In [1]:
```
#31. Create a list of tuples from given list having number and its cube in each tuple.
def cubeoflist(li):
    result=[(num, num**3) for num in li]
    return result
li = [3, 4, 1, 2]
print(cubeoflist(li))
```

```
[(3, 27), (4, 64), (1, 1), (2, 8)]
```

In [2]:
```
#32.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)
```

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

In [3]:
```
#33. Python dictionary with keys having multiple values.
dic = {}

a,b,c= 5, 3, 10

p,q,r= 12, 6, 9
dic["x-y+z"] = [a-b+c,p-q+r]
print(dic)
```

```
{'x-y+z': [12, 15]}
```

In [6]:
```
#34. Python program to find the sum of all items in a dictionary.
dic={ 'x':455, 'y':223, 'z':300, 'p':908 }

print("Dictionary: ", dic)

#using sum() and values()
print("sum: ",sum(dic.values()))
```

```
Dictionary:  {'x': 455, 'y': 223, 'z': 300, 'p': 908}
sum:  1886
```

In [7]:
```
#35. Python program to find the size of a dictionary
import sys
dic1 = {"A": 1, "B": 2, "C": 3}
dic2 = {"Geek1": "Raju", "Geek2": "Nikhil", "Geek3": "Deepanshu"}
dic3 = {1: "Lion", 2: "Tiger", 3: "Fox", 4: "Wolf"}
print("Size of dic1: " + str(sys.getsizeof(dic1)) + "bytes")
print("Size of dic2: " + str(sys.getsizeof(dic2)) + "bytes")
print("Size of dic3: " + str(sys.getsizeof(dic3)) + "bytes")
```

```
Size of dic1: 232bytes
Size of dic2: 232bytes
Size of dic3: 232bytes
```

In [8]:
```
#36. Find the size of a set in Python
import sys
Set1 = {"A", 1, "B", 2, "C", 3}
Set2 = {"Geek1", "Raju", "Geek2", "Nikhil", "Geek3", "Deepanshu"}
Set3 = {(1, "Lion"), ( 2, "Tiger"), (3, "Fox")}
print("Size of Set1: " + str(sys.getsizeof(Set1)) + "bytes")
print("Size of Set2: " + str(sys.getsizeof(Set2)) + "bytes")
print("Size of Set3: " + str(sys.getsizeof(Set3)) + "bytes")
```

```
Size of Set1: 472bytes
Size of Set2: 472bytes
Size of Set3: 216bytes
```

In [10]:
```
#37. Iterate over a set in Python

test_set = set("geEks")


for val in test_set:
    print(val)
```

```
s
e
k
E
g
```

In [11]:
```
#38. Python- Maximum and minimum in a Set
def MAX(sets):
    return (max(sets))
sets=set([23,8,19,28,25,78,98,345,678,999])
print("The maximum element in the set is:",MAX(sets))
def MIN(sets):
    return(min(sets))
print("The minimum element in the set is:",MIN(sets))
```

```
The maximum element in the set is: 999
The minimum element in the set is: 8
```

In [12]:
```
#39. Python-Remove items from a set
languages={'Python','Java','English','C','C++','Tamil','Hindi'}
languages.remove('C')
print(languages)
```

```
{'Python', 'Java', 'C++', 'English', 'Hindi', 'Tamil'}
```

In [15]:
```
#40. Python-Check if two lists have atleast one element in common
def common_data(list1,list2):
    result=False
    for x in list1:
        for y in list2:
            if x==y:
                result=True
                return result
    return result
a=[1,2,3,4,5]
b=[5,6,7,8,9]
print(common_data(a,b))
a=[1,2,3,4,5]
b=[6,7,8,9]
print(common_data(a,b))
```

```
True
False
```

In [23]:
```
#41. Python-Assigning subsequent rows to matrix first row elements
test_list=[[5,8,9],[2,0,9],[5,4,2],[2,3,9]]
print("The original list:"+str(test_list))
res={test_list[0][ele]:test_list[ele+1] for ele in range(len(test_list)-1)}
print("The assigned matrix:" +str(res))
```

```
The original list:[[5, 8, 9], [2, 0, 9], [5, 4, 2], [2, 3, 9]]
The assigned matrix:{5: [2, 0, 9], 8: [5, 4, 2], 9: [2, 3, 9]}
```

In [25]:
```
#42. Adding and Subtracting matrices in python
import numpy as np
A=np.array([[1,2],[3,4]])
B=np.array([[4,5],[6,7]])
print("Printing elements of first matrix")
print(A)
print("Printing elements of second matrix")
print(B)
print("Addition of two matrix")
print(np.add(A,B))
print("Subtraction of two matrix")
print(np.subtract(A,B))
```

```
Printing elements of first matrix
[[1 2]
 [3 4]]
Printing elements of second matrix
[[4 5]
 [6 7]]
Addition of two matrix
[[ 5  7]
 [ 9 11]]
Subtraction of two matrix
[[-3 -3]
 [-3 -3]]
```

In [26]:
```
#43. Python-Group similar elements into Matrix
from itertools import groupby
test_list=[1,3,5,1,3,2,5,4,2]
print("The original list:" +str(test_list))
res =[list(val) for key, val in groupby(sorted(test_list))]
print("Matrix after grouping:"+str(res))
```

```
The original list:[1, 3, 5, 1, 3, 2, 5, 4, 2]
Matrix after grouping:[[1, 1], [2, 2], [3, 3], [4], [5, 5]]
```

In [ ]:
```
#44.Python-Row wise element addition in tuple matrix.
test_list=[('Gfg',3), ('is',3)], [('best',1)], [('for',5), ('geeks',1)]]
print("The original list is:"+ str(test_list))
cus_eles=[6,7,8]
res=[[sub+cus_eles[idx],) for sub in val] for idx, val in enumerate(test_list)]
print("The matrix after row elements addition :"+str(res))
```

In [1]:
```
#45. Create an n x n square matrix, where all the sub-matrix has the sum of opposite corner elements as even.
import itertools


def sub_mat_even(n):

    temp = itertools.count(1)

    # create a 2d array ranging
    # from 1 to n^2
    l = [[next(temp)for i in range(n)]for i in range(n)]

    # If found even we reverse the alternate
    # row elements to get all diagonal elements
    # as all even or all odd
    if n%2 == 0:
        for i in range(0,len(l)):
            if i%2 == 1:
                l[i][:] = l[i][::-1]

# Printing the array formed
    for i in range(n):
        for j in range(n):
            print(l[i][j],end=" ")
        print()

n = 4
sub_mat_even(n)
```

```
1 2 3 4
8 7 6 5
9 10 11 12
16 15 14 13
```

In [1]:
```
#46)How to get list of parameters name from a function in Python?

def fun(a, b):
    return a**b

# import required modules
import inspect

# use signature()
print(inspect.signature(fun))
```

```
(a, b)
```

In [2]:
```
#47)How to Print Multiple Arguments in Python?
def GFG(name, num="25"):
    print("Hello from", name + ', ' + num)


GFG("gfg")
GFG("gfg", "26")
```

```
Hello from gfg, 25
Hello from gfg, 26
```

In [12]:
```
#48)Python program to find the power of a number using recursion
def power(N, P):


    if P == 0:
        return 1

    return (N*power(N, P-1))


if __name__ == '__main__':
    N = 5
    P = 2

    print(power(N, P))
```

```
25
```

In [13]:
```
#49)Sorting objects of user defined class in Python

class GFG:
    def __init__(self, a, b):
        self.a = a
        self.b = b

    def __repr__(self):
        return str((self.a, self.b))



gfg = [GFG("geeks", 1),
       GFG("computer", 3),
       GFG("for", 2),
       GFG("geeks", 4),
       GFG("science", 3)]

print(sorted(gfg, key=lambda x: x.b))
```

```
[('geeks', 1), ('for', 2), ('computer', 3), ('science', 3), ('geeks', 4)]
```

In [14]:
```
#50)Functions that accept variable length key value pair as arguments

def printKwargs(**kwargs):
    print(kwargs)


if __name__ == "__main__":
    printKwargs(Argument_1='gfg', Argument_2='GFG')
```

```
{'Argument_1': 'gfg', 'Argument_2': 'GFG'}
```