Sets

- Unordered collection of distinct elements
- Can contain elements of different types.
- Sets are not hashable
- Can only contain hashable items.
- Enclosed inside curly braces and separated by symbol comma(,)
- basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
- print(basket) # duplicates will not be printed
- s=set() # empty set or null set declaration

Basic set operations methods

- A={2,34,12,56,52,9}
- B={7,15,34,23,52,3}
- A.intersection(B) # Will return Intersection of sets A and B as a new set
- A & B # Will return Intersection of sets A and B as a new set
- A.intersection_update(B) # set A is updated with the intersection of sets A and B
- A.union(B) # Will return union of sets as a new set
- A| B # Will return union of sets as a new set
- A.update(B) # set A is updated with the union of sets A and B

Basic set operations methods

- A.difference(B) # Return difference of two sets A and B as a new set
- print(A – B) # Return difference of two sets A and B as a new set
- A.difference_update(B) # set A is updated with the difference of sets A and B
- A.symmetric_difference(B)) #Return a symmetric difference of two sets A and B as a new set
- A^B # Return a symmetric difference of two sets A and B as a new set
- A.symmetric_difference_update(B) # set A is updated with the symmetric difference of sets A and B

set methods

- C = {'apple', 'orange',23, 34.76,3+5j}
- C.add("cherry") # adding new element
- C.remove('s') # gives error if not a member
- C.discard('a') # do nothing if not a member
- C.pop() # remove and return an arbitrary element
- ----------
- A={2,34,12,56,52,9}
- B={7,15,11,23,3}
- A.isdisjoint(B) # True
- A.issubset(B) # False
- A.issuperset(B)) # False

Frozenset

- Frozenset is a new class that has characteristics of a set, but its elements cannot be changed once assigned.
- Frozensets are immutable sets.
- Frozensets are hashable and hence can be used as keys to a dictionary.
- Can be created using the function:
- frozenset()

Declaration of frozenset

- **Frozensets are hashable.**
- **A=frozenset({2,4,6,8})**
- **B=frozenset([1,3,5,7,9])**
- **C=frozenset((11,23,45,67,89))**
- **D=frozenset(*'harsh'*)**
- **A.isdisjoint(B) # prints True**
- **A.issubset(B) # False**
- **A.issuperset(B)) # False**

- fruits = {"apple", "orange", "banana", "apple", "pear", "papaya", "papaya"}
- fruit_basket = {"apple", "banana", "grapes", "mango", "kiwi"}
- **Predict the output of following statements:**
- print(fruits)
- print(fruits & fruit_basket)
- print(fruits | fruit_basket)
- print(fruits -fruit_basket)

- fruits = {"apple", "orange", "banana", "apple", "pear", "papaya", "papaya"}
- fruit_basket = {"apple", "banana", "grapes", "mango", "kiwi"}
- print(fruits ^ fruit_basket)
- print(len(fruit_basket))
- print("pear" in fruits)
- print("pear" not in fruit_basket)
- print(fruits.issubset(fruit_basket))
- print(fruits.issuperset(fruit_basket))
- print(fruit_basket.copy())

- A={2,34,12,56,52,9}
- B=A
- C=A.copy()
- print(id(A),id(B)) # 2040292757576 2040292757576
- print(id(A),id(C)) # 2040292757576 2040293137320