1. **Python | 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 items() + dict comprehension to remove a dict. pair

# removes Mani

new\_dict = {key: val for key,

      val in test\_dict.items() if key != 'Mani'}

# Printing dictionary after removal

print("The dictionary after remove is : " + str(new\_dict))

**Output:** The dictionary before performingremove is: {'Arushi': 22, 'Anuradha': 21, 'Mani': 21, 'Haritha': 21}

The dictionary after remove is: {'Arushi': 22, 'Anuradha': 21, 'Haritha': 21}

1. **Ways to sort list of dictionaries by values in Python – Using itemgetter**

# Python code demonstrate the working of sorted()

# and itemgetter

# importing "operator" for implementing itemgetter

from operator import itemgetter

# Initializing list of dictionaries

list = [{"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: ", sorted(list, key=itemgetter('age')))

# 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: ", sorted(list, key=itemgetter('age', 'name')))

# using sorted and itemgetter to print list

# sorted by age in descending order

print("The list printed sorting by age in descending order: ", sorted(list, key=itemgetter('age'), reverse=True))

**Output:** The list printed sorting by age: [{'name': 'Nikhil', 'age': 19}, {'name': 'Nandini', 'age': 20}, {'name': 'Manjeet', 'age': 20}]

The list printed sorting by age and name: [{'name': 'Nikhil', 'age': 19}, {'name': 'Manjeet', 'age': 20}, {'name': 'Nandini', 'age': 20}]

The list printed sorting by age in descending order: [{'name': 'Nandini', 'age': 20}, {'name': 'Manjeet', 'age': 20}, {'name': 'Nikhil', 'age': 19}]

1. **Ways to sort list of dictionaries by values in Python – Using lambda function**

# Python code demonstrate the working of

# sorted() with lambda

# Initializing list of dictionaries

list = [{"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(list, 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(list, key=lambda i: (i['age'], i['name'])))

print("\r")

# using sorted and lambda to print list sorted

# by age in descending order

print("The list printed sorting by age in descending order: ")

print(sorted(list, key=lambda i: i['age'], reverse=True))

**Output:** The list printed sorting by age:

[{'name': 'Nikhil', 'age': 19}, {'name': 'Nandini', 'age': 20}, {'name': 'Manjeet', 'age': 20}]

The list printed sorting by age and name:

[{'name': 'Nikhil', 'age': 19}, {'name': 'Manjeet', 'age': 20}, {'name': 'Nandini', 'age': 20}]

The list printed sorting by age in descending order:

[{'name': 'Nandini', 'age': 20}, {'name': 'Manjeet', 'age': 20}, {'name': 'Nikhil', 'age': 19}]

1. **Python | Merging two Dictionaries**

# Python code to merge dict using update() method

def Merge(dict1, dict2):

  return(dict2.update(dict1))

# Driver code

dict1 = {'a': 10, 'b': 8}

dict2 = {'d': 6, 'c': 4}

# This returns None

print(Merge(dict1, dict2))

# changes made in dict2

print(dict2)

**Output:** None

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

1. **Python – Convert key-values list to flat dictionary**

# Python3 code to demonstrate working of

# Convert key-values list to flat dictionary

# Using dict() + zip()

from itertools import product

# initializing dictionary

test\_dict = {'month' : [1, 2, 3],

      'name' : ['Jan', 'Feb', 'March']}

# printing original dictionary

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

# Convert key-values list to flat dictionary

# Using dict() + zip()

res = dict(zip(test\_dict['month'], test\_dict['name']))

# printing result

print("Flattened dictionary : " + str(res))

**Output:** The original dictionary is : {'month': [1, 2, 3], 'name': ['Jan', 'Feb', 'March']}

Flattened dictionary : {1: 'Jan', 2: 'Feb', 3: 'March'}

1. **Python – Insertion at the beginning in OrderedDict**

# Python code to demonstrate

# insertion of items in beginning of ordered dict

from collections import OrderedDict

# initialising ordered\_dict

iniordered\_dict = OrderedDict([('akshat', '1'), ('nikhil', '2')])

# inserting items in starting of dict

iniordered\_dict.update({'manjeet':'3'})

iniordered\_dict.move\_to\_end('manjeet', last = False)

# print result

print ("Resultant Dictionary : "+str(iniordered\_dict))

**Output:** Resultant Dictionary : OrderedDict([('manjeet', '3'), ('akshat', '1'), ('nikhil', '2')])

1. **Python | Check order of character in string using OrderedDict( )**

# Function to check if string follows order of

# characters defined by a pattern

from collections import OrderedDict

def checkOrder(input, pattern):

  # create empty OrderedDict

  # output will be like {'a': None,'b': None, 'c': None}

  dict = OrderedDict.fromkeys(input)

  # traverse generated OrderedDict parallel with

  # pattern string to check if order of characters

  # are same or not

  ptrlen = 0

  for key,value in dict.items():

    if (key == pattern[ptrlen]):

      ptrlen = ptrlen + 1

    # check if we have traverse complete

    # pattern string

    if (ptrlen == (len(pattern))):

      return 'true'

  # if we come out from for loop that means

  # order was mismatched

  return 'false'

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

  input = 'engineers rock'

  pattern = 'er'

  print (checkOrder(input,pattern))

**Output:** True

1. **Dictionary and counter in Python to find winner of election**

# Function to find winner of an election where votes

# are represented as candidate names

from collections import Counter

def winner(input):

  # convert list of candidates into dictionary

  # output will be likes candidates = {'A':2, 'B':4}

  votes = Counter(input)

  # create another dictionary and it's key will

  # be count of votes values will be name of

  # candidates

  dict = {}

  for value in votes.values():

    # initialize empty list to each key to

    # insert candidate names having same

    # number of votes

    dict[value] = []

  for (key,value) in votes.items():

    dict[value].append(key)

  # sort keys in descending order to get maximum

  # value of votes

  maxVote = sorted(dict.keys(),reverse=True)[0]

  # check if more than 1 candidates have same

  # number of votes. If yes, then sort the list

  # first and print first element

  if len(dict[maxVote])>1:

    print (sorted(dict[maxVote])[0])

  else:

    print (dict[maxVote][0])

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

  input =['john','johnny','jackie','johnny',

      'john','jackie','jamie','jamie',

      'john','johnny','jamie','johnny',

      'john']

  winner(input)

**Output:** john

1. **Python – Append Dictionary Keys and Values ( In order ) in dictionary**

# Python3 code to demonstrate working of

# Append Dictionary Keys and Values ( In order ) in dictionary

# Using values() + keys() + list()

# initializing dictionary

test\_dict = {"Gfg" : 1, "is" : 3, "Best" : 2}

# printing original dictionary

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

# + operator is used to perform adding keys and values

res = list(test\_dict.keys()) + list(test\_dict.values())

# printing result

print("The ordered keys and values : " + str(res))

**Output:** The original dictionary is : {'Gfg': 1, 'is': 3, 'Best': 2}

The ordered keys and values : ['Gfg', 'is', 'Best', 1, 3, 2]

1. **Python | Sort Python Dictionaries by Key or Value**

myDict = {'ravi': 10, 'rajnish': 9,'sanjeev': 15, 'yash': 2, 'suraj': 32}

myKeys = list(myDict.keys())

myKeys.sort()

sorted\_dict = {i: myDict[i] for i in myKeys}

print(sorted\_dict)

**Output:** {'rajnish': 9, 'ravi': 10, 'sanjeev': 15, 'suraj': 32, 'yash': 2}