FUNDAMENTAL CATEGORY DATA TYPE

INT DATA TYPE

```
In [2]: a=10
        print(a)
        print(a,type(a))
```

```
10
10 <class 'int'>
```

id -------which gives address of the variable

```
In [3]: print(a,type(a),id(a))
```

```
10 <class 'int'> 140703472761560
```

INT DATA TYPE

we can also store the different number systems

1) decimal number system 2)binary number system 3)octal number system 4)hexa decimalnumber

binary number system

```
In [8]: a=0b1111
        print(a,type(a),id(a))
```

```
15 <class 'int'> 140703472761720
```

```
In [9]: a=0B10001
        print(a,type(a))
```

```
17 <class 'int'>
```

```
In [11]: a=0B110011
         print(a,type(a),id(a))
```

```
51 <class 'int'> 140703472762872
```

```
In [12]: bin(9)
```

```
Out[12]: '0b1001'
```

```
In [13]: bin(122)
```

```
Out[13]: '0b1111010'
```

```
In [14]: bin(2356)
```

```
Out[14]: '0b100100110100'
```

octal number system

```
In [16]: a=0o122                  ## octal to decimal conversion
         print(a,type(a))
```

```
82 <class 'int'>
```

```
In [17]: a=0o1456
         print(a,type(a))
```

```
814 <class 'int'>
```

```
In [18]: #a=0o1287
         #print(a)
```

```
  Cell In[18], line 1
    a=0o1287
          ^
SyntaxError: invalid digit '8' in octal literal
```

```
In [19]: oct(26)  ## decimal to octal conversion
```

```
Out[19]: '0o32'
```

```
In [20]: oct(98)
```

```
Out[20]: '0o142'
```

```
In [21]: oct(256)
```

```
Out[21]: '0o400'
```

hexa decimal number system

```
In [23]: a=0xac             ##hexa decimal to decimal conversion
         print(a,type(a))
```

```
172 <class 'int'>
```

```
In [24]: a=0xbee
         print(a,type(a))
```

```
3054 <class 'int'>
```

```
In [26]: a=0xFACE
         print(a,type(a))
```

```
64206 <class 'int'>
```

```
In [27]: a=0x1B
         print(a,type(a))
```

```
27 <class 'int'>
```

```
In [29]: hex(15)     # decimal conversion to hexa decimal
```

```
Out[29]: '0xf'
```

```
In [30]: hex(16)
```

```
Out[30]: '0x10'
```

```
In [31]: hex(20)

Out[31]: '0x14'
```

```
In [33]: hex(26)

Out[33]: '0x1a'
```

base conversion calculator

binary to octal/hexa decimal

```
In [37]: a=0b1110
         b=oct(a)
         print(b)

0o16
```

```
In [38]: a=0b1100
         b=oct(a)
         print(b)

0o14
```

```
In [40]: a=0b1100
         b=hex(a)
         print(b,type(b))

0xc <class 'str'>
```

```
In [41]: a=0b1001
         b=hex(a)
         print(b,type(b))

0x9 <class 'str'>
```

octal to binary /hexa decimal

In [42]:

```
a=0o123
b=bin(a)
print(b,type(b))
```

```
0b1010011 <class 'str'>
```

In [43]:

```
a=0o167
b=bin(a)
print(b,type(b))
```

```
0b1110111 <class 'str'>
```

In [44]:

```
a=0o123
b=hex(a)
print(b,type(b))
```

```
0x53 <class 'str'>
```

In [45]:

```
a=0o1245
b=hex(a)
print(b,type(b))
```

```
0x2a5 <class 'str'>
```

hexa decimal to octal / binary

In [47]:

```
a=0xa12
b=oct(a)
print(b,type(b))
```

```
0o5022 <class 'str'>
```

In [48]:

```
a=0xAB
b=oct(a)
print(b,type(b))
```

```
0o253 <class 'str'>
```

In [49]:

```
a=0xA
b=bin(a)
print(b,type(b))
```

```
0b1010 <class 'str'>
```

In [50]:

```
a=0xBEE
b=bin(a)
print(b,type(b))
```

```
0b101111101110 <class 'str'>
```

In [ ]:

```

```

FLOAT DATA TYPE

In [52]:

```
a=1.2
print(a,type(a))
```

```
1.2 <class 'float'>
```

In [53]:

```
a=2.3
b=3.2
c=a+b
print(a,type(a))
print(b,type(b))
print(c,type(c))
```

```
2.3 <class 'float'>
3.2 <class 'float'>
5.5 <class 'float'>
```

In [54]:

```
a=3e2
print(a)
```

```
300.0
```

In [55]:

```
a=4e-2
print(a)
```

```
0.04
```

bool data type

In [58]:
```python
a=False
a
```

Out[58]:
```
False
```

In [59]:
```python
print(True-3)
```

```
-2
```

In [61]:
```python
print(2+True+3)
```

```
6
```

In [62]:
```python
print(3-False+True)
```

```
4
```

In [63]:
```python
print(True*4+1)
```

```
5
```

In [64]:
```python
print(0b1111-True)
```

```
14
```

In [65]:
```python
print(0xf+True)
```

```
16
```

In [66]:
```python
print(False/True)
```

```
0.0
```

In [67]:
```python
#print(False/False)
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[67], line 1
----> 1 print(False/False)

ZeroDivisionError: division by zero
```

In [68]:
```python
#print(True/False)
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[68], line 1
----> 1 print(True/False)

ZeroDivisionError: division by zero
```

complex data type

In [69]:
```
a=2+3j
print(a,type(a))
```

```
(2+3j) <class 'complex'>
```

In [70]:
```
b=2-3j
print(a,type(a))
```

```
(2+3j) <class 'complex'>
```

In [71]:
```
a=2-3.5j
print(a)
```

```
(2-3.5j)
```

In [72]:
```
c=-1.5-1.4j
print(c)
```

```
(-1.5-1.4j)
```

In [75]:
```
a=3.5j
print(a)
```

```
3.5j
```

In [77]:
```
a=-1.5-1.6j
print(a,type(a))
print(a.imag)
print(a.real)
```

```
(-1.5-1.6j) <class 'complex'>
-1.6
-1.5
```

In [78]:

```
a=3.5j
print(a.imag)
print(a.real)
```

```
3.5
0.0
```

SEQUENCE CATEGORY DATA TYPE

Str DATA TYPE

OPERATION ON Str data

two types

1. indexing
2. slicing

In [80]:

```
s="PYTHON"
```

In [81]:

```
s[4]
```

Out[81]:

```
'O'
```

TURPLE

EMPTY TURPLE()

In [2]:

```
varname=()
```

In [3]:

```
varname=tuple()
```

Non-EMPTY TURPLE()

syntax

In [4]:

```
var1=(19,10,23,"True",True)
print(var1,type(var1),id(var1))
```

```
(19, 10, 23, 'True', True) <class 'tuple'> 2000676930880
```

In [5]:

```
var1=23,12,True,False,"HYD",22.7
print(var1,type(var1),id(var1))
```

```
(23, 12, True, False, 'HYD', 22.7) <class 'tuple'> 2000656189280
```

In [8]:

```
var1=tuple("str")
print(var1,type(var1),id(var1))
```

```
('s', 't', 'r') <class 'tuple'> 2000683186560
```

In [12]:

```
var1=tuple([12,True,False])
print(var1,type(var1),id(var1))
```

```
(12, True, False) <class 'tuple'> 2000677093504
```

In [13]:

```
var1=tuple([10])
print(var1,type(var1),id(var1))
```

```
(10,) <class 'tuple'> 2000678247664
```

In [15]:

```
var1=tuple((10,))
print(var1,type(var1),id(var1))
```

```
(10,) <class 'tuple'> 2000678509616
```

In [17]:

```
t1=(10,20,30,12,34,89,11,20)
print(t1,type(t1),id(t1))
```

```
(10, 20, 30, 12, 34, 89, 11, 20) <class 'tuple'> 2000678141120
```

```
In [18]: t2=(100,"rs",34.56,True,2+3j)
         print(t2,type(t2),id(t2))

         (100, 'rs', 34.56, True, (2+3j)) <class 'tuple'> 2000683490640
```

```
In [19]: t2[0]

Out[19]: 100
```

```
In [20]: t2[2]

Out[20]: 34.56
```

```
In [21]: t2[0:3]

Out[21]: (100, 'rs', 34.56)
```

```
In [22]: t2[::-1]

Out[22]: ((2+3j), True, 34.56, 'rs', 100)
```

```
In [23]: t2[::2]

Out[23]: (100, 34.56, (2+3j))
```

```
In [24]: t2=(100,"rs",34.56,True,2+3j)
```

```
In [25]: t2[0]

Out[25]: 100
```

```
In [26]: #t2[0]=100    because the tuple is immutable cant update /modify

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[26], line 1
----> 1 t2[0]=100

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

In [28]:
```
t1=()
print(t1,type(t1),id(t1))
```

```
() <class 'tuple'> 140737274140392
```

In [29]:
```
len(t1)
```

Out[29]:
```
0
```

In [30]:
```
t2=tuple()
print(t2,id(t2),type(t2))
```

```
() 140737274140392 <class 'tuple'>
```

In [31]:
```
t2=(100,"rs",34.56,True,2+3j)
print(t2,type(t2),id(t2))
```

```
(100, 'rs', 34.56, True, (2+3j)) <class 'tuple'> 2000683669824
```

In [32]:
```
len(t2)
```

Out[32]:
```
5
```

In [33]:
```
t3=10,20,True,100,False,23.7
print(t3,type(t3),id(t3))
```

```
(10, 20, True, 100, False, 23.7) <class 'tuple'> 2000683428128
```

In [34]:
```
s="PYTHON"
print(s,type(s))
```

```
PYTHON <class 'str'>
```

In [36]:
```
t=tuple(s)
print(t,type(t))
```

```
('P', 'Y', 'T', 'H', 'O', 'N') <class 'tuple'>
```

In [37]:
```
s="PYTHON"
print(s,type(s))
```

```
PYTHON <class 'str'>
```

In [38]:

```
t=tuple([s])
print(t,type(t))
```

```
('PYTHON',) <class 'tuple'>
```

In [42]:

```
x=(10)
print(x,type(x))
```

```
10 <class 'int'>
```

In [43]:

```
x=(10,)
print(x,type(x))
```

```
(10,) <class 'tuple'>
```

In [44]:

```
a=10
print(a,type(a))
```

```
10 <class 'int'>
```

In [45]:

```
t=tuple(a)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[45], line 1
----> 1 t=tuple(a)

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

In [47]:

```
t=tuple([a])
print(t,type(t))
```

```
(10,) <class 'tuple'>
```

In [48]:

```
t=tuple((a,))
print(t,type(t))
```

```
(10,) <class 'tuple'>
```

In [49]:

```
t2=(100,"rs",34.56,True,2+3j)
print(t2,type(t2),id(t2))
```

```
(100, 'rs', 34.56, True, (2+3j)) <class 'tuple'> 2000683881216
```

```
In [50]: t2.index(100)

Out[50]: 0

In [51]: t2.index(20)

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[51], line 1
----> 1 t2.index(20)

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

In [52]: t2.index("rs")

Out[52]: 1

In [53]: t2.count(10)

Out[53]: 0

In [54]: t2.count(100)

Out[54]: 1

In [56]: t1=(10,0,20,10,20,30,0,10)

In [57]: print(t1,type(t1),id(t1))

(10, 0, 20, 10, 20, 30, 0, 10) <class 'tuple'> 2000679616832

In [58]: t1.count(10)

Out[58]: 3

In [59]: t1.count(0)

Out[59]: 2
```

deep copy is possible but not the shallow copy()

```
In [60]: t1=(10,20,30,40,50,10,20)
         print(t1,type(t1),id(t1))
```

```
(10, 20, 30, 40, 50, 10, 20) <class 'tuple'> 2000683424000
```

```
In [62]: t2=t1
```

```
In [63]: print(t2,type(t2))
```

```
(10, 20, 30, 40, 50, 10, 20) <class 'tuple'>
```

```
In [64]: t3=t1
```

```
In [65]: print(t3)
```

```
(10, 20, 30, 40, 50, 10, 20)
```

```
In [66]: t1=(10,20,30,40,50,10,20)
         print(t1,type(t1),id(t1))
```

```
(10, 20, 30, 40, 50, 10, 20) <class 'tuple'> 2000683125632
```

```
In [67]: del t1[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[67], line 1
----> 1 del t1[0]

TypeError: 'tuple' object doesn't support item deletion
```

```
In [68]: del t1[0:1]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[68], line 1
----> 1 del t1[0:1]

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

```
In [69]: del t1
```

```
In [70]: print(t1,type(t1),id(t1))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[70], line 1
----> 1 print(t1,type(t1),id(t1))

NameError: name 't1' is not defined
```

sorted

this function is used to sort the immutable iterable object like tuple,str,bytes etc and give the sorted data in the form of list

```
In [71]: ## syntax:  listobj=sorted(tuple object)
```

```
In [73]: t1=(10,20,30,40,50,10,20)
         print(t1,type(t1),id(t1))
```

```
(10, 20, 30, 40, 50, 10, 20) <class 'tuple'> 2000684135712
```

```
In [75]: x=sorted(t1)
         print(x)
```

```
[10, 10, 20, 20, 30, 40, 50]
```

```
In [76]: t2=t1[::-1]
```

```
In [77]: print(t2,type(t2))
```

```
(20, 10, 50, 40, 30, 20, 10) <class 'tuple'>
```

```
In [78]: t1=(10,20,30,40,50,10,-1,23,67,9,-4,20)
         print(t1,type(t1))
```

```
(10, 20, 30, 40, 50, 10, -1, 23, 67, 9, -4, 20) <class 'tuple'>
```

```
In [80]: x=sorted(t1)
         print(x,type(x))
```

```
[-4, -1, 9, 10, 10, 20, 20, 23, 30, 40, 50, 67] <class 'list'>
```

In [81]:

```python
x=tuple(sorted(t1))
print(x,type(x))
```

```
(-4, -1, 9, 10, 10, 20, 20, 23, 30, 40, 50, 67) <class 'tuple'>
```

In [83]:

```python
x=tuple(sorted(t1)[::-1])
print(x)
```

```
(67, 50, 40, 30, 23, 20, 20, 10, 10, 9, -1, -4)
```

In [84]:

```python
x=tuple(sorted(t1,reverse=True))
print(x)
```

```
(67, 50, 40, 30, 23, 20, 20, 10, 10, 9, -1, -4)
```

In [85]:

```python
t1=(10,20,30,40,50,10,-1,23,67,9,-4,20)
print(t1,type(t1))
```

```
(10, 20, 30, 40, 50, 10, -1, 23, 67, 9, -4, 20) <class 'tuple'>
```

In [87]:

```python
l1=list(t1)
print(l1,type(l1))
```

```
[10, 20, 30, 40, 50, 10, -1, 23, 67, 9, -4, 20] <class 'list'>
```

In [89]:

```python
l1.sort()
print(l1)
```

```
[-4, -1, 9, 10, 10, 20, 20, 23, 30, 40, 50, 67]
```

In [90]:

```python
t1=tuple(l1)
print(t1,type(t1))
```

```
(-4, -1, 9, 10, 10, 20, 20, 23, 30, 40, 50, 67) <class 'tuple'>
```

In [93]:

```python
t2=t1[::-1]
print(t2)
```

```
(67, 50, 40, 30, 23, 20, 20, 10, 10, 9, -1, -4)
```

NESTED OR INNER TUPLE

tuple in tuple (possible)

```
In [94]: t1=(10,"rossum",(14,15,17),(22,77,66),"oucet")
         print(t1,type(t1))
```

```
(10, 'rossum', (14, 15, 17), (22, 77, 66), 'oucet') <class 'tuple'>
```

```
In [95]: t1[1]
```

```
Out[95]: 'rossum'
```

```
In [96]: t1[2][0:2]
```

```
Out[96]: (14, 15)
```

```
In [97]: t1[3][0:1]
```

```
Out[97]: (22,)
```

```
In [98]: t1[-1]
```

```
Out[98]: 'oucet'
```

list in tuple(possible)

list inside and tuple outside

```
In [99]: t1=(10,"rossum",[17,24,18],[77,78,66],"oucet")
         print(t1,type(t1))
```

```
(10, 'rossum', [17, 24, 18], [77, 78, 66], 'oucet') <class 'tuple'>
```

```
In [102… print(t1[2],type(t1[2]))
```

```
[17, 24, 18] <class 'list'>
```

```
In [103… print(t1[3],type(t1[3]))
```

```
[77, 78, 66] <class 'list'>
```

```
In [105…  t1[2].append(20)
```

```
In [106…  t1[3].insert(1,20)
```

```
In [107…  t1[2].append(33)
```

```
In [108…  print(t1)
```

```
(10, 'rossum', [17, 24, 18, 20, 33], [77, 20, 78, 66], 'oucet')
```

```
In [109…  t1[2].sort()
```

```
In [110…  print(t1)
```

```
(10, 'rossum', [17, 18, 20, 24, 33], [77, 20, 78, 66], 'oucet')
```

```
In [111…  t1[3].sort()
```

```
In [112…  print(t1)
```

```
(10, 'rossum', [17, 18, 20, 24, 33], [20, 66, 77, 78], 'oucet')
```

```
In [113…  t1[2].sort(reverse=True)
```

```
In [114…  print(t1)
```

```
(10, 'rossum', [33, 24, 20, 18, 17], [20, 66, 77, 78], 'oucet')
```

tuple in list

tuple inside and list outside

```
In [116…  lst1=[10,"rossum",(12,67,23,77),(77,78,66),"oucet"]
          print(lst1)
```

```
[10, 'rossum', (12, 67, 23, 77), (77, 78, 66), 'oucet']
```

```
In [117…  lst1[1]
```

```
Out[117…  'rossum'
```

```
In [119…  print(lst1[2],type(lst1[2]))
```

```
(12, 67, 23, 77) <class 'tuple'>
```