In [6]:
```
def uniquedata(li):
    unique=[]
    for element in li:


        if element not in unique:
            unique.append(element)
    return unique
li=[1,2,3,3,2,1]
uniquedata(li)
```

Out[6]:
```
[1, 2, 3]
```

In [11]:
```
def unique(li):
    u=[]
    for i in li:
        if i not in u:
            u.append(i)
    return u
li=[1,2,3,3,2,1,4,5,6,6,5,4]
unique(li)

```

Out[11]:
```
[1, 2, 3, 4, 5, 6]
```

In [ ]:
```
def Average(li):
    unique = []
    c,sum =0,0
    for i in li:
        if i not in unique:
            unique.append(i)
    l=list(filter(lambda i: len(i)==3,unique))
    for i in l:
        sum+=int(i)
        c+=1
    print(sum//c)
#li=[100 200 100 200 300 400]
li=input().split()
Average(li)
```

In [22]:
```
list(filter(lambda i:len(i)==3,['1','2','3','4','100','200']))
```

Out[22]:
```
['100', '200']
```

In [ ]:
```
list(filter(lambda x:(x%2!=0),[1,2,3,4,5,6]))
```

In [ ]:
```
def Division(li):
    unique=[]
    for i in range(len(li)-1):
        for j in range(i+1,len(li)):
            d=abs(int(li[i])-int(li[j]))
            if d not in unique:
                unique.append(d)
    print(min(unique))
Division (li)
t=int(input())
for i in range(1,t+1):
    li=input().split()
    Division(li)
```

In [28]:
```
def urls(li):
    t=int(input())
    unique =[]
    u1 =[]
    for i in range(t):
        li=input()
        unique.append(li)
    for i in unique:
         if i not in u1:
            u1.append(i)
    print(len(u1))
    for i in u1:
        print(i,end='\n')
urls(li)
```

```
5
www.google.com
www.yaahoo.com
www.google.com
www.amazon.com
www.rgukt.com
4
www.google.com
www.yaahoo.com
www.amazon.com
www.rgukt.com
```

In [5]:
```
# Function to write encryption
keyfile='Data/key.txt'
def dictionarykeyfile(keyfile):
    key={}
    with open(keyfile,'r') as f:
        for line in f:

            line=line.split()
            key[line[0]]=line[1]
    return key

#dictionarykeyfile(keyfile)
def encryptmarksdata(datafile,keyfile):
    key=dictionarykeyfile(keyfile)
    with open(datafile,'r') as f:
        filedata=f.read().split('\n')
    with open('Data/encryptedmarks.txt','w') as f:
        for mark in filedata:
            line=' '
            for n in mark:
                line=line+key[n]
            f.write(line+'\n')
    return
datafile='Data/marks.txt'
import timeit
st=timeit.default_timer()
decryptmarksdata(datafile,keyfile)

encryptmarksdata(datafile,keyfile)
```

In [ ]:
```
def decryptmarksdata(encryptedfile,keyfile):
    key=dictionarykeyfile(keyfile)
    newkey={}
    for i,j in key.items():
        newkey[j]=i
    with open(encryptedfile,'r') as f:
        encrypteddata=f.read().split('\n')
    with open('Data/decryptedmarks.txt','w') as f:
        for encryptedmark in encrypteddata:
            line=' '
            for n in encryptedmark:
                line +=newkey[n]
            f.write(line +'\n')
    return
encryptedfile='Data/encryptedmarks.txt'
keyfile='Data/key.txt'
decryptmarksdata(encryptedfile,keyfile)

```

## Comphrehensios

In [22]:
```
keyfile='Data/key.txt'
key=dictionarykeyfile(keyfile)

#evenkeys={item for item in key.items() if int(item[0])%2==0}
evenkeys={item for item in key}
evenkeys
```

Out[22]:
```
{'0', '1', '2', '3', '4', '5', '6', '7', '8', '9'}
```

### Numpy Library

- Processing N-Dimensional array

In [2]:
```
import numpy as np
li=[1,2,3,'z']
a=np.array(li)
a
type(a)
a
```

Out[2]:
```
array(['1', '2', '3', 'z'], dtype='<U11')
```

In [3]:
```
li=[1,2,3,'z']
a=np.array(li)
b=np.arange(15)
b
```

Out[3]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14])
```

In [4]:
```
rn=np.random.randint(0,100,size=10)
rn
```

Out[4]:
```
array([67,  1, 94, 13, 74, 91, 13, 61, 80, 24])
```

In [21]:
```
rn=np.random.randint(6,size=(3,3))
rn
```

Out[21]:
```
array([[4, 5, 4],
       [4, 5, 4],
       [0, 4, 5]])
```

In [13]:
```
rn[2][1]
```

Out[13]:
```
0
```

In [22]:
```
rn.ndim
```

Out[22]:
```
2
```

In [23]:
```
rn.size
```

Out[23]:
```
9
```

In [24]:
```
1 rn.shape
```

Out[24]:
```
(3, 3)
```

In [25]:
```
1 rn.itemsize
```

Out[25]:
```
4
```

In [26]:
```
1 rn.nbytes
```

Out[26]:
```
36
```

In [27]:
```
1 rn[:2,:3]
```

Out[27]:
```
array([[4, 5, 4],
       [4, 5, 4]])
```

In [30]:
```
1 rn[:3,::2]    #skip
```

Out[30]:
```
array([[4, 5],
       [4, 5],
       [0, 4]])
```

In [31]:
```
1 rn[::-1,::-1]   # Tranpose
```

Out[31]:
```
array([[5, 4, 0],
       [4, 5, 4],
       [4, 5, 4]])
```

In [35]:
```
1 print(rn)
```

```
[[4 5 4]
 [4 5 4]
 [0 4 5]]
```

In [36]:
```
1 print(b)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
```

In [43]:
```
1 c=b.reshape(5,3)
2 c
```

Out[43]:
```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11],
       [12, 13, 14]])
```

In [44]:
```
d=c+1
d
```

Out[44]:
```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8,  9],
       [10, 11, 12],
       [13, 14, 15]])
```

In [51]:
```
import numpy as np
m=np.ones((3,3))
print("matrix m:\n",m)
```

```
matrix m:
 [[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

## Pandas

- Usecases:
  - it is name of library
  - it is used for data analysis,data cleaning,data transformation
- Notations
  - series
  - Data Frames

In [5]:
```
import pandas as pd

internal1={'s1':21,'s2':18,'s3':34}
internal1=pd.Series(internal1)
internal2={'s1':15,'s2':18,'s3':12}
internal2=pd.Series(internal2)
internal2

```

Out[5]:
```
s1    15
s2    18
s3    12
dtype: int64
```

In [6]:
```
final={'Internal1':internal1,'Internal2':internal2}
final=pd.DataFrame(final)
final
```

Out[6]:

| | Internal1 | Internal2 |
|---|---|---|
| **s1** | 21 | 15 |
| **s2** | 18 | 18 |
| **s3** | 34 | 12 |

In [7]:
```
final={'Internal1':internal1,'Internal2':internal2}
final=pd.DataFrame(final)
final['Internal1']
```

Out[7]:
```
s1    21
s2    18
s3    34
Name: Internal1, dtype: int64
```

In [8]:
```
final.columns   # Name of all columns
```

Out[8]:
```
Index(['Internal1', 'Internal2'], dtype='object')
```

In [9]:
```
final.values   # List of all rows
```

Out[9]:
```
array([[21, 15],
       [18, 18],
       [34, 12]], dtype=int64)
```

In [10]:
```
final.values[2]
```

Out[10]:
```
array([34, 12], dtype=int64)
```

In [11]:
```
final.values[2,0]
#final.values[2][0]
```

Out[11]:
```
34
```

In [14]:
```
for row in final.values:
    print('Internal1- ', row[0],'Internal2 -',row[1])
```

```
Internal1-  21 Internal2 - 15
Internal1-  18 Internal2 - 18
Internal1-  34 Internal2 - 12
```

In [18]:
```
final.loc['s4']=[20,46]
final.drop(3)
```

Out[18]:

| | Internal1 | Internal2 |
|---|---|---|
| **s1** | 21 | 15 |
| **s2** | 18 | 18 |
| **s3** | 34 | 12 |
| **s4** | 20 | 46 |

In [22]:
```
final.values[2]=[12,25]
final.drop(3)
```

In [ ]:
```

```