In [ ]:

```
### Booleans
The two constants True and False are defined .
The usual boolean operators are also available : == ,!= , > ,>= ,< , <=
```

In [13]:

```
a = 12
b = 13
print(a == b - 1, a == b, a != b, a < b, a >= b)
```

```
True False True True False
```

In [15]:

```
a = 4
b = 5
c = a < b
print(a < b , a > b , a >= b , a <= b ,)
print(type(c))
```

```
True False False True
<class 'bool'>
```

In [11]:

```
type(c)
```

Out[11]:

```
bool
```

In [18]:

```
a = True
b = False
type(a)
```

Out[18]:

```
bool
```

In [26]:

```
## Complex
z = 2 + 3j
z
print(type(z))
```

```
<class 'complex'>
```

In [29]:

```
z.conjugate
```

Out[29]:

```
<function complex.conjugate>
```

In [30]:

```
z.conjugate() # here is conjucate is funcation that is apploicable on the object 'z'
```

Out[30]:

```
(2-3j)
```

In [31]:

```
z.imag # here imag is a property
```

Out[31]:

```
3.0
```

In [32]:

```
z.real # here is real is a property of the imaginary numbers
```

Out[32]:

```
2.0
```

In [33]:

```
abs(z) #
```

Out[33]:

```
3.605551275463989
```

In [36]:

```
a = 4
b = 5
a.conjugate()
```

Out[36]:

```
4
```

In [38]:

```
Z = 3 - 4j
abs(z)
```

Out[38]:

```
3.605551275463989
```

In [39]:

```
# Let's Talk about string data types

name = "sunil"
print(name)
type(name)
```

```
sunil
```

Out[39]:

```
str
```

In [43]:

```
Firstname = "sunil kumar"
Lastname = "Thakur"
print(Firstname + Lastname)
print(Firstname +" "+Lastname)
```

```
sunil kumarThakur
sunil kumar Thakur
```

In [44]:

```
Firstname.upper()
```

Out[44]:

```
'SUNIL KUMAR'
```

In [48]:

```
len(Firstname)
```

Out[48]:

```
11
```

In [46]:

```
len(Lastname)
```

Out[46]:

```
6
```

In [55]:

```
len(Firstname),len(Lastname)
```

Out[55]:

```
(11, 6)
```

In [58]:

```
Firstname[2] # Indexing
```

Out[58]:

```
'n'
```

In [59]:

```
Firstname[2:5] # slicing
```

Out[59]:

```
'nil'
```

In [60]:

```
a = 12345 # Numeric data we canote find lenth
a[1] # indexing is not possible on the numbers
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call las
t)
<ipython-input-60-344e891389d5> in <module>
      1 a = 12345
----> 2 len()

TypeError: len() takes exactly one argument (0 given)
```

In [63]:

```
Firstname
```

Out[63]:

```
'sunil kumar'
```

In [66]:

```
Firtname[3] = "n"
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call las
t)
<ipython-input-66-44bb0aa9ec20> in <module>
----> 1 Firtname[3] = "n"

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

In Python , we canote modify the strings that are once .That's why string data types comes under immutable data types

In [67]:

```
players = ("ganguly" ,"sehgwag" , "ganguly" ,"ranatunga" )
print(players)
type(players)
```

```
('ganguly', 'sehgwag', 'ganguly', 'ranatunga')
```

Out[67]:

```
tuple
```

In [72]:

```
len(players) , len("sehgwag")
```

Out[72]:

```
(4, 7)
```

In [73]:

```
players[2]
```

Out[73]:

```
'ganguly'
```

In [74]:

```
players[2] = "dhoni" # Tuples are unmutable here is the instance
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
<ipython-input-74-fe5a4f29bd0e> in <module>
----> 1 players[2] = "dhoni"

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

In [77]:

```
players1 = ("ganguly" ,"sehgwag" , "ganguly" ,"ranatunga" )
players1
type(players1)
```

Out[77]:

```
tuple
```

In [ ]:

```
## Mutable data type in Python
LIST
```

In [81]:

```
actors_list =["Manoj Bajpai" , "Pankaj Tripathi" , "Sushant singh rajput", "Irfankhan"]
type(actors_list)
```

Out[81]:

```
list
```

In [82]:

```
actors_list[2] = "kumar vishwas"
actors_list
```

Out[82]:

```
['Manoj Bajpai', 'Pankaj Tripathi', 'kumar vishwas', 'Irfankhan']
```

In [ ]:

```

```