Todays's objective

- Functions in python
- Python Data Strauctre
  - Lists
  - Tuples
  - sets
  - Dictionaries

In [ ]:
```
# Functions in python
#Collection of statements
#code reusblity
    #* 4 Types
    #1.A function with arg[] and with retrun value
    #2.A function with arg[] and without  retrun value
    #3.A function without arg[] and with retrun value
    #4.A function without arg[] and without retrun value
```

In [4]:
```
#1.A function with arg[] and with retrun value
    #syntax of funtion
    #arg1=input()
    #arg2=input()
        #def fun_name(arg1,arg2):
            #...stams
           # return arg1+arg2
        #fun_name(arg1,arg2)
#Funtion to addition two number
def addition(a,b):
    return a+b
a=int(input("Enter the a value:"))
b=int(input("Enter the a value:"))
addition(a,b)



```

```
Enter the a value:4
Enter the a value:6
```

Out[4]:
```
10
```

In [2]:
```
addition(89,33)
```

Out[2]:
```
122
```

In [5]:

```python
#2.A function with arg[] and without retrun value
def subtration(a,b):
    print(a-b)
subtration(23,6)


```

```
17
```

In [7]:

```python
#3.A function without arg[] and with retrun value
a=6
b=3
def multiplition():
    return a*b
multiplition()
```

Out[7]:

```
18
```

In [10]:

```python
#4.A function without arg[] and without retrun value
def floorDivision():
    print(a//b)
a=int(input())
b=int(input())
floorDivision()
```

```
5
2
2
```

In [14]:

```python
def sqrt():
     return 2**3
sqrt()

```

Out[14]:

```
8
```

In [18]:
```python
#task 1:
    #Funtion to create a Calculator app
#     Explanation:
#         userchoice:1-->addition
#                    2-->sub
#                    3-->sqrt
#                    4-->floor Division
#                    5-->Exit
#                    --->Invalid option
def calApp():
    while True:
        print("1.Add\n2.sub\n3.Sqrt\n4.FloorDiv\n5.Exit")
        uc=int(input("Enter the Choice.."))
        if uc==1:
            addition(2,4)
        elif uc==2:
            subtration(4,2)
        elif uc==3:
            sqrt()
        elif uc==4:
            floorDivision()
        elif uc==5:
            return True
calApp()


```

```
1.Add
2.sub
3.Sqrt
4.FloorDiv
5.Exit
Enter the Choice..1
1.Add
2.sub
3.Sqrt
4.FloorDiv
5.Exit
Enter the Choice..2
2
1.Add
2.sub
3.Sqrt
4.FloorDiv
5.Exit
Enter the Choice..4
2
1.Add
2.sub
3.Sqrt
4.FloorDiv
5.Exit
Enter the Choice..5
```

Out[18]:
```
True
```

In [1]:
```python
# prime Number
# Funtion with arguments and with return value
def isprime(n):
    count=0
    for i in range(1,n+1):
        if(n%i==0):
            count+=1
    if(count==2):
        return True
    else:
        return False
n=int(input())
if(isprime(n)):
    print("it is prime ")
else:
    print("not a prime")
```

```
10
not a prime
```

In [26]:
```python
# input:1 100
# output:2 5 7 11 .... 97

# n1=int(input())
# n2=int(input())
def prime_series(n1,n2):
    while n1<=n2:
        if(isprime(n1)):
            print(n1,end=" ")
        n1=n1+1
```

In [27]:
```python
prime_series(1,10)
```

```
2 3 5 7
```

In [28]:
```python
prime_series(10,20)
```

```
11 13 17 19
```

In [29]:
```python
prime_series(20,30)
```

```
23 29
```

- List in python
    - Collection of elements like..chr,int,spaces,spl ch,str..etc
    - can reprasents []
    - It is mutable
        - we add elements in list
        - we can remove
    - It contains duplicates data
    - list can reprasents-list(data type)

In [34]:
```
#List
t=[1,4,2,5,2,1]
print(t)
print(type(t))
```

```
[1, 4, 2, 5, 2, 1]
<class 'list'>
```

In [36]:
```
print(dir(list),end=" ")
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__dir_
_', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem
__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass
__', '__iter__', '__le__', '__len__', '__lt__', '__mul__', '__ne__', '__new__',
'__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setat
tr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'c
lear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'revers
e', 'sort']
```

In [62]:
```
#for add new ele
t.append(8)
t.count(8)

```

Out[62]: 7

In [39]:
```
t[4:]
```

Out[39]: [2, 1, 8]

In [40]:
```
len(t)
```

Out[40]: 7

In [41]:
```
sum(t)
```

Out[41]: 23

In [42]:
```
#min
min(t)
```

Out[42]: 1

In [43]:
```
max(t)
```

Out[43]: 8

In [44]:
```
#avg
len(t)//2
```

Out[44]: 3

In [45]:
```
t
```
Out[45]:
```
[1, 4, 2, 5, 2, 1, 8]
```

In [46]:
```
#lastv ele remove
t.pop()
```
Out[46]:
```
8
```

In [47]:
```
#remove
t.remove(5)
```

In [48]:
```
t
```
Out[48]:
```
[1, 4, 2, 2, 1]
```

In [51]:
```
t2=[6,7,8]
t=t2.copy()
t
```
Out[51]:
```
[6, 7, 8]
```

In [74]:
```
#Strngs
s=["ap","cbit","vijay","muni","lokesh","sai"]
s.sort()
s.reverse()
print(s)
```
```
['vijay', 'sai', 'muni', 'lokesh', 'cbit', 'ap']
```

In [ ]:
```

```

In [55]:
```
t.sort()
t
```
Out[55]:
```
[6, 7, 8]
```

In [71]:
```
#Sort the elements based on length
s

```
Out[71]:
```
['vijay', 'sai', 'muni', 'lokesh', 'cbitvbit', 'ap']
```

In [72]:
```
sorted(s)
```
Out[72]:
```
['ap', 'cbitvbit', 'lokesh', 'muni', 'sai', 'vijay']
```

In [75]:
```
print(sorted(s,key=len))
```
```
['ap', 'sai', 'muni', 'cbit', 'vijay', 'lokesh']
```

In [76]:
```
t
```
Out[76]:
```
[6, 7, 8, 8, 8, 8, 8, 8, 8]
```

In [77]:
```
t.remove(8)
t
```
Out[77]:
```
[6, 7, 8, 8, 8, 8, 8, 8]
```

In [78]:
```
#remove the duplicate ele
el=[]
for ele in t:
    if ele not in el:
        el.append(ele)
print(el)
```
```
[6, 7, 8]
```

In [90]:
```
#Find the 3rd largest number in the list
li=[3,4,1,7,2,89,23,55,58]
li2=sorted(li)
li2
```
Out[90]:
```
[1, 2, 3, 4, 7, 23, 55, 58, 89]
```

In [89]:
```
li2[-3]
```
Out[89]:
```
23
```

In [93]:
```
#insert
li.insert(4,"cbit")
li
```
Out[93]:
```
[3, 4, 1, 7, 'cbit', 345, '345', 2, 89, 23, 55, 58]
```

Tuple

- it also contain list of elements
  - reprasented as tuple and symbol as ()
  - it duplicates
  - it is immutable (doesn't modify)

In [105]:
```
t1=(1,3,4,6,3)
type(t1)
print(t1)
```
```
(1, 3, 4, 6, 3)
```

In [96]:
```
t1.count(3)
```
Out[96]:
```
1
```

In [99]:
```
sum(t1)//2
min(t1)
```
Out[99]:
```
1
```

In [101]:
```
t1.insert(2,7)
```
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-101-119959e33e47> in <module>
----> 1 t1.insert(2,7)

AttributeError: 'tuple' object has no attribute 'insert'
```

In [102]:
```
t1.remove(3)
```
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-102-8e4f1a7376f5> in <module>
----> 1 t1.remove(3)

AttributeError: 'tuple' object has no attribute 'remove'
```

In [103]:
```
dir(tuple)
```
Out[103]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
```

In [104]:
```
t1[:3]
```
Out[104]:
```
(1, 3, 4)
```

Sets

- doesn't duplicates data

- reprasentes as like {}
- data type is set

In [115]:
```
s={1,3,3,4,2,5,1}
s
```

Out[115]:
```
{1, 2, 3, 4, 5}
```

In [107]:
```
dir(set)
```

Out[107]:
```
['__and__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__gt__',
 '__hash__',
 '__iand__',
 '__init__',
 '__init_subclass__',
 '__ior__',
 '__isub__',
 '__iter__',
 '__ixor__',
 '__le__',
 '__len__',
 '__lt__',
 '__ne__',
 '__new__',
 '__or__',
 '__rand__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__ror__',
 '__rsub__',
 '__rxor__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__sub__',
 '__subclasshook__',
 '__xor__',
 'add',
 'clear',
 'copy',
 'difference',
 'difference_update',
 'discard',
 'intersection',
 'intersection_update',
 'isdisjoint',
 'issubset',
 'issuperset',
 'pop',
 'remove',
 'symmetric_difference',
 'symmetric_difference_update',
 'union',
 'update']
```

In [113]:
```
s
```
Out[113]:
```
{9, 10, 56}
```

In [109]:
```
#removes the first ele
s.pop()
```
Out[109]:
```
1
```

In [110]:
```
s.add(8)#new elem.. adding
s
```
Out[110]:
```
{2, 3, 4, 5, 8}
```

In [112]:
```
s1={9,56,10}
s=s1.copy()
s
```
Out[112]:
```
{9, 10, 56}
```

In [114]:
```
s
```
Out[114]:
```
{9, 10, 56}
```

In [116]:
```
s
```
Out[116]:
```
{1, 2, 3, 4, 5}
```

In [117]:
```
s1
```
Out[117]:
```
{9, 10, 56}
```

In [118]:
```
s.update(s1)
s
```
Out[118]:
```
{1, 2, 3, 4, 5, 9, 10, 56}
```

In [134]:
```
st1={10,20,30,40,500}
st2={50,40,20,100,200}
```

In [136]:
```
st1.difference(st2)
st2.difference(st1)
```
Out[136]:
```
{50, 100, 200}
```

In [125]:
```
st1.intersection(st2)
```
Out[125]:
```
{20, 40}
```

In [126]:
```
st1.union(st2)
```
Out[126]:
```
{10, 20, 30, 40, 50, 100, 200}
```

In [130]:
```
st1.difference_update(st2)
```

In [131]:
```
st1
```

Out[131]:
```
{10, 30, 500}
```

In [133]:
```
st1.discard(30)
st1
```

Out[133]:
```
{10, 500}
```

In [ ]:
```

```

In [142]:
```
# scanf("%d%d%d%d",&a,&b,&c&d)
data=list(input().split())
print(data)
```

```
1 2 3 4 5 6 6 7 8
['1', '2', '3', '4', '5', '6', '6', '7', '8']
```

In [143]:
```
data2=list(map(int,data))
data2
```

Out[143]:
```
[1, 2, 3, 4, 5, 6, 6, 7, 8]
```

In [145]:
```
data3=list(map(str,data2))
data3
```

Out[145]:
```
['1', '2', '3', '4', '5', '6', '6', '7', '8']
```

In [148]:
```
v1='2'
v2=list(map(int,v1))
print(v2)
```

```
[2]
```

In [154]:
```
s="apssdc cbit vijayawada ap"
print(list(s))
s2="vijay"
print(list(s2))
print(list(s.split()))
```

```
['a', 'p', 's', 's', 'd', 'c', ' ', 'c', 'b', 'i', 't', ' ', 'v', 'i', 'j',
'a', 'y', 'a', 'w', 'a', 'd', 'a', ' ', 'a', 'p']
['v', 'i', 'j', 'a', 'y']
['apssdc', 'cbit', 'vijayawada', 'ap']
```

In [157]:
```
# Addition of two Number
# input:10 20
# output:Addition of 10 and 20 is 30
```

In [161]:
```
i_data=input().split()
print(i_data,i_data[0],i_data[1],int(i_data[0])+int(i_data[1]))

```

```
10 20
['10', '20'] 10 20 30
```

In [2]:
```
#primenumbers in the list
isprime(56)
```

Out[2]:
```
False
```

In [3]:
```
prime_list=[]
def generate_prime(ub,lb):
    while(ub<=lb):
        if(isprime(ub)):
            prime_list.append(ub)
        ub=ub+1
generate_prime(1,100)
print(prime_list)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 7
3, 79, 83, 89, 97]
```

In [5]:
```
print(prime_list)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 7
3, 79, 83, 89, 97]
```

In [15]:
```
data=str(prime_list[5])# getting value from primlist
d_data=list(data)#separating value into digits in form of char
i_data=list(map(int,d_data))# converting char into int
s_sum=sum(i_data)# calculating Sum
s_sum# printing sum value
```

Out[15]:
```
4
```

In [16]:
```
s_sum=sum(list(map(int,list(str(prime_list[5])))))
s_sum
```

Out[16]:
```
4
```

In [19]:
```
final_data=[]
for ele in prime_list:
    s_sum=sum(list(map(int,list(str(ele)))))
    final_data.append(s_sum)
print(final_data)
#     print(s_sum,end=" ")
```

```
[2, 3, 5, 7, 2, 4, 8, 10, 5, 11, 4, 10, 5, 7, 11, 8, 14, 7, 13, 8, 10, 16, 11,
17, 16]
```

In [20]:
```
print(prime_list)
print(final_data)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 7
3, 79, 83, 89, 97]
[2, 3, 5, 7, 2, 4, 8, 10, 5, 11, 4, 10, 5, 7, 11, 8, 14, 7, 13, 8, 10, 16, 11,
17, 16]
```

In [26]:
```
# 2 2 3 3 5 5 7 7 11 2 13 2 17 8 19 10 ....
data=[]
pt=0
ft=0
for i in range(len(prime_list)):
    if(i%2==0):
        data.append(prime_list[pt])
        pt=pt+1
    if(i%2==1):
        data.append(final_data[ft])
        ft=ft+1
print(data)
```

```
[2, 2, 3, 3, 5, 5, 7, 7, 11, 2, 13, 4, 17, 8, 19, 10, 23, 5, 29, 11, 31, 4, 37,
10, 41]
```

In [28]:
```
# input:1 100
# ouput:2 2 3 4 5 5 7 7 11 2 13 4 17 8....
```

In [30]:
```
# to check given number prime or not
# to generate prime numbers b/w two ranges
# to calculatre each prime value seperated  digitssum
# even position primenumbers
#      - odd digitsum values
#print(final output)
```

In [36]:
```python
# li=[1,3,5,56,7,12,3,6,8,9,3,5,6,6,3,1,2,3,51,1]
li="AAAABBBBCCCCDDDD"
li=list(li)
u_data=[]
for ele in li:
    if (ele not in u_data):
        u_data.append(ele)
freq_ele=[]
for u_ele in u_data:
    c=li.count(u_ele)
    freq_ele.append(c)
print(u_data)
print(freq_ele)
for i in range(len(u_data)):
    print(u_data[i],"--->",freq_ele[i])
```

```
['A', 'B', 'C', 'D']
[4, 4, 4, 4]
A ---> 4
B ---> 4
C ---> 4
D ---> 4
```

In [74]:
```python
li=[1,2,4,56,17,29,49,6,23,55,54,57,87,63,5,7]
li=sorted(li)
print(li)
middle=len(li)//2
mid=len(li)//2
print(mid)
print(li[middle])
# input:[1,2,4,56,17,29,49,6,23,55,54,57,87,63,5,7]
# output1:1,2,4,5,6, 7, 17, 23, 29,87,63,57,56,55,54,49
# output2:23,17,7,6,5,4,3,2,1,29,49, 54, 55, 56, 57, 63, 87
# output3:23,17,7,6,5,4,3,2,1,29,87,63,57,56,55,54,49
```

```
[1, 2, 4, 5, 6, 7, 17, 23, 29, 49, 54, 55, 56, 57, 63, 87]
8
29
```

In [84]:
```python
r_values=li[middle+1:]
l_values=li[:middle]
print(l_values)
print(r_values)
l=list(sorted(l_values,reverse=True)+r_values)
l.insert(middle,li[middle])
print(l)
```

```
[1, 2, 4, 5, 6, 7, 17, 23]
[49, 54, 55, 56, 57, 63, 87]
[23, 17, 7, 6, 5, 4, 2, 1, 29, 49, 54, 55, 56, 57, 63, 87]
```

In [77]:
```
l_values=li[:middle+1]
print(r_values)
print(l_values)

```
```
[49, 54, 55, 56, 57, 63, 87]
[1, 2, 4, 5, 6, 7, 17, 23, 29]
```

In [68]:
```
r_values=li[middle+1:]
l_values=li[:middle]
print(l_values)
print(r_values)
li.insert(8,8)
print(sorted(l_values,reverse=True)+sorted(r_values,reverse=False))
```
```
[1, 2, 4, 5, 6, 7, 17, 23]
[8, 29, 49, 54, 55, 56, 57, 63, 87]
[23, 17, 7, 6, 5, 4, 2, 1, 8, 29, 49, 54, 55, 56, 57, 63, 87]
```

In [85]:
```
## input 227 331 492 196 142
li=[227,331,492,196,142]
for i in li:
    if(isprime(i)):
        print(i)
```
```
227
331
```

In [ ]:
```
Task :
    input:["APPLE","BANANA","GRAPES","ORANGE","PINAPPLE","MANGO"]
    ouput:"APPLE"-->"ELPPA
           "BANANA"-->"ANANAB"
            "GRAPES"-->"SEPARG"
            ...
```

In [98]:
```
fruits=["APPLE","BANANA","GRAPES","ORANGE","PINAPPLE","MANGO"]

```

In [99]:
```
fruits
```
Out[99]:
```
['APPLE', 'BANANA', 'GRAPES', 'ORANGE', 'PINAPPLE', 'MANGO']
```

In [ ]:
```

```

In [104]:
```
em=[ ]
for everyfruit in fruits:
    em.append(everyfruit[::-1])
for i in range(len(fruits)):
    print(fruits[i],"--->",em[i])
```

```
APPLE ---> ELPPA
BANANA ---> ANANAB
GRAPES ---> SEPARG
ORANGE ---> EGNARO
PINAPPLE ---> ELPPANIP
MANGO ---> OGNAM
```

In [94]:
```
li[1][::-1]
```

Out[94]:
```
'ananab'
```

In [92]:
```
s=li[1]
s
```

Out[92]:
```
'banana'
```

In [93]:
```
s[::-1]
```

Out[93]:
```
'ananab'
```

In [9]:
```
n=int(input())
for i in range(1,n+1):
    n1=int(input())
    n2=int(input())
    n3=int(input())
    print(n1-(n2*n3))
```

```
3
1
2
3
-5
4
5
6
-26
6
7
8
-50
```

6

#####