Tuples

In [10]:
```
## Declaration
## Tuples are immutable they cant change their value
t=(1,2,3,'abc',3.5)
print(t)
t[3:5]
```

```
(1, 2, 3, 'abc', 3.5)
```

Out[10]:
```
('abc', 3.5)
```

In [12]:
```
t1=((1,2,3),(4,5,6))
print(t1[0][2])
```

```
3
```

In [ ]:
```
# format()
# capitalize()
# isalnum()
# isalpha()
# islower()
# isupper()
# lower()
# upper()
# isnumeric()
# count(value,start,stop)
# find(value,start,stop) ## If it doesnt exits in string returns -1
# index(value,start,stop)## If it doesnt exits in string returns error
# split(separator,maxsplit) ## Default space
# strip()## Default space
#       Strip method removes any characters from the begining and the end
# translate()
#   maketrans('what to transform','with what' , 'what to remove')
# """
```

Iteration

In [14]:
```
for i in t:
    print(i)
print()
for i in range(len(t)):
    print(t[i])
```

```
1
2
3
abc
3.5

1
2
3
abc
3.5
```

In [23]:
```
## We can concatenation
t1=(1,2,3,4)
t2=(1,2,3,3)
t=t1+t2
print(t)
```

```
(1, 2, 3, 4, 1, 2, 3, 3)
```

In [24]:
```
## comparing Two tuples
print(t1<t2)
```

```
False
```

In [33]:
```
a=(1,2,3)
b=('a','b','c')## String and int can't be compared
print(a<b)
print(ord('#'))## ord returns asci value of charcter
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-33-9a4ffa382ee5> in <module>
      1 a=(1,2,3)
      2 b=('a','b','c')## String and int can't be compared
----> 3 print(a<b)
      4 print(ord('#'))## ord returns asci value of charcter

TypeError: '<' not supported between instances of 'int' and 'str'
```

In [42]:
```
a=(1,2)
b=(0,0,3)
print(a<b)
print(ord('A'))
```

```
False
65
```

In [43]:
```
print('A'<'65')
```

```
False
```

Tuple Method

In [49]:
```
# 1)count()
# 2)index()

help(tuple.count)

help(str.count)
help(tuple.index)
```

```
Help on method_descriptor:

count(self, value, /)
    Return number of occurrences of value.

Help on method_descriptor:

count(...)
    S.count(sub[, start[, end]]) -> int

    Return the number of non-overlapping occurrences of substring sub in
    string S[start:end].  Optional arguments start and end are
    interpreted as in slice notation.

Help on method_descriptor:

index(self, value, start=0, stop=9223372036854775807, /)
    Return first index of value.

    Raises ValueError if the value is not present.
```

In [78]:
```python
s="abcaaa"
print(s.count('a'))
t=('a','t','t','t')
t[1]=0
print(t)
print(t.count('t'))
```

```
4

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-78-f00564e5b543> in <module>
      2 print(s.count('a'))
      3 t=('a','t','t','t')
----> 4 t[1]=0
      5 print(t)
      6 print(t.count('t'))

TypeError: 'tuple' object does not support item assignment
```

In [62]:
```python
help(tuple)
# underscore vada functiomns magic functions che ene call na karva pade e
```

```
Help on class tuple in module builtins:

class tuple(object)
 |  tuple(iterable=(), /)
 |
 |  Built-in immutable sequence.
 |
 |  If no argument is given, the constructor returns an empty tuple.
 |  If iterable is specified the tuple is initialized from iterable's item
s.
 |
 |  If the argument is a tuple, the return value is the same object.
 |
 |  Built-in subclasses:
 |      asyncgen_hooks
 |      UnraisableHookArgs
 |
 |  Methods defined here:
 |
```

In [63]:
```
help(str)
```

```
Help on class str in module builtins:

class str(object)
 |  str(object='') -> str
 |  str(bytes_or_buffer[, encoding[, errors]]) -> str
 |
 |  Create a new string object from the given object. If encoding or
 |  errors is specified, then the object must expose a data buffer
 |  that will be decoded using the given encoding and error handler.
 |  Otherwise, returns the result of object.__str__() (if defined)
 |  or repr(object).
 |  encoding defaults to sys.getdefaultencoding().
 |  errors defaults to 'strict'.
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
```

In [66]:
```
tup=(1,2,3)
del(tup)

```

In [71]:
```
t1=((1,2,3),(3,4))
t=t1+tuple('abc')
print(t)
```

```
((1, 2, 3), (3, 4), 'a', 'b', 'c')
```

In [72]:
```
"""WAP to find the sum of odd and even numbers in a tuple"""
t=(1,2,3,4,5,6,7,8,9,10)
even_sum=odd_sum=0
for i in range(len(t)):
    if t[i]%2==0:
        even_sum+=t[i]
    else:
        odd_sum+=t[i]
print("EVEN",even_sum)
print("ODD",odd_sum)
```

```
EVEN 30
ODD 25
```

UNIT-5 MUTABLE DATA STRUCTURES

Lists

In [77]:
```
# lists are exactly like array
# lists are mutable l[]=empty list

# 1.They are ordered index=0,1 or -1 -2 etc
# 2.Slicing possible
# 3.Also allows duplicates just like tuples
# 4.MUTABLE we can change a particular value in list

l=[1,2,3,'abc',True,3.5]
print(l)
l[2]=5
print(l)
li=list((1,2,3,4))
print(li)
```

```
[1, 2, 3, 'abc', True, 3.5]
[1, 2, 5, 'abc', True, 3.5]
[1, 2, 3, 4]
```

In [ ]:
```
Lists Methods/Functions

1.append()
2.clear()
3.copy()
4.count()
5.index() # Returns error
6.extend() # we can add tuples also and added as individual element
7.insert(pos,elmnt) pos=position elmnt=element
8.remove(index)
9.reverse() # index ma element ni value pass karvani reverses the list [l.
10.sort(reverse,key) #reverse-dec ma sort karva Key-used to sort it using
11.pop(pos) # Position apya vagar pop karisu to last element remove thase
```

In [82]:
```
l=[1,2,3]
l.append(4)
print(l)
l.append([1,2,3,6,7,8])
print(l)
```

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

In [4]:
```
l1=[1,2,3]
print(l1)
l1.clear()
print(l1)
l3=[]
print(l3)
```

```
[1, 2, 3]
[]
[]
```

In [87]:
```
l=[1,2,3]
l2=l[:] # here [:] means slice change nai thai
l3=l.copy()
l[1]=4
print(l)
print(l2)
print(l3)
```

```
[1, 4, 3]
[1, 2, 3]
[1, 2, 3]
```

In [127]:
```
l=[1,2,3,1,2,3,4,1,1,1]
l.count(1)
```

Out[127]:
```
5
```

In [93]:
```
l.index(4)
```

Out[93]:
```
6
```

In [120]:
```
l=[1,2,3]
l.extend([4]) #agar khali 4 add kaarie emnem then nai thai extend but appe
l.append(5)
print(l)
```

```
[1, 2, 3, 4, 5]
```

In [121]:
```

l.insert(1,0)
print(l)

```

```
[1, 0, 2, 3, 4, 5]
```

In [126]:
```
a=l.pop()
print(a)
print(l)
l.remove(1)
print(l)
```

```
2
[1]
[]
```

In [138]:
```
l=[1,2,3,1,2,3,4]
print(l[::-1])
l1=[2,3,5,4]
l1.reverse()
print(l1)
```

```
[4, 3, 2, 1, 3, 2, 1]
[4, 5, 3, 2]
```

In [84]:

```
help(list)
```

```
Help on class list in module builtins:

class list(object)
 |  list(iterable=(), /)
 |
 |  Built-in mutable sequence.
 |
 |  If no argument is given, the constructor creates a new empty list.
 |  The argument must be an iterable if specified.
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return key in self.
 |
 |  __delitem__(self, key, /)
```

In [8]:

```
# """WAP to accept i and j from the user as matrix rows and colums and wap
# i=int(input("Enter number of rows :"))
# j=int(input("Enter number of colums :"))
# l1=[]
# l2=[]
# print("enter elements of MAT -1")
# for a in range(i*j):
#     x=int(input("Enter  element of matrix 1 :"))
#     l1.append(x)

# print("enter elements of MAT -2")
# for a in range(i*j):
#     x=int(input("Enter  element of matrix 1 :"))
#     l2.append(x)
# print("ADDED matrix")
# l3=[]
# for i in range(len(l1)):
#     x=l1[i]+l2[i]
#     l3.append(x)

# print("ADDED: ",l3)

AA 1-D MA THAYO MATRIX NAI BANYO AA


```

```
Enter number of rows :2
Enter number of colums :2
enter elements of MAT -1
Enter  element of matrix 1 :1
Enter  element of matrix 1 :2
Enter  element of matrix 1 :3
Enter  element of matrix 1 :4
enter elements of MAT -2
Enter  element of matrix 1 :1
Enter  element of matrix 1 :2
Enter  element of matrix 1 :3
Enter  element of matrix 1 :4
ADDED matrix
ADDED:  [2, 4, 6, 8]
```

In [18]:

```python
r=int(input("Enter number of rows :"))
c=int(input("Enter number of colums :"))
mat1=[]
mat2=[]
print("MAT-1")
for i in range(r):
    mat1.append([])
    for j in range(c):
        el=int(input("Insert element "))
        mat1[i].append(el)
print("MAT-2")
for i in range(r):
    mat2.append([])
    for j in range(c):
        el=int(input("Insert element "))
        mat2[i].append(el)
mat3=[]
for i in range(r):
    mat3.append([])
    for j in range(c):
        el=mat1[i][j]+mat2[i][j]
        mat3[i].append(el)


print(mat1)
print(mat2)
print(mat3)
for i in range(r):
    print()
    for j in range(c):
        print(mat3[i][j],end="  ")
```

```
Enter number of rows :2
Enter number of colums :2
MAT-1
Insert element 1
Insert element 2
Insert element 3
Insert element 4
MAT-2
Insert element 1
Insert element 2
Insert element 3
Insert element 4
[[1, 2], [3, 4]]
[[1, 2], [3, 4]]
[[2, 4], [6, 8]]

2  4
6  8
```

In [1]:
```
l=[4,2,5,6,1,3]
l.sort()
print(l)
```

```
[1, 2, 3, 4, 5, 6]
```

In [3]:
```
l=[1,5,9,7,4,2,3]
l.sort(reverse=True)
print(l)
```

```
[9, 7, 5, 4, 3, 2, 1]
```

In [5]:
```
cars=['Volvo','kia','Hyundai','BMW','Audi','VM']
# Length wise sort karva Func banai ne key ma value apvani
def fun(e):
    return len(e)
cars.sort(reverse=True,key=fun)
print(cars)
```

```
['Hyundai', 'Volvo', 'Audi', 'kia', 'BMW', 'VM']
```

List Comprehension

In [ ]:
```

```

In [7]:
```
fruits=['apple','bananas','cherry','kiwi','mango']
newlist=[]
for x in fruits:
    if 'a' in x:
        newlist.append(x)
print(newlist)
```

```
['apple', 'bananas', 'mango']
```

In [8]:
```
newlist=[x for x in fruits if 'a' in x] #[x-variable for x in fruits-Loop
print(newlist)
```

```
['apple', 'bananas', 'mango']
```

Data Structures and Dictionaries

In [ ]:
```
1. Key value pairs
Dictionary-ordered since cersion 3.7


Functions:
    1.
    2.
    3.
```

In [15]:
```
D={'a':123,'b':245,'c':567,'a':222}
d=dict(name='john',age=35,height=6.4)
print(D)
print(d)
D['b']=333
print(D)
```

```
{'a': 222, 'b': 245, 'c': 567}
{'name': 'john', 'age': 35, 'height': 6.4}
{'a': 222, 'b': 333, 'c': 567}
```

In [90]:
```
help(dict)
```

```
Help on class dict in module builtins:

class dict(object)
 |  dict() -> new empty dictionary
 |  dict(mapping) -> new dictionary initialized from a mapping object's
 |      (key, value) pairs
 |  dict(iterable) -> new dictionary initialized as if via:
 |      d = {}
 |      for k, v in iterable:
 |          d[k] = v
 |  dict(**kwargs) -> new dictionary initialized with the name=value pairs
 |      in the keyword argument list.  For example:  dict(one=1, two=2)
 |
 |  Built-in subclasses:
 |      StgDict
 |
 |  Methods defined here:
 |
 |  __contains__(self, key, /)
```

In [19]:
```
## Iteration
D={'a':123,'b':245,'c':567,'a':222}
for i in D:
    print(i)
    print(D[i])
D={'a':[10,20,30,40],'b':{'a':20,'d':30,'c':40}}
print(D['a'][1])
print(D['b']['d'])
```

```
a
222
b
245
c
567
20
30
```

In [35]:
```
s="This is a string. It shows that this string is iterable."
s=s.lower()
print(s)
mytable=s.maketrans('','','.,?,!')
s=s.translate(mytable)
print(s)
a=s.split(" ")
print(a)
word_count={}
for word in s.split():
    if word in word_count:
        word_count[word]+=1
    else:
        word_count[word]=1
print(word_count)





```

```
this is a string. it shows that this string is iterable.
this is a string it shows that this string is iterable
['this', 'is', 'a', 'string', 'it', 'shows', 'that', 'this', 'string', 'is',
'iterable']
{'this': 2, 'is': 2, 'a': 1, 'string': 2, 'it': 1, 'shows': 1, 'that': 1, 'it
erable': 1}
```

In [39]:
```
D={'a':20,'b':30,'c':40}
print(D)
D.clear()
print(D)
```

```
{'a': 20, 'b': 30, 'c': 40}
{}
```

In [44]:
```
D={'a':20,'b':30,'c':40}
a=D.copy()
print(a)
```
```
{'a': 20, 'b': 30, 'c': 40}
```

In [75]:
```
D={'a':20,'b':30,'c':40}
s=D.get('m',50) # returns none when key not foun
print(s)
print(D)
```
```
50
{'a': 20, 'b': 30, 'c': 40}
```

In [85]:
```
help(dict.values)
```
```
Help on method_descriptor:

values(...)
    D.values() -> an object providing a view on D's values
```

In [50]:
```
D={'a':20,'b':30,'c':40}
D.keys()#a set-like object providing a view on D's keys
```
Out[50]:
```
dict_keys(['a', 'b', 'c'])
```

In [57]:
```
D={'a':20,'b':30,'c':40}
D.pop('F')# if key not found then keyError Occurs
print(D)
```
```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-57-ad7968ef6b37> in <module>
      1 D={'a':20,'b':30,'c':40}
----> 2 D.pop('F')
      3 print(D)

KeyError: 'F'
```

In [87]:
```
D={'a':20,'b':30,'c':40}
D.popitem()# if item passed then TypeError Ocuurs
```
Out[87]:
```
('c', 40)
```

In [89]:
```
D={'a':20,'b':30,'c':40}
z=D.setdefault('f',60)
print(z)
print(D)
```
```
60
{'a': 20, 'b': 30, 'c': 40, 'f': 60}
```

In [84]:
```
D={'a':20,'b':30,'c':40}
D.update({1:'maths'})
print(D)
```

```
{'a': 20, 'b': 30, 'c': 40, 1: 'maths'}
```

In [86]:
```
D={'a':20,'b':30,'c':40}
D.values()
```

Out[86]:
```
dict_values([20, 30, 40])
```

In [91]:
```
D={'a':20,'b':30,'c':40}
x=D.values()
D['a']=30
print(D)
print(x)
```

```
{'a': 30, 'b': 30, 'c': 40}
dict_values([30, 30, 40])
```

In [8]:

```
l=[]

while True:
    e=input('Enter a number or stop :')
    if e=='stop':
        break
    else:
        l.append(e)
print(l)
D={}
for i in l:
    if i not in D:
        D[i]=[i]
        #D[i].append(i)
    else:
        D[i].append(i)
    print(D)
l1=list(D.items())
def func(t):
    return len(t[1])
l1.sort(reverse=True,key=func)
d1=dict(l1)
print(d1)
```

```
Enter a number or stop :2
Enter a number or stop :4
Enter a number or stop :2
Enter a number or stop :4
Enter a number or stop :5
Enter a number or stop :3
Enter a number or stop :1
Enter a number or stop :2
Enter a number or stop :5
Enter a number or stop :6
Enter a number or stop :4
Enter a number or stop :stop
['2', '4', '2', '4', '5', '3', '1', '2', '5', '6', '4']
{'2': ['2']}
{'2': ['2'], '4': ['4']}
{'2': ['2', '2'], '4': ['4']}
{'2': ['2', '2'], '4': ['4', '4']}
{'2': ['2', '2'], '4': ['4', '4'], '5': ['5']}
{'2': ['2', '2'], '4': ['4', '4'], '5': ['5'], '3': ['3']}
{'2': ['2', '2'], '4': ['4', '4'], '5': ['5'], '3': ['3'], '1': ['1']}
{'2': ['2', '2', '2'], '4': ['4', '4'], '5': ['5'], '3': ['3'], '1': ['1']}
{'2': ['2', '2', '2'], '4': ['4', '4'], '5': ['5', '5'], '3': ['3'], '1':
['1']}
{'2': ['2', '2', '2'], '4': ['4', '4'], '5': ['5', '5'], '3': ['3'], '1':
['1'], '6': ['6']}
{'2': ['2', '2', '2'], '4': ['4', '4', '4'], '5': ['5', '5'], '3': ['3'],
'1': ['1'], '6': ['6']}
{'2': ['2', '2', '2'], '4': ['4', '4', '4'], '5': ['5', '5'], '3': ['3'],
'1': ['1'], '6': ['6']}
```

SETS

In [ ]:
```
# Duplication not allowed
# Unchangable
# Unordered
# Half muttable
# Frozenset: u cant add or remove completely immutable
```

In [17]:
```
s={1,3,5,6,4,7,8,9}
print(s)
s[0]=5
se=set((1,2,3,4,5))
se
```

```
{1, 3, 4, 5, 6, 7, 8, 9}

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-17-80a7823aa450> in <module>
      1 s={1,3,5,6,4,7,8,9}
      2 print(s)
----> 3 s[0]=5
      4 se=set((1,2,3,4,5))
      5 se

TypeError: 'set' object does not support item assignment
```

In [10]:
```
myset=frozenset([1,3,4,5])
myset.add(5)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-10-dbae248fea9d> in <module>
      1 myset=frozenset([1,3,4,5])
----> 2 myset.add(5)

AttributeError: 'frozenset' object has no attribute 'add'
```

In [69]:
```
x={1,2,3}
y={1,2,3}
print(x.issubset(y))
print(y.issuperset(x))
print(x.union(y))
print(x.intersection(y))
print(y.difference(x))
print(y.symmetric_difference(x))
print(x.copy())
x.add(1)
y.remove(4)
print(x)
print(y)
```

```
True
True
{1, 2, 3}
{1, 2, 3}
set()
set()
{1, 2, 3}

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-69-e4aa15658c79> in <module>
      9 print(x.copy())
     10 x.add(1)
---> 11 y.remove(4)
     12 print(x)
     13 print(y)

KeyError: 4
```

In [23]:
```
help(set)
```

```
Help on class set in module builtins:

class set(object)
 |  set() -> new empty set object
 |  set(iterable) -> new set object
 |
 |  Build an unordered collection of unique elements.
 |
 |  Methods defined here:
 |
 |  __and__(self, value, /)
 |      Return self&value.
 |
 |  __contains__(...)
 |      x.__contains__(y) <==> y in x.
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
```

In [41]:
```
# WAP to check if a string contains all unique characters.
s='aabcd'
x=set((s))
if len(x)==len(s):
    print("All are unique")
else:
    print("NOO")
print(x)
```

```
NOO
{'d', 'a', 'b', 'c'}
```

In [ ]:
```
317 WAP to change the first half of the String into upper case
320 wap to check if 2 strings are balanced or not S1 and S2 are balanced
if all the characters of s1 are s2 charcters podition
doent matter
325 WAp to shift decimal digits  n places to the left wrapping the extra d
grater than num of digits then rev the string
```

In [68]:
```
s='Jeel'
b=tuple(s)
print(b)
a=len(s)
c=(a//2)
q=[]
for i in range (0,c):
    q.append(b[i].upper())
print(q)
for i in range(c,len(s)):
     q.append(b[i])
print(q)
qw=''
for i in range (len(s)):
    qw+=(q[i])
print(qw)

```

```
('J', 'e', 'e', 'l')
['J', 'E']
['J', 'E', 'e', 'l']
JEel
```

In [80]:
```
s1='jeel'
s2='pala'
set1=set(s1)
set2=set(s2)
s=''
if len(s1)==len(s2):
    print("balanced")
else:

    print("Not balance")
#         for i in range(len(s1)):
#             q=set1.difference(set2)
#             print(q)
#             s+=q[i]




```

```
balanced
```

In [ ]:
```
325 WAp to shift decimal digits  n places to the left wrapping the extra d
grater than num of digits then rev the string
```

In [ ]:

```
# 327 : min 8 char
#     atleast one from a-to z
#     atleast one from A to Z
#     atleast one from 0-9
#     one special char _ or @ or $


while True:

    flag_li=False
    flag_d=flag_u=flag_l=flag_sp=False

    pwd=input("Enter the Pass  : ")
    if len(pwd)>=8:
        flag_li=True


    for i in range(len(pwd)):
        if pwd[i]=='@' or pwd[i]=='_' or pwd[i]=='$':
            flag_sp=True
        if pwd[i].isdigit()==True:
            flag_d=True
        if pwd[i].isupper()==True:
            flag_u=True
        if pwd[i].islower()==True:
            flag_l=True
#     print(flag_li)
#     print(flag_d)
#     print(flag_u)
#     print(flag_l)
#     print(flag_sp)
    if flag_li and flag_d and flag_u and flag_l and flag_sp:

        print("Pass Verified")
    else:
          print("Enter valid pass")
```

```
Enter the Pass  : abcABC123@
True
True
True
True
True
Pass Verified
Enter the Pass  : ghjkhljk
True
False
False
True
False
Enter valid pass
```

Built in Function For Data Structures In Python

In [2]:
```
1. len()
2.min()
3.max()
4.enumerate()
5.reversed()
6.sorted()

```

In [56]:
```
l=[4,1,3,2]
d={'def':123,'abc':456}
t=(4,2,3,1)
S={4,2,3,4,1}
s='hello'
print(min(l))
print(min(d))# Keys ma compare karse  weather aplha or num
print(min(t))
print(min(S))
print(min(s))
print(max(l))
print(max(d))
print(max(t))
print(max(S))
print(max(s))
print(list(enumerate(s)))## All items will be assigned a number and should
print(list(reversed(s)))# Sets are not reverible tuples,set  and list are
#Sorted works without type cast
print(sorted(l))
print(sorted(t))
print(dict(sorted(d.items())))
print(sorted(s))
print(sorted(S))

```

```
1
abc
1
1
e
4
def
4
4
o
[(0, 'h'), (1, 'e'), (2, 'l'), (3, 'l'), (4, 'o')]
['o', 'l', 'l', 'e', 'h']
[1, 2, 3, 4]
[1, 2, 3, 4]
{'abc': 456, 'def': 123}
['e', 'h', 'l', 'l', 'o']
[1, 2, 3, 4]
```

In [68]:
```python
# Useing sorted Function sort dictionary in desc order
d={'zxy':654,'abc':123}
# d.items()
def func(t):
    return t[1]
print(sorted(d.items(),reverse=True,key=func))


```

```
[('zxy', 654), ('abc', 123)]
```

In [64]:
```python
help(sorted)
```

```
Help on built-in function sorted in module builtins:

sorted(iterable, /, *, key=None, reverse=False)
    Return a new list containing all items from the iterable in ascending ord
er.

    A custom key function can be supplied to customize the sort order, and th
e
    reverse flag can be set to request the result in descending order.
```

In [ ]:
```python
# WAP to remove duplicates from a list without using sets
l=[1,2,3,4,1,2,3,5,6,1]
def func(a):
    a.sort()
    print(a)
    for i in range(len(a)):
        for j in range(len(a)):
            if i+1==len(a) or j+1==len(a):
                break
            else:
                if a[i]==a[j+1]:
                    a.remove(a[i])
                else:
                    pass
func(l)
print(l)
```

Lambda Function

Anonymous Function

In [82]:
```python
x=lambda a:a+10
print(x(5))
```

```
15
```

In [84]:
```python
x=lambda a,b,c:a+b+c
print(x(1,2,3))
```

```
6
```

In [87]:
```python
def func(n):
    return lambda a:a*n
doubler=func(2)
print(doubler(11))
tripler=func(3)
print(tripler(11))
```

```
22
33
```

In [89]:
```python
d={'apples':50,'banans':60,'mangoes':55}
print(sorted(d.items(),reverse=False,key=lambda t:t[1]))
```

```
[('apples', 50), ('mangoes', 55), ('banans', 60)]
```

Map Function

In [92]:
```python
# # Map Function returns a map object which is an iterator of the results
# function to each item of the given iterator
# map(func,iter)
# It is a function which map passes to each element of the give iterable
# Iter:is the iterable which is to be mapped
# NOTE : You can pass more than 1 iterable


# WAP to double all the numbers in a listusing map and Lembda

l=[2,3,7,4,1]
result=map(lambda x:x*x ,l)
print(list(result))
```

```
[4, 9, 49, 16, 1]
```

In [94]:
```python
# WAP to addd 2 list using map and lambda
x=[1,2,3]
y=[4,5,6]
result=map(lambda a,b:a+b,x,y)
print(list(result))
```

```
[5, 7, 9]
```

In [100]:
```python
l=['sat','mat','cat']
r=map(list,l)
print(list(r))
```

```
[['s', 'a', 't'], ['m', 'a', 't'], ['c', 'a', 't']]
```

In [111]:
```
# reduce(func,iter)
# It is used to apply a function to iterables and returns a singlle value
import functools
l=[21,23,1]
print(functools.reduce(lambda a,b:a+b,l))
print(functools.reduce(lambda a,b:a if a>b else b,l))
```

```
51
23
```

In [113]:
```
# #  Filter
# Filter method filters a given sequence with the help of a function that
# sequence to be true or false
# filter(func,iter)

def func(c):
    letters=['a','e','i','o','u']
    if c in letters:
        return False
    else:
        return True
print(list(filter(func,['h','e','l','l','N','o'])))
```

```
['h', 'l', 'l', 'N']
```

In [116]:
```
def func(c):
    letters=['0','1','2','3','4','5','6','7','8','9']
    if c in letters:
        return False
    else:
        return True
print(list(filter(func,['h','e','l','l','N','O','0','5'])))
```

```
['h', 'e', 'l', 'l', 'N', 'o']
```

In [118]:
```
def func(c):
    c1=str(c)
    if c.isalpha():
        return True
    else:
        return False
print(list(filter(func,['h','e','l','l','N','O','0','5'])))
```

```
['h', 'e', 'l', 'l', 'N', 'O']
```

In [ ]:
```

```