1. Python – Sort Dictionary key and values List

test\_dict = {'gfg': [7, 6, 3],

'is': [2, 10, 3],

'best': [19, 4]}

print("The original dictionary is : " + str(test\_dict))

res = dict()

for key in sorted(test\_dict):

res[key] = sorted(test\_dict[key])

print("The sorted dictionary : " + str(res))

1. Handling missing keys in Python dictionaries

d = { 'a' : 1 , 'b' : 2 }

print ("The value associated with 'c' is : ")

print (d['c'])

1. Python dictionary with keys having multiple inputs

import random as rn

dict = {}

x, y, z = 10, 20, 30

dict[x, y, z] = x + y - z;

x, y, z = 5, 2, 4

dict[x, y, z] = x + y - z;

print(dict)

1. Print anagrams together in Python using List and Dictionary

are\_anagrams = lambda x, y: str(sorted(x.lower())) == str(sorted(y.lower()))

print(are\_anagrams('cat', 'tac'))

print(are\_anagrams('cat', 'Tac'))

print(are\_anagrams('cat', 'dog'))

1. K’th Non-repeating Character in Python using List Comprehension and OrderedDict

from collections import OrderedDict

def kthRepeating(input,k):

dict=OrderedDict.fromkeys(input,0)

for ch in input:

dict[ch]+=1

nonRepeatDict = [key for (key,value) in dict.items() if value==1]

if len(nonRepeatDict) < k:

return 'Less than k non-repeating characters in input.'

else:

return nonRepeatDict[k-1]

if \_\_name\_\_ == "\_\_main\_\_":

input = "geeksforgeeks"

k = 3

print (kthRepeating(input, k))

1. Check if binary representations of two numbers are anagram

from collections import Counter

def checkAnagram(num1,num2):

bin1 = bin(num1)[2:]

bin2 = bin(num2)[2:]

zeros = abs(len(bin1)-len(bin2))

if (len(bin1)>len(bin2)):

bin2 = zeros \* '0' + bin2

else:

bin1 = zeros \* '0' + bin1

dict1 = Counter(bin1)

dict2 = Counter(bin2)

if dict1 == dict2:

print('Yes')

else:

print('No')

if \_\_name\_\_ == "\_\_main\_\_":

num1 = 8

num2 = 4

checkAnagram(num1,num2)

1. Python Counter to find the size of largest subset of anagram words

from collections import Counter

def maxAnagramSize(input):

input = input.split(" ")

for i in range(0,len(input)):

input[i]=''.join(sorted(input[i]))

freqDict = Counter(input)

print (max(freqDict.values()))

if \_\_name\_\_ == "\_\_main\_\_":

input = 'ant magenta magnate tan gnamate'

maxAnagramSize(input)

1. Python | Remove all duplicates words from a given sentence

from collections import Counter

def remov\_duplicates(input):

input = input.split(" ")

UniqW = Counter(input)

s = " ".join(UniqW.keys())

print (s)

if \_\_name\_\_ == "\_\_main\_\_":

input = 'Python is great and Java is also great'

remov\_duplicates(input)

1. Python Dictionary to find mirror characters in a string

def mirrorChars(input,k):

original = 'abcdefghijklmnopqrstuvwxyz'

reverse = 'zyxwvutsrqponmlkjihgfedcba'

dictChars = dict(zip(original,reverse))

prefix = input[0:k-1]

suffix = input[k-1:]

mirror = ''

for i in range(0,len(suffix)):

mirror = mirror + dictChars[suffix[i]]

print (prefix+mirror)

if \_\_name\_\_ == "\_\_main\_\_":

input = 'paradox'

k = 3

mirrorChars(input,k)

1. Counting the frequencies in a list using dictionary in Python

def CountFrequency(my\_list):

freq = {}

for item in my\_list:

if (item in freq):

freq[item] += 1

else:

freq[item] = 1

for key, value in freq.items():

print ("% d : % d"%(key, value))

if \_\_name\_\_ == "\_\_main\_\_":

my\_list =[1, 1, 1, 5, 5, 3, 1, 3, 3, 1, 4, 4, 4, 2, 2, 2, 2]

CountFrequency(my\_list)