In [7]:
```
# s = 123abc456def
# 0 1 1 1 1 1 1 0 0 0   (String) - Frequency of sorted numbers
# count(1) --> 1 4 1 1 1 1 4 4 4

# s = c
# 0 0 0 0 0 0 0 0 0 0

# s = 1234567890
# 1 1 1 1 1 1 1 1 1 1

def uniqueData(allnumbers):
    unique = []
    for n in allnumbers:
        if n not in unique:
            unique.append(n)
    return unique

def digitFrequency1(s):
    allnumbers = []
    for i in s:
        if i.isdigit():
            allnumbers.append(i)
    unique = uniqueData(allnumbers)
    for i in range(0,10):
        if str(i) not in unique:
            print(0,end = ' ')
        else:
            c=allnumbers.count(str(i))
            print(c, end = ' ')
digitFrequency1('212abc456def111')


```

```
0 4 2 0 1 1 1 0 0 0
```

In [4]:
```
#Second solution:

def digitFrequency2(s):
    for i in range(0,10):
        count = s.count(str(i))

digitFrequency2('212abc456def111')
```

In [ ]:
```

```

In [ ]:
```

```

Contacts Application :

- Addcontact(name,phone,email)
- searchcontact(name)
- listcontact()
- modify/editcontact(name, newphone, newemail)
- deletecontact(name)
- contactsapp()

In [8]:

```
#Function to validate a phone number
import re

def phoneNumValidator(num):
    pattern ='^[6-9][0-9]{9}$|^[0][6-9][0-9]{9}$|^[+][9][1][6-9][0-9]{9}$'
    if re.match(pattern,str(num)):
        return True
    return False


def emailValidator(email):
    pattern="^[0-9a-z][0-9a-z_.]{4,13}[0-9a-z][@][a-z0-9]{3,18}[.][a-z]{2,4}
    if re.match(pattern,email):
        return True
    return False
#Checking whether the file contains the contacts
def contactExists(name):
    filename='DataFiles/contacts.txt'
    with open(filename,'r') as f:
        p=name+'r'
        fd=f.read()
    return re.search(p,fd)

#Add contact
def addContact(name, phone, email):
    filename='DataFiles/contacts.txt'
    # store data as name,phone,email in the contacts file
    if not contactExists(name):
        if phoneNumValidator(phone) and emailValidator(email):
            with open(filename, 'a') as f:
                line = name + ',' + str(phone) + ',' + email + '\n'
                f.write(line)
            print(name, ' is added to contacts')
        else:
            print('Invalid Phone number or Email')
            return
    else:
        print(name, 'already exists')
    return
addContact('name6',9876543219,'name6_12@gmail.com')
#Search contact

def searchContact(name):
    filename='DataFiles/contacts.txt'
    if contactExists(name):
        with open(filename,'r') as f:
            for i in f:
                i=i.split(',')
                if i[0]==name:
                    print(i[0],i[1],i[2])

#searchContact("name2")
# files to list

def listContacts():
    with open('DataFiles/contacts.txt','r') as f:
```

```
        if len(f.read()) != 0:
            for i in f:
                i = i.split(',')
                print(i[0],i[1],i[2])
        else:
            print('No Contact exists')
listContacts()



def saveToFile(s):
    with open("DataFiles/contacts.txt",'w') as f:
        f.write(s)
        print("\n \nData updated...")


def listToContacts(ls):
    s = ''
    for i in range(0,len(ls)):
        lls = ls[i]
        for j in range(0,len(lls)):
            s+=str(lls[j])+','
    return s
#listToContacts(ls)

def editContact(name,number,email):
    ls = contactsToList('DataFiles/contacts.txt')
    for i in range(0,len(ls)):
        lss = ls[i]
        for j in range(0,len(lss)):
            if(name == lss[j]):
                lss[0] = name
                lss[1] = number
                lss[2] = email
                print(name," Updated..")
        else:
            print("not updated")
    s=listToContacts(ls)
    saveToFile(s)
    return s

#editContact('name2',9876543219,'name3_n@gmail.com')

```

```
name6  is added to contacts
```

In [13]:

```
# Function to check if two strings are anagrams

# abc cba --> True

# {a:1,b:1,c:!} {c:1,a:1,b:1}
# aabbcc ccbbaaa ---> False
# {a:2,b:2,c:2} {a:3,b:2,c:2}

# abccc --> {a:1,b:2,c:3}
# aabcb --> {a:2,b:2,c:1}

# uncommon [e,e,d,d,d]
def checkAnagrams(s1,s2):
    if len(s1) != len(s2):
        return False
    if sorted(s1) == sorted(s2):
        return True
    return False

checkAnagrams('abc','bca')



def charDeletionsAnagrams(s1,s2):
    # to select all uncommon characters - characters occuring
    uncommon = []
    for i in s1:
        if i not in s2:
            uncommon.append(i)
    for i in s2:
        if i not in s1:
            uncommon.append(i)
    count = len(uncommon)

    # freqs1  -> Frequency of common charcters in s1
    # freqs2  -> Frequency of common charcters in s2

    freqs1 = {}
    freqs2 = {}

    # unique characters in s1 and s2
    uniqs1 = []
    uniqs2 = []

    # Frequency of unique characters in s1
    for i in s1:
        if i not in uncommon and i not in uniqs1:
            freqs1[i] = s1.count(i)
            uniqs1.append(i)
    # Frequency of unique characters in s2
    for i in s2:
        if i not in uncommon and i not in uniqs2:
            freqs2[i] = s2.count(i)
            uniqs2.append(i)
    # Difference in frequencies for common characters and add the difference
    for key in freqs1.keys():
```

```
57          count += abs(freqs1[key] - freqs2[key])
58      return count
59
60  charDeletionsAnagrams('aabbcc','ccbbaaa')
61
```

Out[13]: 1

In [15]:

```
1   #Function to give average range
2
3   def averageRange(lb,ub):
4       sum = 0
5       for i in range(lb,ub+1):
6           sum += i
7       count = ub-lb+1
8       return sum//count
9
10  averageRange(1000,123456)
11
```

Out[15]: 62228

In [22]:

```
# Frequency of a word exam problem 2

#{a:4,g:9,i:6,p:213,c:6}
#[4,6,6,9,213]
#[213,9,6,6,4]
# [a,c,g,i,p]
#k=3
#li=[]
#for item in d.items():
    #if item[1] == 6:
       # li.append(item[0])

#li = [i,c]


def kLargestFrequency(s,k):
    # Construct the frequency dictionary for all unique characters
    #unique = []
    freq = {}
    for i in s:
        if i not in freq.keys():
            freq[i] = s.count(i)

    #Extract  unique frequencies in descending order
    values = sorted(freq.values(),reverse = True)
    uniquevalues = list(set(values))
    uniquevalues = sorted(uniquevalues, reverse = True)

    #identify kth largest frequency
    if k < len(freq.keys()):
        kvalue = uniquevalues[k-1]
    else:
        return -1
    #get all elements with kth largest frequency
    li = []
    for item in freq.items():
        if item[1] == kvalue:
            li.append(item[0])
    #Minimum of kth largest frequency
    return min(li)

kLargestFrequency('aabcdcc',3)

```

Out[22]: 'b'

In [16]:
```
def searchContact(name):
    filename='DataFiles/contacts.txt'
    if contactExists(name):
        with open(filename,'r') as f:
            for i in f:
                i=i.split(',')
                if i[0]==name:
                    print(i[0],i[1],i[2])
        return

searchContact("name2")
```

In [3]:
```
def listContacts():
    with open('DataFiles/contacts.txt','r') as f:
        x=f.read().split()
        if len(x) != 0:
            print(i[0],i[1],i[2])
        else:
            print('No Contact exists')
    return
listContacts()

```

In [ ]:
```

```