Python Set and Dictionary

BY ROHIT S. AGRAWAL

Python Set

- A set is an unordered collection of items.
- Each element in the set must be unique, immutable, and the sets remove the duplicate elements.
- However, a set itself is mutable. We can add or remove items from it.
- Sets can also be used to perform mathematical set operations like union, intersection, symmetric difference, etc.
- There is no index attached to the elements of the set.
- However, we can print them all together, or we can get the list of elements by looping through the set.

Creating a set

- A set is created by placing all the items (elements) inside curly braces ({}), separated by comma, or by using the built-in ‘set()’ function.
- It can have any number of items and they may be of different types (integer, float, tuple, string etc.).
- But a set cannot have mutable elements like lists, sets or dictionaries as its elements.
- Try to create a set of integers.
- Create a set of days and print it.
- Create a set of mixed types.
- Create a set using set() method

Lets create set…

Create an empty set.

Create a set from given list.

Create a set with mutable objects(list, set).

Create a set with duplicate values.

Modifying a set

- Sets are mutable. However, since they are unordered, indexing has no meaning.
- We cannot access or change an element of a set using indexing or slicing. Set data type does not support it.
- We can add a single element using the add() method, and multiple elements using the update() method.
- The update() method can take tuples, lists, strings or other sets as its argument.
- In all cases, duplicates are avoided.

Playing with set

Create a set()

Add elements to the set using add().

Try to add multiple elements to the set().

Try to add the element at index 4.

Try to add a list to the set.

Try to add a set to the set.

Removing elements

- A particular item can be removed from a set using the methods discard() and remove().
- The only difference between the two is that the discard() function leaves a set unchanged if the element is not present in the set.
- On the other hand, the remove() function will raise an error in such a condition (if element is not present in the set).

Removing elements..

- Create a set().
- Add the element to the set.
- Discard an element from the set.
- Discard an element not present in the set.
- Remove an element from the set.
- Remove an element not present in the set.
- Try to remove element using the pop() method.

Removing elements..

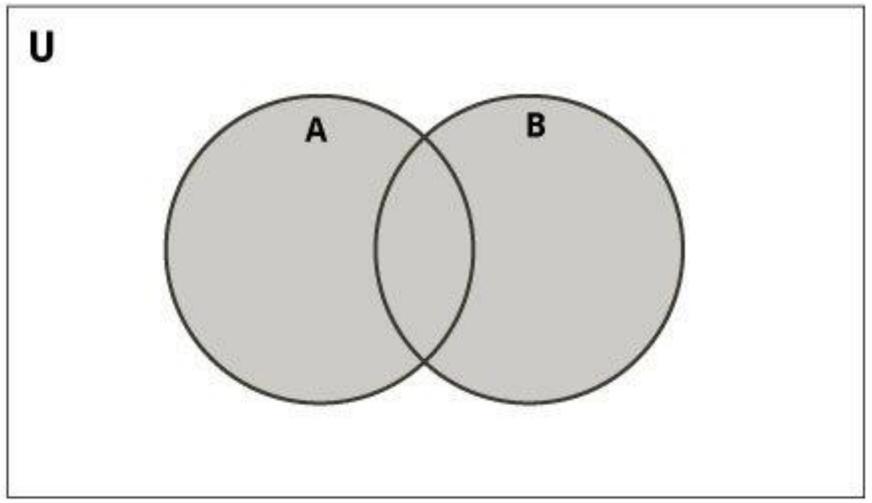
- we can remove and return an item using the pop() method.
- Since set is an unordered data type, there is no way of determining which item will be popped. It is completely arbitrary.
- We can also remove all the items from a set using the clear() method.
- Try to remove all the elements from the set().

Set Operations

- Sets can be used to carry out mathematical set operations like union, intersection, difference and symmetric difference.
- We can do this with operators or methods.
- Python provides different method to achieve the set operations.

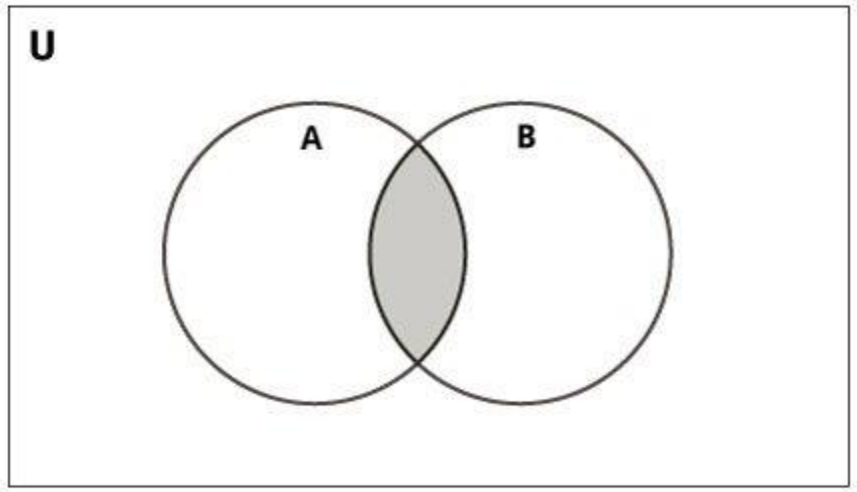
Union operation

- The union of two sets is calculated by using the pipe (|) operator.
- The union of the two sets contains all the items that are present in both the sets.
- Same can be accomplished using the union() method.
- Create two different sets A and B
- Perform the union operation on these sets.



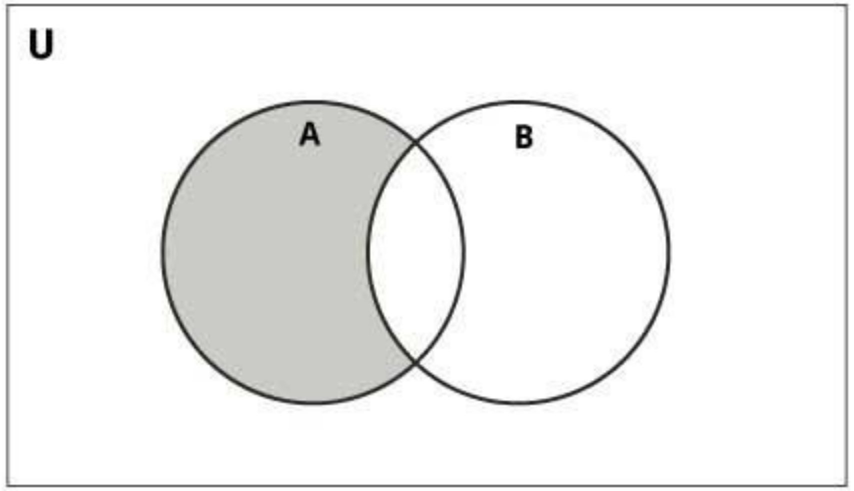
Set Intersection

- Intersection of A and B is a set of elements that are common in both the sets.
- Intersection is performed using '&' operator.
- Same can be accomplished using the intersection() method.
- Create two sets.
- Try to create intersection in set A and B.
- Try using the method.



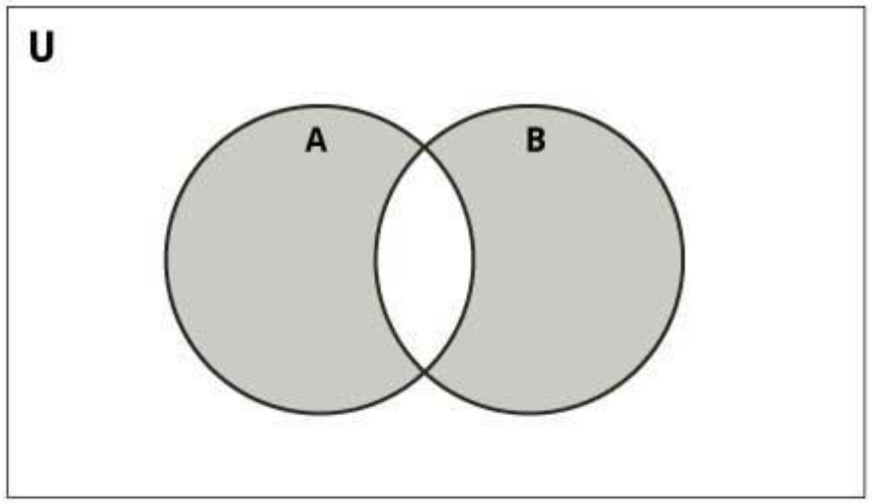
Set Difference

- Difference of the set B from set A(A-B) is a set of elements that are only in A but not in B.
- Similarly, B-A is a set of elements in B but not in A.
- Difference is performed using '–' operator and also by using difference() method.
- Create two sets.
- Try to print difference in set A and B.
- Try using the method.



Set Symmetric Difference

- Symmetric Difference of A and B is a set of elements in A and B but not in both.
- Symmetric difference is performed using ‘^’ operator.
- Same can be accomplished using the method symmetric_difference() method.
- Create two sets.
- Try to print symmetric difference in set A and B.
- Try using the method.



Methods with set

- Copy() :Returns a copy of the set
- Difference_update() :Removes all elements of another set from this set
- Isdisjoint() : Returns True if two sets have a null intersection.
- Issubset : Returns True if another set contains this set.

Frozenset

- Frozenset is a new class that has the characteristics of a set, but its elements cannot be changed once assigned.
- While tuples are immutable lists, frozensets are immutable sets.
- Sets being mutable are unhashable, so they can't be used as dictionary keys.
- frozensets are hashable and can be used as keys to a dictionary.
- Frozensets can be created using the frozenset() function.
- Being immutable, it does not have methods that add or remove elements.

Python Dictionary

- Python dictionary is an unordered collection of items.
- Each item of a dictionary has a key/value pair.
- Dictionaries are optimized to retrieve values when the key is known.
- Creating a dictionary is as simple as placing items inside curly braces{} separated by commas.
- An item has a key and a corresponding value that is expressed as a pair (**key: value**).
- While the values can be of any data type and can repeat, keys must be of immutable type (string, number or tuple with immutable elements) and must be unique.

Creating a Dictionary

- Create an empty dictionary
- Create a dictionary of numbers
- Create a dictionary of mixed types
- Create a dictionary of using dict() function
- Create a dictionary from a sequence having each item as a pair.

Accessing Elements

- While indexing is used with other data types to access values, a dictionary uses keys.
- Keys can be used either inside square brackets [] or with the get() method.
- If we use the square brackets[], KeyError is raised in case a key is not found in the dictionary.
- On the other hand, the get() method returns None if the key is not found.
- Create a dictionary
- Try to access elements using [].
- Try to access elements using get() method.
- Try to access elements which does not exists.

Changing and Adding elements

- Dictionaries are mutable. We can add new items or change the value of existing items using an assignment operator.
- If the key is already present, then the existing value gets updated.
- n case the key is not present, a new (**key: value**) pair is added to the dictionary.
- Get a dictionary
- Add elements
- Update an element

Removing elements

- We can remove a particular item in a dictionary by using the pop() Method.
- This method removes an item with the provided key and returns the value.
- The popitem() method can be used to remove and return an arbitrary (key-value) item pair from the dictionary.
- All the items can be removed at once, using the clear() method.
- We can also use the del keyword to remove individual items or the entire dictionary itself.

Dictionary Comprehension

- Dictionary comprehension is an elegant and concise way to create a new dictionary from an iterable in Python.
- Dictionary comprehension consists of an expression pair (**key: value**) followed by a ‘for’ statement inside curly braces({}).
- Squares = {a: a*a for a in range(6)}
- A dictionary comprehension can optionally contain more for or if statements.
- An optional if statement can filter out items to form the new dictionary.
- Print the squares of odd numbers