1. Python | Convert a list of Tuples into Dictionary

def Convert(tup, di):

for a, b in tup:

di.setdefault(a, []).append(b)

return di

tups = [("akash", 10), ("gaurav", 12), ("anand", 14),

("suraj", 20), ("akhil", 25), ("ashish", 30)]

dictionary = {}

print (Convert(tups, dictionary))

1. Python counter and dictionary intersection example (Make a string using deletion and rearrangement)

from collections import Counter

def makeString(str1,str2):

dict1 = Counter(str1)

dict2 = Counter(str2)

result = dict1 & dict2

return result == dict1

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

str1 = 'ABHISHEKsinGH'

str2 = 'gfhfBHkooIHnfndSHEKsiAnG'

if (makeString(str1,str2)==True):

print("Possible")

else:

print("Not Possible")

1. Python dictionary, set and counter to check if frequencies can become same

from collections import Counter

def allSame(input):

dict=Counter(input)

same = list(set(dict.values()))

if len(same)>2:

print('No')

elif len (same)==2 and same[1]-same[0]>1:

print('No')

else:

print('Yes')

1. Scraping And Finding Ordered Words In A Dictionary using Python

import requests

def Words\_find():

my\_url = ""#put thisurl of .txt files in any website

my\_fetchData = requests.get(my\_url)

my\_wordList = my\_fetchData.content

my\_wordList = my\_wordList.decode("utf-8").split()

return my\_wordList

def wordordered():

collection = Words\_find()

collection = collection[16:]

my\_word = ''

for my\_word in collection:

result = 'ordered'

i = 0

l = len(my\_word) - 1

if (len(my\_word) < 3):

continue

while i < l:

if (ord(my\_word[i]) > ord(my\_word[i+1])):

result = 'not ordered'

break

else:

i += 1

if (result == 'ordered'):

print(my\_word,': ',result)

if \_\_name\_\_ == '\_\_main\_\_':

wordordered()

1. Possible Words using given characters in Python

def charCount(word):

dict = {}

for i in word:

dict[i] = dict.get(i, 0) + 1

return dict

def possible\_words(lwords, charSet):

for word in lwords:

flag = 1

chars = charCount(word)

for key in chars:

if key not in charSet:

flag = 0

else:

if charSet.count(key) != chars[key]:

flag = 0

if flag == 1:

print(word)

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

input = ['goo', 'bat', 'me', 'eat', 'goal', 'boy', 'run']

charSet = ['e', 'o', 'b', 'a', 'm', 'g', 'l']

possible\_words(input, charSet)

1. Python – Keys associated with Values in Dictionary

test\_dict = {'gfg' : [4, 5], 'is' : [8], 'best' : [10, 12]}

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

val\_list = [5, 10]

temp = {}

for key, vals in test\_dict.items():

for val in vals:

temp[val] = key

res = [temp[ele] for ele in val\_list]

print("The keys mapped to " + str(val\_list) + " are : " + str(res))

1. Python program to Find the size of a Tuple

import sys

Tuple1 = ("A", 1, "B", 2, "C", 3)

Tuple2 = ("Geek1", "Raju", "Geek2", "Nikhil", "Geek3", "Deepanshu")

Tuple3 = ((1, "Lion"), ( 2, "Tiger"), (3, "Fox"), (4, "Wolf"))

print("Size of Tuple1: " + str(sys.getsizeof(Tuple1)) + "bytes")

print("Size of Tuple2: " + str(sys.getsizeof(Tuple2)) + "bytes")

print("Size of Tuple3: " + str(sys.getsizeof(Tuple3)) + "bytes")

1. Python – Maximum and Minimum K elements in Tuple

test\_tup = (5, 20, 3, 7, 6, 8)

print("The original tuple is : " + str(test\_tup))

K = 2

res = []

test\_tup = list(sorted(test\_tup))

for idx, val in enumerate(test\_tup):

if idx < K or idx >= len(test\_tup) - K:

res.append(val)

res = tuple(res)

print("The extracted values : " + str(res))

1. Create a list of tuples from given list having number and its cube in each tuple

list1 = [1, 2, 5, 6]

res = [(val, pow(val, 3)) for val in list1]

print(res)

1. Python – Adding Tuple to List and vice – versa

test\_list = [5, 6, 7]

print("The original list is : " + str(test\_list))

test\_tup = (9, 10)

test\_list += test\_tup

print("The container after addition : " + str(test\_list))

1. Python – Closest Pair to Kth index element in Tuple

test\_list = [(3, 4), (78, 76), (2, 3), (9, 8), (19, 23)]

print("The original list is : " + str(test\_list))

tup = (17, 23)

K = 1

min\_dif, res = 999999999, None

for idx, val in enumerate(test\_list):

dif = abs(tup[K - 1] - val[K - 1])

if dif < min\_dif:

min\_dif, res = dif, idx

print("The nearest tuple to Kth index element is : " + str(test\_list[res]))