## Python Exercises- Sets

[https://www.w3resource.com/python-exercises/sets/](https://www.w3resource.com/python-exercises/dictionary/)

#1. Write a Python program to create a set

set1={1,2,3,4,5}

print(set1,type(set1))

#Textbook

print("Create a new set:")

x = set()

print(x)

print(type(x))

print("\nCreate a non empty set:")

n = set([0, 1, 2, 3, 4])

print(n)

print(type(n))

print("\nUsing a literal:")

a = {1,2,3,'foo','bar'}

print(type(a))

print(a)

#2. Write a Python program to iterate over sets

set1={1,2,3,3,4,5,6}

for i in set1:

print(i) #1 2 3 4 5 6

#Textbook

#Create a set

num\_set = set([0, 1, 2, 3, 4, 5])

for n in num\_set:

print(n, end=' ')

print("\n\nCreating a set using string:")

char\_set = set("w3resource")

# Iterating using for loop

for val in char\_set:

print(val, end=' ')

#3. Write a Python program to add member(s) to a set

set1=set([1,2,3,4,5])

set1.add(6)

print(set1)

#Textbook

#A new empty set

color\_set = set()

print(color\_set)

print("\nAdd single element:")

color\_set.add("Red")

print(color\_set)

print("\nAdd multiple items:")

color\_set.update(["Blue", "Green"])

print(color\_set)

#4. Write a Python program to remove item(s) from a given set

set1={1,2,3,4,5}

set1.remove(4)

print(set1)

#Textbook

num\_set = set([0, 1, 3, 4, 5])

print("Original set:")

print(num\_set)

num\_set.pop()

print("\nAfter removing the first element from the said set:")

print(num\_set) #{1, 3, 4, 5}

'''

5. Write a Python program to remove an item from a set if it is present

in the set.

'''

set1=set([1,2,3,4,5])

set1.remove(3)

print(set1)

#Textbook

#Create a new set

num\_set = set([0, 1, 2, 3, 4, 5])

print("Original set elements:")

print(num\_set)

print("\nRemove 0 from the said set:")

num\_set.discard(4)

print(num\_set)

#warmup - discard will not raise error if the element does not exist

set2={1,2,3,4,5,6}

set2.discard(7)

print(set2) #no error

#6. Write a Python program to create an intersection of sets

set1=set([1,2,3,4,5])

set2=set([4,5,6,7,8])

print(set1.intersection(set2))

#Textbook

setx = set(["green", "blue"])

sety = set(["blue", "yellow"])

print("Original set elements:")

print(setx)

print(sety)

print("\nIntersection of two said sets:")

setz = setx & sety

print(setz)

#7. Write a Python program to create a union of sets

set1=set([1,2,3,4,5])

set2=set([1,3,5,7,9,11,13])

print(set1.union(set2))

#8. Write a Python program to create set difference

set1=set([1,2,3,4,5])

set2=set([2,3,5,7,11])

print(set1.difference(set2)) # {1,4} set1實驗組 set2對照組

print(set2.difference(set1)) # {11,7}

#9. Write a Python program to create a symmetric difference

set1=set([1,2,3,4,5])

set2=set([1,3,5,7,9,11])

print(set1.symmetric\_difference(set2))

print(set2.symmetric\_difference(set1)) #the same {2,4,7,9,11}

print(set1^set2) #same meaning

#10. Write a Python program to check if a set is a subset of another set

set1=set([1,2,3,4,5,6,7,8,9])

set2=set([3,5,7])

set3=set([2,3,4,5,6])

set4=set([7,8,9,10,11])

print(set2.issubset(set1)) #True

print(set1.issubset(set2)) #False

print(set3.issubset(set1)) #True

print(set4.issubset(set1)) #False

print(set2<=set1) #True

print(set1>set2) #True

'''

11. Write a Python program to create a shallow copy of sets.

Note : Shallow copy is a bit-wise copy of an object. A new object is created that has an exact copy of the values in the original object.

'''

set1=set([1,2,3,4,4,5])

set2=set1.copy()

print(set2)

set1.add(6)

print(set1,set2) #{1, 2, 3, 4, 5, 6} {1, 2, 3, 4, 5}

# shallow copy, deep copy只有在 import copy時 nested list才有差

#12. Write a Python program to remove all elements from a given set

set1={1,2,3,4,4,5}

set1.clear()

print(set1)

'''

13. Write a Python program that uses frozensets.

Note: Frozensets behave just like sets except they are immutable.

'''

fset1=frozenset([1,2,3,4,5])

set1=set([1,2,3,4,5])

#Because frozenset objects are immutable, they can be used as dictionary keys, whereas set objects cannot

d = {frozenset({1, 2}): 'hello'}

print(d[frozenset({1, 2})]) # 'hello'

#Textbook

x = frozenset([1, 2, 3, 4, 5])

y = frozenset([3, 4, 5, 6, 7])

#use isdisjoint(). Return True if the set has no elements in common with other.

print(x.isdisjoint(y)) #False

#use difference(). Return a new set with elements in the set that are not in the others.

print(x.difference(y)) # frozenset({1, 2})

#new set with elements from both x and y

print(x | y) # frozenset({1, 2, 3, 4, 5, 6, 7})

'''

14. Write a Python program to find the maximum and minimum values in a set

'''

set1=set([1,2,3,4,5])

print(max(set1),min(set1))

#15. Write a Python program to find the length of a set

set1=set([1,2,3,4,5])

print(len(set1))

'''

16. Write a Python program to check if a given value is present in a set or not

'''

set1=set([1,2,3,4,5])

print(7 in set1)

print(3 in set1)

'''

17. Write a Python program to check if two given sets have no elements

in common

'''

def neic(set1,set2):

return not set1.intersection(set2)

print(neic({1,2,3,4,5},{6,7,8,9,10}))

print(neic({2,3,4},{4,5,6}))

#Textbook

x = {1,2,3,4}

y = {4,5,6,7}

z = {8}

print("Original set elements:")

print(x)

print(y)

print(z)

print("\nConfirm two given sets have no element(s) in common:")

print("\nCompare x and y:")

print(x.isdisjoint(y))

print("\nCompare x and z:")

print(z.isdisjoint(x))

'''

18. Write a Python program to check if a given set is a superset of

itself and a superset of another given set

'''

set1=set([1,2,3,4,5])

set2=set([3,4])

set3=set([5,6,7,8])

print(set1.issuperset(set2)) #True

print(set1.issuperset(set3)) #False

print(set1>set2) #True

'''

19. Write a Python program to find elements in a given set that are not

in another set

'''

def fenias(gset,aset):

return gset.difference(aset)

set1=set([1,2,3,4,5])

set2=set([4,5,6,7,8])

print(fenias(set1,set2))

print(set1-set2) #same meaning as above

'''

20. Write a Python program to remove the intersection of a second set

with a first set

'''

set1=set([1,2,3,4,5])

set2=set([3,4,5,6,7])

print(set2- (set1 & set2)) #{6,7}

#Textbook

sn1 = {1,2,3,4,5}

sn2 = {4,5,6,7,8}

print("Original sets:")

print(sn1)

print(sn2)

print("\nRemove the intersection of a 2nd set from the 1st set using difference\_update():")

sn1.difference\_update(sn2)

print("sn1: ",sn1) #{1,2,3}

print("sn2: ",sn2) #{4,5,6,7,8}

sn1 = {1,2,3,4,5}

sn2 = {4,5,6,7,8}

print(sn1.difference(sn2)) #{1,2,3}

print(sn1) #{1,2,3,4,5}

'''

21. Write a Python program to find all the unique words and count the

frequency of occurrence from a given list of strings.

Use Python set data type

'''

from collections import Counter

def fuw(lsts):

print(set(lsts))

print(Counter(lsts))

lst1=['apple','banana','gueva','egg','apple','egg','watermelon']

fuw(lst1)

#Textbook

def word\_count(words):

word\_set = set(words)

word\_counts = {}

for word in word\_set:

word\_counts[word] = words.count(word)

return word\_counts

words = ['Red', 'Green', 'Red', 'Blue', 'Red', 'Red', 'Green']

print(word\_count(words))

'''

22. Write a Python program that finds all pairs of elements in a list

whose sum is equal to a given value

'''

def fp(lsts,gv):

return [(lsts[i],lsts[j]) for i in range(len(lsts)) for j in range(len(lsts)) if i!=j and lsts[i]+lsts[j]==gv]

lst1=[2,4,5,8,6,1,7]

print(fp(lst1,10))

#Textbook

def find\_pairs(nums, target\_val):

nums\_set = set(nums)

pairs = []

for n in nums\_set:

complement = target\_val - n

if complement in nums\_set:

pairs.append({n, complement})

return pairs

'''

23. Write a Python program to find the longest common prefix of

all strings.

'''

import re

#warmup

lsts=['topone','toptwo','topthree']

chk= [lsts[0][:i] for i in range(1,len(lsts[0])+1)]

# print( [j for j in chk if j in k for k in lsts] )

print(chk)

res=[]

for k in range(1,len(lsts)):

for j in chk:

if j not in lsts[k]:break

else:res.append(j)

print(sorted(set(res),key=lambda x:len(x))[-1])

def lcp(lsts):

chk= [lsts[0][:i] for i in range(1,len(lsts[0])+1)]

res=[]

for k in range(1,len(lsts)):

for j in chk:

if j not in lsts[k]:break

else:res.append(j)

return sorted(set(res),key=lambda x:len(x))[-1]

lst1=['pitpengs','pitsteelers','pitbucs']

print(lcp(lst1))

#Textbook

def longest\_Common\_Prefix(strs):

if not strs:

return ""

min\_length = min([len(word) for word in strs])

for i in range(min\_length):

chars = set([word[i] for word in strs])

if len(chars) > 1:

return strs[0][:i]

return strs[0][:min\_length]

strs = ["pqrefgh","pqrsfgh"]

print("Original list of strings:")

print(strs)

print("Longest common prefix of all said strings:")

print(longest\_Common\_Prefix(strs))

strs = ["w3r","w3resource"]

print("\nOriginal list of strings:")

print(strs)

print("Longest common prefix of all said strings:")

print(longest\_Common\_Prefix(strs))

strs = ["Python","PHP", "Perl"]

print("\nOriginal list of strings:")

print(strs)

print("Longest common prefix of all said strings:")

print(longest\_Common\_Prefix(strs))

strs = ["Python","HTML", "PHP"]

print("\nOriginal list of strings:")

print(strs)

print("Longest common prefix of all said strings:")

print(longest\_Common\_Prefix(strs))

'''

24. Write a Python program to find the two numbers whose product is

maximum among all the pairs in a given list of numbers.

Use the Python set

'''

from itertools import product

def fpm(lsts):

return max([i[0]\*i[1] for i in product(set(lsts),set(lsts)) if i[0]!=i[1]])

print(fpm([4,6,3,8,7,2]))

#Textbook

def max\_product(nums):

nums\_set = set(nums)

max\_product = float('-inf')

max\_num1 = None

max\_num2 = None

for n1 in nums\_set:

for n2 in nums\_set:

if n1 != n2 and n1 \* n2 > max\_product:

max\_product = n1 \* n2

max\_num1 = n1

max\_num2 = n2

return (max\_num1, max\_num2)

nums = [1, 2, 3, 4, 5, 6, 7, 8, 9]

print("Original list of integers:")

print(nums)

print("Maximum product of two numbers among all pairs in the said list:")

print(max\_product(nums))

'''

25. Given two sets of numbers, write a Python program to find the missing

numbers in the second set as compared to the first and vice versa.

Use the Python set

'''

def fmni2s(set1,set2):

return set1.difference(set2)

print(fmni2s({3,4,5,6},{3,5,7,8,9}))

#Textbook

def missing\_numbers(set\_nums1, set\_nums2):

return set(set\_nums1) - set(set\_nums2),set(set\_nums2) - set(set\_nums1)

set\_nums1 = {1, 2, 3, 4, 5, 6}

set\_nums2 = {3, 4, 5, 6, 7, 8}

result = missing\_numbers(set\_nums1, set\_nums2)

print("Original sets:")

print(set\_nums1)

print(set\_nums2)

print("Missing numbers in the second set as compared to the first:")

print(result[0])

'''

26. Write a Python program to find all the anagrams and group them

together from a given list of strings. Use the Python data type

'''

from itertools import permutations

str1=input('input the word for anagram game:\n')

res=[]

for i in permutations(str1,len(str1)):

res.append(''.join(i))

print(res)

#Textbook -different meaning

def group\_anagrams(strs):

result = {}

for s in strs:

sorted\_string = ''.join(sorted(s))

if sorted\_string in result:

result[sorted\_string].append(s)

else:

result[sorted\_string] = [s]

return list(result.values())

strs = ['eat', 'cba', 'tae', 'abc', 'xyz']

print("Original list of strings:")

print(strs)

print("Find and group all anagrams in the said list:")

print(group\_anagrams(strs))

'''

27. Write a Python program to find all the anagrams in a given list of

strings and then group them together. Use the Python data type

'''

#Textbook

def group\_anagrams(strs):

result = {}

for s in strs:

sorted\_string = ''.join(sorted(s))

if sorted\_string in result:

result[sorted\_string].append(s)

else:

result[sorted\_string] = [s]

return list(result.values())

'''

28. Write a Python program to find all the unique combinations of

3 numbers from a given list of numbers, adding up to a target number

'''

from itertools import combinations

def uco3(lsts):

for i in combinations(lsts,3):

print(i,sum(i))

uco3([3,6,7,1,9,5,8])

#Textbook

def find\_combinations\_of\_three(nums, target\_val):

nums = list(set(nums))

result = set()

for i in range(len(nums)):

for j in range(i+1, len(nums)):

complement = target\_val - nums[i] - nums[j]

if complement in nums[:i] + nums[j+1:]:

result.add(tuple(sorted((nums[i], nums[j], complement))))

return list(result)

nums = [1, 2, 3, 4, 5, 6, 7, 8, 9]

target\_val = 12

print("Original list of numbers:")

print(nums)

print("Target value:",target\_val)

print("Combine three numbers whose sum equal to a target number:")

print(find\_combinations\_of\_three(nums, target\_val))

'''

29. Write a Python program to find the third largest number from a given

list of numbers.Use the Python set data type

'''

lsts=[5,3,2,0,5,6,1]

print(sorted(set(lsts))[-3]) #3

#Textbook

def third\_largest(nums):

nums = set(nums)

if len(nums) < 3:

return None

nums = list(nums)

nums.sort(reverse=True)

return nums[2]

'''

30. Write a Python program to remove all duplicates from a given list

of strings and return a list of unique strings. Use the Python set data type

'''

lst1=[2,3,6,7,5,8,9,2]

print(list(set(lst1)))