baz', 'foo', 'qux'])
print(x)
print(len(x))
print(x &{'baz','qux','quux'})
```

Output:



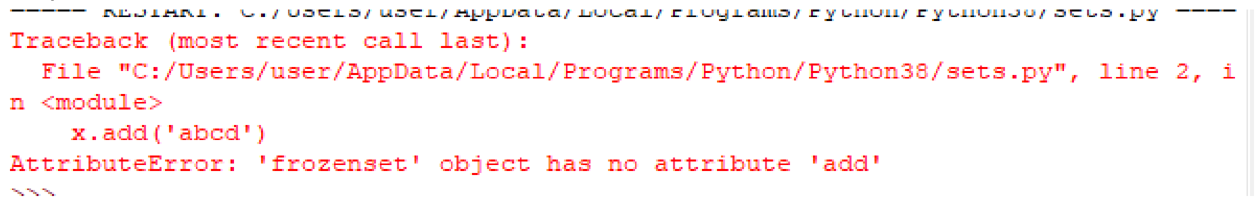
```
===== RESTART: C:/Users/user/AppData/Lo
frozenset({'foo', 'qux', 'bar', 'baz'})
4
frozenset({'qux', 'baz'})
```

Methods that attempt to alter a frozenset, on the other hand, fail:



```python
x = frozenset(['foo', 'bar', 'baz', 'foo', 'qux'])
x.add('abcd')
```

Output:



~~~~
===== RESTART: C:/Users/user/AppData/Local/Programs/Python/Python38/sets.py ====
Traceback (most recent call last):
  File "C:/Users/user/AppData/Local/Programs/Python/Python38/sets.py", line 2, i
n <module>
    x.add('abcd')
AttributeError: 'frozenset' object has no attribute 'add'
```
~~~~