PYTHON SETS:

myset = {"apple", "banana", "cherry"}
Set
Sets are used to store multiple items in a single variable.

Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Tuple, and Dictionary, all with different qualities and usage.

A set is a collection which is unordered, unchangeable*, and unindexed.

* Note: Set items are unchangeable, but you can remove items and add new items.

Sets are written with curly brackets.

ExampleGet your own Python Server
Create a Set:

thisset = {"apple", "banana", "cherry"}
print(thisset)
Note: Sets are unordered, so you cannot be sure in which order the items will appear.

Set Items
Set items are unordered, unchangeable, and do not allow duplicate values.

Unordered
Unordered means that the items in a set do not have a defined order.

Set items can appear in a different order every time you use them, and cannot be referred to by index or key.

Unchangeable
Set items are unchangeable, meaning that we cannot change the items after the set has been created.

Once a set is created, you cannot change its items, but you can remove items and add new items.

Duplicates Not Allowed
Sets cannot have two items with the same value.

Example
Duplicate values will be ignored:

```
thisset = {"apple", "banana", "cherry", "apple"}

print(thisset)
```

Note: The values True and 1 are considered the same value in sets, and are treated as duplicates:

Example
True and 1 is considered the same value:

```
thisset = {"apple", "banana", "cherry", True, 1, 2}

print(thisset)
```

Get the Length of a Set
To determine how many items a set has, use the len() function.

Example
Get the number of items in a set:

```
thisset = {"apple", "banana", "cherry"}

print(len(thisset))
```

Set Items - Data Types
Set items can be of any data type:

Example
String, int and boolean data types:

```
set1 = {"apple", "banana", "cherry"}
set2 = {1, 5, 7, 9, 3}
set3 = {True, False, False}
```

A set can contain different data types:

Example
A set with strings, integers and boolean values:

```
set1 = {"abc", 34, True, 40, "male"}
```

type()
From Python's perspective, sets are defined as objects with the data type 'set':

<class 'set'>
Example
What is the data type of a set?

```
myset = {"apple", "banana", "cherry"}
print(type(myset))
```

The set() Constructor

It is also possible to use the set() constructor to make a set.

Example

Using the set() constructor to make a set:

```
thisset = set(("apple", "banana", "cherry")) # note the double round-brackets
print(thisset)
```

SET FUNCTIONS:

1)Python Set add() Method

ExampleGet your own Python Server

Add an element to the fruits set:

```
fruits = {"apple", "banana", "cherry"}

fruits.add("orange")

print(fruits)
```

Definition and Usage

The add() method adds an element to the set.

If the element already exists, the add() method does not add the element.

2)Python Set clear() Method

ExampleGet your own Python Server

Remove all elements from the fruits set:

```
fruits = {"apple", "banana", "cherry"}

fruits.clear()

print(fruits)
```

Definition and Usage

The clear() method removes all elements in a set.

3)Python Set copy() Method

ExampleGet your own Python Server
Copy the fruits set:

```
fruits = {"apple", "banana", "cherry"}

x = fruits.copy()

print(x)
```

Definition and Usage
The copy() method copies the set.

4)Python Set difference() Method

ExampleGet your own Python Server
Return a set that contains the items that only exist in set x, and not in set y:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.difference(y)

print(z)
```

Definition and Usage
The difference() method returns a set that contains the difference between two sets.

Meaning: The returned set contains items that exist only in the first set, and not in both sets.

5)Python Set discard() Method

Remove "banana" from the set:

```
fruits = {"apple", "banana", "cherry"}

fruits.discard("banana")

print(fruits)
```

Definition and Usage
The discard() method removes the specified item from the set.

This method is different from the remove() method, because the remove() method will raise an error if the specified item does not exist, and the discard() method will not.

6)Python Set intersection() Method

Return a set that contains the items that exist in both set x, and set y:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.intersection(y)

print(z)
```

Definition and Usage
The intersection() method returns a set that contains the similarity between two or more sets.

Meaning: The returned set contains only items that exist in both sets, or in all sets if the comparison is done with more than two sets.

7)Python Set isdisjoint() Method

ExampleGet your own Python Server
Return True if no items in set x is present in set y:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "facebook"}

z = x.isdisjoint(y)

print(z)
```

Definition and Usage
The isdisjoint() method returns True if none of the items are present in both sets, otherwise it returns False.

8)Python Set issubset() Method

ExampleGet your own Python Server
Return True if all items in set x are present in set y:

```
x = {"a", "b", "c"}
y = {"f", "e", "d", "c", "b", "a"}

z = x.issubset(y)

print(z)
```

Definition and Usage

The issubset() method returns True if all items in the set exists in the specified set, otherwise it returns False.

9)Python Set pop() Method

ExampleGet your own Python Server
Remove a random item from the set:

```
fruits = {"apple", "banana", "cherry"}

fruits.pop()

print(fruits)
```

Definition and Usage
The pop() method removes a random item from the set.

This method returns the removed item.

10)Python Set remove() Method

ExampleGet your own Python Server
Remove "banana" from the set:

```
fruits = {"apple", "banana", "cherry"}

fruits.remove("banana")

print(fruits)
```

Definition and Usage
The remove() method removes the specified element from the set.

This method is different from the discard() method, because the remove() method will raise an error if the specified
item does not exist, and the discard() method will not.

11)Python Set union() Method

ExampleGet your own Python Server
Return a set that contains all items from both sets, duplicates are excluded:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.union(y)
```

```
print(z)
```

12)Python Set update() Method

Insert the items from set y into set x:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

x.update(y)

print(x)
```