Functions and its Types:

```
# what is a functions
def fun(name,age=25): # position,default value,keyword,var
iable length
    statements
    ...
    ...
    ...
    return value
fun(25,'c')
fun('c',25)
fun('c')
fun('c',10,30,40)
```

In [5]:

```
# without return value and without argument
def fun():
    print("Anits College")
fun()
```

```
Anits College
```

In [6]:

```
# without return value and with argument
def fun(a):
    print(a)
fun('Anits College')
```

...

In [7]:

```
# with return value and without argument
def fun():
    a = 'Anits College'
    return a
fun()
```

...

In [8]:

```
# with return value and argument
def fun(a):
    return a
fun('Anits College')
```

...

In [3]:

```
# write a python code using args and return value to find leap
def isLeapyear(y):
    if(y%400==0) or (y%4==0 and y%100!=0):
        return True
    return False
isLeapyear(1900)
```

Out[3]:

```
False
```

In [4]:

```
# print the nanoseconds of the year, consider leap year (366 a
def nanos(y):
    if(isLeapyear(y)):
        print(366*24*60*60*100*1000)
    else:
        print(365*24*60*60*100*1000)
i = int(input())
nanos(i)
```

```
1990
3153600000000
```

In [3]:

```
# print whether the given input is prime or not(input should a
def isPrime(n):
    count= 0
    for i in range(1,n+1):
        if (n%i==0):
            count+=1
    if(count==2):
        return True
    return False
i = int(input())
isPrime(i)
```

...

In [4]:

```
#print prime numbers in the given range(input dynamic)
def printp(n):
    for i in range(n):
        if(isPrime(i)):
            print(i)
i = int(input())
printp(i)
```

...

Regular Expressions

In [5]:

```
#^[9][1][6-9][0-9]{9}$|^[6-9][0-9]{9}|^[0][6-9][0-9]{9}
import re
i = int(input())
p = '^[9][1][6-9][0-9]{9}$|^[6-9][0-9]{9}|^[0][6-9][0-9]{9}'
if (re.match(p,str(i))):
    print("valid Number")
else:
    print('invalid Number')
```

```
9876543210
valid Number
```

In [6]:

```
# write the regex code for email validation
```

In [11]:

```
import re
p = "This is anits college.Python is the workshop"
if(re.search('Python',p)):
    print('find')
else:
    print('Not Available')
```

```
find
```

Data Structures in Python:

Similar to Collections simply call it as iterators

- List
- Tuple

- Set and
- Dictionary

Lists:

- Collections of heterogenoes(Different)type of data elements
- It can be defined as [] and can be typecasted by list() method.
- It can changes the values, Slicing is done due to index
- Ordered Format of data because of index

In [1]:

```
1 print(dir(list))
```

```
['__add__', '__class__', '__contains__', '__delattr_
_', '__delitem__', '__dir__', '__doc__', '__eq__',
'__format__', '__ge__', '__getattribute__', '__getit
em__', '__gt__', '__hash__', '__iadd__', '__imul__',
'__init__', '__init_subclass__', '__iter__', '__le_
_', '__len__', '__lt__', '__mul__', '__ne__', '__new
__', '__reduce__', '__reduce_ex__', '__repr__', '__r
eversed__', '__rmul__', '__setattr__', '__setitem_
_', '__sizeof__', '__str__', '__subclasshook__', 'ap
pend', 'clear', 'copy', 'count', 'extend', 'index',
'insert', 'pop', 'remove', 'reverse', 'sort']
```

In [4]:

```
p = [2,5,6,12,1,3]
print(p)
print(type(p))
print(p[3])
print(p[2:5])
print(p[1:4:-1])
print(p[1:4:1])
print(p[4:1:-1])
print(p[-1:-4:-1])
print(p[-1:-4:1])
print(p[-4:-1:-1])
print(p[-4:-1:1])
```

```
[2, 5, 6, 12, 1, 3]
<class 'list'>
12
[6, 12, 1]
[]
[5, 6, 12]
[1, 12, 6]
[3, 1, 12]
[]
[]
[6, 12, 1]
```

In [5]:

```
print(p)
p.append(int(100))
print(p)
```

```
[2, 5, 6, 12, 1, 3]
[2, 5, 6, 12, 1, 3, 100]
```

In [11]:

```
k=p.copy()
print(k)
print(p)
k.append(int(0))
print(k)
print(p)
p.append(int(234))
print(k)
print(p)
```

```
[2, 5, 6, 12, 1, 3, 100]
[2, 5, 6, 12, 1, 3, 100]
[2, 5, 6, 12, 1, 3, 100, 0]
[2, 5, 6, 12, 1, 3, 100]
[2, 5, 6, 12, 1, 3, 100, 0]
[2, 5, 6, 12, 1, 3, 100, 234]
```

In [14]:

```
print(p.count(500))
print(p.count(2))
```

```
0
1
```

In [17]:

```
print(p)
print(k)
c=list(k.extend(p))
print(c)
print(k)
print(p)
```

```
[2, 5, 6, 12, 1, 3, 100, 234]
[2, 5, 6, 12, 1, 3, 100, 0, 2, 5, 6, 12, 1, 3, 100,
234, 2, 5, 6, 12, 1, 3, 100, 234]

---------------------------------------------------
----------------------
TypeError                                 Traceback
 (most recent call last)
<ipython-input-17-c39e0a4d16d3> in <module>
      1 print(p)
      2 print(k)
----> 3 c=list(k.extend(p))
      4 print(c)
      5 print(k)

TypeError: 'NoneType' object is not iterable
```

In [18]:

```
c=k+p
print(c)
print(k)
print(p)
```

```
[2, 5, 6, 12, 1, 3, 100, 0, 2, 5, 6, 12, 1, 3, 100,
234, 2, 5, 6, 12, 1, 3, 100, 234, 2, 5, 6, 12, 1, 3,
100, 234, 2, 5, 6, 12, 1, 3, 100, 234]
[2, 5, 6, 12, 1, 3, 100, 0, 2, 5, 6, 12, 1, 3, 100,
234, 2, 5, 6, 12, 1, 3, 100, 234, 2, 5, 6, 12, 1, 3,
100, 234]
[2, 5, 6, 12, 1, 3, 100, 234]
```

In [22]:

```
print(p)
print(p.index(234))
```

```
[2, 5, 6, 12, 1, 3, 100, 234]
7
```

In [25]:

```
print(p)
p.insert(0,1)
print(p)
p.insert(9,99)
print(p)
```

```
[1, 2, 5, 6, 12, 1, 3, 100, 234]
[1, 1, 2, 5, 6, 12, 1, 3, 100, 234]
[1, 1, 2, 5, 6, 12, 1, 3, 100, 99, 234]
```

In [26]:

```
print(p)
p.pop(1)
print(p)
```

```
[1, 1, 2, 5, 6, 12, 1, 3, 100, 99, 234]
[1, 2, 5, 6, 12, 1, 3, 100, 99, 234]
```

In [30]:

```
print(p)
p.pop(5)
print(p)
```

```
[1, 2, 5, 6, 12, 100, 99, 234]
[1, 2, 5, 6, 12, 99, 234]
```

In [31]:

```
print(p)
p.remove(6)
print(p)
```

```
[1, 2, 5, 6, 12, 99, 234]
[1, 2, 5, 12, 99, 234]
```

In [32]:

```
print(p)
p.remove(100)
print(p)
```

```
[1, 2, 5, 12, 99, 234]

---------------------------------------------------
-----------------------
ValueError                                Traceback
 (most recent call last)
<ipython-input-32-e8b46125d762> in <module>
      1 print(p)
----> 2 p.remove(100)
      3 print(p)

ValueError: list.remove(x): x not in list
```

In [33]:

```
print(p)
p.reverse()
print(p)
```

```
[1, 2, 5, 12, 99, 234]
[234, 99, 12, 5, 2, 1]
```

In [34]:

```
print(p)
p.sort()
print(p)
```

```
[234, 99, 12, 5, 2, 1]
[1, 2, 5, 12, 99, 234]
```

In [35]:

```
print(p)
p.sort(reverse=True)
print(p)
```

```
[1, 2, 5, 12, 99, 234]
[234, 99, 12, 5, 2, 1]
```

In [36]:

```
print(p)
p.clear()
print(p)
```

```
[234, 99, 12, 5, 2, 1]
[]
```

In [37]:

```
1 print(p)
2 del p
3 print(p)
```

[]

```
---------------------------------------------------------
-----------------------
NameError                                 Traceback
 (most recent call last)
<ipython-input-37-7519e7043e9d> in <module>
      1 print(p)
      2 del p
----> 3 print(p)

NameError: name 'p' is not defined
```

Input:

23 34 56 23 56 raju kiran

Output:

23 23 34 56 56

kiran raju

In [50]:

```
n = input().split()
nol = []
stl = []
for i in n:
    if (str(i).isdigit()):
        nol.append(int(i))
    else:
        stl.append(i)
nol.sort()
print(nol)
print(stl)
```

```
23 34 56 34 56 kiran raju
[23, 34, 34, 56, 56]
['kiran', 'raju']
```

Tuple:

- Collection of heterogenous data type elements
- It can be defined as () and typecasting as tuple()
- It can't change the values, Slicing can be done
- Based on index the slicing is done and it is also ordered data

In [52]:

```
d = (23,34,34.00,'kiran')
print(d)
print(type(d))
print(d[2])
print(d[1:5])
print(d[0:3:1])
```

```
(23, 34, 34.0, 'kiran')
<class 'tuple'>
34.0
(34, 34.0, 'kiran')
(23, 34, 34.0)
```

In [53]:

```
print(dir(tuple))
```

```
['__add__', '__class__', '__contains__', '__delattr_
_', '__dir__', '__doc__', '__eq__', '__format__', '_
_ge__', '__getattribute__', '__getitem__', '__getnew
args__', '__gt__', '__hash__', '__init__', '__init_s
ubclass__', '__iter__', '__le__', '__len__', '__lt_
_', '__mul__', '__ne__', '__new__', '__reduce__', '_
_reduce_ex__', '__repr__', '__rmul__', '__setattr_
_', '__sizeof__', '__str__', '__subclasshook__', 'co
unt', 'index']
```

In [54]:

```
print(d)
print(d.count(1))
print(d.count(34))
```

```
(23, 34, 34.0, 'kiran')
0
2
```

In [57]:

```
print(d)
print(d.index(34))
print(d.index(2))
```

```
(23, 34, 34.0, 'kiran')
1

---------------------------------------------------------
----------------------
ValueError                                Traceback
 (most recent call last)
<ipython-input-57-5c25933fc3e0> in <module>
      1 print(d)
      2 print(d.index(34))
----> 3 print(d.index(2))

ValueError: tuple.index(x): x not in tuple
```

In [77]:

```
p=[23,45,32,23.45,100,2]
t=(45,23.45,32.50,100,2,32)
# print(p.index(54))
# print(t.index(54))
print(t.index(32))
```

```
5
```

In [78]:

```
d =[2,4,1,34,56.0,23.78]
z =(2,4,1,34,56.0,23.78)
print(d.index(34))
print(z.index(34))
```

```
3
3
```

Sets:

- Collection of heterogenous data elements
- It can be defined as {} and typecaste as set()
- It can change the value but it doesn't supports the slicing
- Unordered Format data, index is not available
- Removes duplicate elements

In [85]:

```
g = {34,3,34,3,3,3,3,3,3,3,3,3,3,334.564,'rajesh',23,'somu',3}
print(g)
```

```
{34, 3, 'somu', 334.564, 'rajesh', 23}
```

In [86]:

```
g
```

Out[86]:

```
{23, 3, 334.564, 34, 'rajesh', 'somu'}
```

In [87]:

```
print(dir(set))
```

```
['__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', 'int
ersection_update', 'isdisjoint', 'issubset', 'issupe
rset', 'pop', 'remove', 'symmetric_difference', 'sym
metric_difference_update', 'union', 'update']
```

In [88]:

```
print(g)
g.add(56)
print(g)
```

```
{34, 3, 'somu', 334.564, 'rajesh', 23}
{34, 3, 'somu', 334.564, 'rajesh', 23, 56}
```

In [ ]:

```

```