# ANSWERS

# PYTHON - DATA STRUCTURE

# TOPICS – Methods of Dictionary & Sets

Sets:**.**

**1) add():**

# add()\*  #Adds an element to the set

# Add an item to a set, using the add() method:

thisSet = {"apple", "banana", "cherry"}

thisSet.add("orange")

print(thisSet)  # return sets

**2) update():**

# update()\* #Update the set with the union of this set and others

# Add multiple items to a set, using the update() method:

thisSet = {"apple", "banana", "cherry"}

thisSet.update(["orange", "mango", "grapes"])

print(thisSet)

# Return the updated list {'banana', 'mango', 'apple', 'orange', 'grapes', 'cherry'}

**3).len():**

# len()  # Return count of key entities

# Get the number of items in a set:

thisSet = {"apple", "banana", "cherry"}

print(len(thisSet))  # 3

**4) remove():**

# remove()\* #Removes the specified element

# Remove "banana" by using the remove() method:

thisSet = {"apple", "banana", "cherry"}

thisSet.remove("banana")

# Return set

print(thisSet)  # {'cherry', 'apple'}

**5) discard():**

# discard() #Remove the specified item

# Remove "banana" by using the discard() method:

thisSet = {"apple", "banana", "cherry"}

thisSet.discard("banana")

# Return set

print(thisSet)  # {'cherry', 'apple'}

**6) pop():**

# pop()\*  # Removes an element from the set

# Remove the last item by using the pop() method:

thisSet = {"apple", "banana", "cherry"}

# remove the last element

x = thisSet.pop()

# Return the remove element

print(x)  # cherry

print(thisSet)  # {'banana', 'cherry'}

**7) clear():**

# clear()\*  #Removes all the elements from the set

# Return the blank set

# The clear() method empties the set:

thisSet = {"apple", "banana", "cherry"}

thisSet.clear()

print(thisSet)  # set()

**8) del:**

# delete the set

# The del keyword will delete the set completely:

thisSet = {"apple", "banana", "cherry"}

del thisSet

print(thisSet)

**9) union():**

# union() # Return a set containing the union of sets

# You can use the union() method that returns a new set containing

# all items from both sets, or the update() method that inserts

# all the items from one set into another:

set1 = {"a", "b", "c"}

set2 = {1, 2, 3}

set3 = set1.union(set2)

print(set3)  # {1, 2, 3, 'a', 'c', 'b'}

**10) copy():**

# copy()\* # Returns a copy of the set

# Copy the fruits set:

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

fruits1 = fruits.copy()

print(fruits1)

**11) difference():**

# difference()  # Returns a set containing the difference between two or more sets

# 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"}

diff1 = x.difference(y)

print(diff1)  # {'banana', 'cherry'}

diff2 = y.difference(x)

print(diff2)  # {'microsoft', 'google'}

**12) difference\_update():**

# difference\_update() # Removes the items in this set that are also included

# in another, specified set

# Remove the items that exist in both sets:

x = {"apple", "banana", "cherry"}

y = {"google", "microsoft", "apple"}

x.difference\_update(y)

print(x)  # return set # {'cherry', 'banana'}

**13) intersection():**

# intersection()  # Returns a set, that is the intersection of two other sets

# 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)  # {'apple'}

**14)intersection\_ update():**

# intersection\_update() # Removes the items in this set that are not

# present in other, specified set(s)

# Remove the items that is not present in both x, and set y:

x = {"apple", "banana", "cherry"}

y = {"google", "microsoft", "apple"}

x.intersection\_update(y)

print(x)  # return remove item  # {'apple'}

**15) isdisjoint():**

# isdisjoint()  # Returns whether two sets have a intersection or not

# 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)  # True

**16) issubset()**

# issubset()  #Returns whether another set contains this set or not

# Return True if all items set x are present in set y:

x = {"a", "b", "c"}

y = {"f", "e", "d", "c", "b", "a"}

z = x.issubset(y)

print(z)  # True

**17) issuperset():**

# issuperset()  #Returns whether this set contains another set or not

# Return True if all items set y are present in set x:

x = {"f", "e", "d", "c", "b", "a"}

y = {"a", "b", "c"}

z = x.issuperset(y)

print(z)  # True

**18) symmetric\_difference():**

# symmetric\_difference()  #Returns a set with the symmetric differences of two sets

# Return a set that contains all items from both sets,

# except items that are present in both sets:

x = {"apple", "banana", "cherry"}

y = {"google", "microsoft", "apple"}

z = x.symmetric\_difference(y)  # return set

print(z)  # return set

# {'google', 'banana', 'microsoft', 'cherry'}

**19) symmetric\_difference\_update():**

# symmetric\_difference\_update() #inserts the symmetric differences

from this set and another

# Remove the items that are present in both sets,

# AND insert the items that is not present in both sets:

x = {"apple", "banana", "cherry"}

y = {"google", "microsoft", "apple"}

x.symmetric\_difference\_update(y)

print(x)  # return set

# {'cherry', 'banana', 'google', 'microsoft'}