Python | Sort Python Dictionaries by Key or Value

# Function calling

def dictionairy():

 # Declare hash function

 key\_value ={}

# Initializing value

 key\_value[2] = 56

 key\_value[1] = 2

 key\_value[5] = 12

 key\_value[4] = 24

 key\_value[6] = 18

 key\_value[3] = 323

 print ("Task 1:-\n")

 print ("Keys are")

 # iterkeys() returns an iterator over the

 # dictionary’s keys.

 for i in sorted (key\_value.keys()) :

     print(i, end = " ")

def main():

    # function calling

    dictionairy()

# Main function calling

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

    main()

OR

 # sorted(key\_value) returns an iterator over the

 # Dictionary’s value sorted in keys.

 for i in sorted (key\_value) :

    print ((i, key\_value[i]), end =" ")

# Ways to remove a key from dictionary

# Initializing dictionary

test\_dict = {"Arushi" : 22, "Anuradha" : 21, "Mani" : 21, "Haritha" : 21}

# Printing dictionary before removal

print ("The dictionary before performing remove is : " + str(test\_dict))

# Using del to remove a dict

# removes Mani

del test\_dict['Mani']

OR

# Initializing dictionary

test\_dict = {"Arushi" : 22, "Anuradha" : 21, "Mani" : 21, "Haritha" : 21}

# Printing dictionary before removal

print ("The dictionary before performing remove is : " + str(test\_dict))

# Using pop() to remove a dict. pair

# removes Mani

removed\_value = test\_dict.pop('Mani')

# Ways to sort list of dictionaries by values in Python – Using itemgetter

# importing "operator" for implementing itemgetter

from operator import itemgetter

# Initializing list of dictionaries

lis = [{ "name" : "Nandini", "age" : 20},

{ "name" : "Manjeet", "age" : 20 },

{ "name" : "Nikhil" , "age" : 19 }]

# using sorted and itemgetter to print list sorted by age

print "The list printed sorting by age: "

print sorted(lis, key=itemgetter('age'))

print ("\r")

# using sorted and itemgetter to print list sorted by both age and name

# notice that "Manjeet" now comes before "Nandini"

print "The list printed sorting by age and name: "

print sorted(lis, key=itemgetter('age', 'name'))

# Ways to sort list of dictionaries by values in Python – Using lambda function

# Initializing list of dictionaries

lis = [{ "name" : "Nandini", "age" : 20},

{ "name" : "Manjeet", "age" : 20 },

{ "name" : "Nikhil" , "age" : 19 }]

# using sorted and lambda to print list sorted

# by age

print "The list printed sorting by age: "

print sorted(lis, key = lambda i: i['age'])

print ("\r")

# using sorted and lambda to print list sorted

# by both age and name. Notice that "Manjeet"

# now comes before "Nandini"

print "The list printed sorting by age and name: "

print sorted(lis, key = lambda i: (i['age'], i['name']))

# Python | Merging two Dictionaries

>>> dict1 = {'a': 10, 'b': 8}

>>> dict2 = {'d': 6, 'c': 4}

>>> dict2.update(dict1)

>>> print(dict2)

{'d': 6, 'c': 4, 'a': 10, 'b': 8}

>>> print(dict1)

{'a': 10, 'b': 8}

>>>

# Python Dictionary | Check if binary representations of two numbers are anagram

Given two numbers you are required to check whether they are anagrams of each other or not in binary representation.

Examples:

Input : a = 8, b = 4

Output : Yes

Binary representations of both

numbers have same 0s and 1s.

Input : a = 4, b = 5

Output : No

1. Convert both number into it’s binary using [bin()](https://www.geeksforgeeks.org/bin-in-python/) function.
2. Since binary representation of both numbers could differ in length so we will append zeros in start of shorter string to make both string of equal length. ie.; append zeros = abs(len(bin1)-len(bin2)).
3. Convert both output string containing 0 and 1 returned by bin function into dictionary using **Counter()** function, having 0 and 1 keys and their count as value. Compare both dictionaries, if value of 0’s and 1’s in both dictionaries are equal then binary representations of two numbers are anagram otherwise not.

from collections import Counter

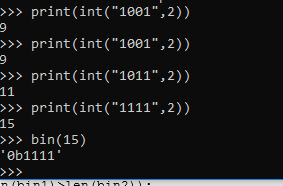
def checkAnagram(num1,num2):

    # convert numbers into in binary

    # and remove first two characters of

    # output string because bin function

    # '0b' as prefix in output string



    bin1 = bin(num1)[2:]

    bin2 = bin(num2)[2:]

    # append zeros in shorter string

    zeros = abs(len(bin1)-len(bin2))

    if (len(bin1)>len(bin2)):

         bin2 = zeros \* '0' + bin2

    else:

         bin1 = zeros \* '0' + bin1

    # convert binary representations

    # into dictionary

    dict1 = Counter(bin1)

    dict2 = Counter(bin2)

    # compare both dictionaries

    if dict1 == dict2:

         print('Yes')

    else:

         print('No')

# Driver program

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

    num1 = 8

    num2 = 4

    checkAnagram(num1,num2)