PHYTON DATA TYPE

23-10-2024

In [ ]:
```
1. varialbe name = value (this is synatax)
```

In [ ]:
```
2. value have data types as below   || PHYTON DATA TYPES ||
```

In [ ]:
```
INTEGER    :The value without decimal like ( 100,560,999,)
```

In [ ]:
```
FLOAT      :value with decmial like (58.64, 89.52, 100.56)
```

In [ ]:
```
BOOLEAN    :True or False
```

In [ ]:
```
COMPLEX    :( a+bj) where a mean real and b means inaginary and j means square r
```

In [ ]:
```
STRING     : Always enter in single , double or multile quotes ('NIT'  or  ''NIT
```

INTEGER DATA TYPE

In [12]:
```
i = 30
i
```

Out[12]:
```
30
```

In [2]:
```
type(i)
```

Out[2]:
```
int
```

In [ ]:
```
1. the type function is used for to know the type of data type.
```

In [ ]:
```
2. we can call it as interger data type or integer value.
```

In [ ]:
```
3. and we can aslo like write as below.
```

In [ ]:
```
4. Integer will never have decimal value.
```

In [4]:
```
print(type(i))
```

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

In [9]:
```
a,b,c = 10,20,30
a,b ,c
```

Out[9]:
```
(10, 20, 30)
```

In [13]:
```
i
```

Out[13]: 30

In [19]:
```
i,i1, i2 =30,  20, 30
i + i1 + i2
```

Out[19]: 80

In [17]:
```
i - i2+ i1
```

Out[17]: 20

In [18]:
```
i - (i2+ i1)
```

Out[18]: -20

In [ ]:
```
1. In the above case BODMAS rule is applied
```

In [ ]:
```
WHY WE NEED TO LEARN PHYTON FOR DATA ANAYLYST: TO CLEAN THE DATA
```

FLOAT DATA TYPE

1. Float division will always have decimal value

In [21]:
```
f = 110.23
f
```

Out[21]: 110.23

In [22]:
```
type (f)
```

Out[22]: float

In [26]:
```
f, f1 , f2, f3 = 110.23, 2.3, 3.4, 5.1
f,f1, f2,f3
```

Out[26]: (110.23, 2.3, 3.4, 5.1)

In [30]:
```
print (f,f1, f2, f3)
```

```
110.23 2.3 3.4 5.1
```

1. if you want output one after the other write the code in above pattern

In [31]:
```
print(f)
print (f1)
print (f2)
print (f3)
```

```
110.23
2.3
3.4
5.1
```

1. if you want output in straight line oreder write the code in above pattern

In [32]:
```
f1 = 1e0
f1
```

Out[32]: 1.0

In [33]:
```
f2 = 2e1
f2
```

Out[33]: 20.0

In [38]:
```
f3 = 3e2
f3
```

Out[38]: 300.0

In [39]:
```
f4= 3e3
f4
```

Out[39]: 3000.0

In Mathamatics exponent (e to the power of zero is one)

In [ ]:
```
e0 is 1
```

In [ ]:
```
e1 is 10
```

In [ ]:
```
e2 is 100
```

In [ ]:
```
e3 is 1,000
```

In [ ]:
```
e4 is 10,000
```

In [ ]:
```
e5 is 10,00,000 and so on
```

Only 'e' letter allowed in float

In [40]:
```
f5=2.4e2
f5
```

Out[40]: 240.0

BOOL or BOOLEAN DATA TYPE || True or FALSE

In [41]:
```
b = true
b
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[41], line 1
----> 1 b = true
      2 b

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

In [ ]:
```
1. It is showing error because, in Python when ever you define true or fasle the
```

In [ ]:
```
2. we need to enter code as show as below
```

In [42]:
```
b = True
b
```

Out[42]: True

In [43]:
```
b1 = False
b1
```

Out[43]: False

In [44]:
```
print(b)
print(b1)
```

```
True
False
```

In [45]:
```
True + False
```

Out[45]: 1

In [46]:
```
True - False
```

Out[46]: 1

In [47]:
```
False - True
```

Out[47]: -1

In [48]:
```
True + True + True + False - True
```

Out[48]: 2

In [49]:
```
False * True
```

Out[49]: 0

In [50]:
```
True * True
```

Out[50]: 1

In [51]:
```
False / True
```

Out[51]: 0.0

In [ ]:
```
1. In the above  codes the out put showing as 1 since there are no digit, becaus
```

In [ ]:
```
2. 1 + 0 = 1 (True + False)
```

In [ ]:
```
3. 1 - 0 = 1 (True - False)
```

In [ ]:
```
4. 0 - 1 = -1 (False - True)
```

In [ ]:
```
5. 1 + 1 + 1 + 0 - 1 = 2 ( True + True + True + False - True )
```

In [ ]:
```
6. 0 * 1 = 0 ( Falae * True ) 'Multiplication'
```

In [ ]:
```
7. 1 * 1 = 1 ( True * True) 'Multiplication'
```

In [ ]:
```
8. 0 / 0 = 0 ( False / True) 'Divison'
```

In [ ]:
```
True / False
```

In [ ]:
```
1. The above code it is showing as zero division error
```

In [ ]:
```
2.  In python there a 3 typres of errors
     --> Compile time error
     --> run time error
     --> Zero divison error
```

In [ ]:
```
3. Complie time error is: a user whicle writing code some words are missing like
```

In [ ]:
```
4. Run time error: A computer shows an error, Which is not from user end  but fr
```

In [ ]:
```
5. Zero time error: No error from user, but system understand the error and will
```

COMPLEX DATA TYPE

In [ ]:
```
1. The fromula for complex is = (a+bj)
```

In [ ]:
```
2. a means real part
```

In [ ]:
```
3. b means imaginary part
```

In [ ]:
```
4. j menas square root of minus one (-1)
```

In [53]:
```
c= 1 + 20j
c
```

Out[53]:
```
(1+20j)
```

In [54]:
```
type(c)
```

Out[54]:
```
complex
```

In [60]:
```
c
```

Out[60]:
```
(1+20j)
```

In [61]:
```
c.real
```

Out[61]:
```
1.0
```

In [62]:
```
c.imag
```

Out[62]:
```
20.0
```

type c. and hit tab button you can see the options

In [67]:
```
c1 = 10+20j
c2 = 30+40j

print (c1 + c2)

print (c1 - c2)
```

```
(40+60j)
(-20-20j)
```

No space should be given befro j (20j not 20 j)

STRING DATA TYPE

In [68]:
```
s = 'nit'
s
```

Out[68]: 'nit'

In [70]:
```
type (s)
```

Out[70]: str

In [82]:
```
s1= 'hello python'
s1
```

Out[82]: 'hello python'

In [77]:
```
s2 = '''nit hello python'''
s2
```

Out[77]: 'nit hello python'

In [ ]:
```
1. We can define string in 3 ways
```

In [ ]:
```
single qutoe      '   '
```

In [ ]:
```
double quote     ''    ''
```

In [ ]:
```
thriple quote    '''     '''
```

In [ ]:
```
Single qutoe and Double quotes  used for single line comment
```

In [ ]:
```
Thriple quote we use for multiline comments
```

STRING INDEXING

In [ ]:
```
1. In python numbers begin from 0 not from 1
   For rxample: 0,1,2,3,4, and so on.
```

In [ ]:
```
2. n= (n-1)
```

In [ ]:
```
3. String are divied in 2 types  Forward indexing and backward indexing.
```

In [ ]:
```
4. Forward indexing means --> LEFT to RIGHT ( 0,1,2,3,4,.....)
```

In [ ]:
```
5. Backward indexing means -->  RIGHT to LEFT (-1, -2, -3, -4 ..........)
```

In [85]:
```
s1
```

Out[85]:
```
'hello python'
```

In [86]:
```
s1 [0]
```

Out[86]:
```
'h'
```

In [87]:
```
s1[-4]
```

Out[87]:
```
't'
```

In [88]:
```
s1[4]
```

Out[88]:
```
'o'
```

In [90]:
```
s1[5]
```

Out[90]:
```
' '
```

1. In hello phtyon string the fifth place is given as space that is why when you enter s1[5] it is showning space in the out put as above.

In [91]:
```
s1[-7]
```

Out[91]:
```
' '
```

In [92]:
```
s
```

Out[92]:
```
'nit'
```

In [94]:
```
print(s[0])
print(s[1])
print(s[2])
```

```
n
i
t
```

In the above if we want to print in straight line use inner brackets like this [ ]

STRING SLICING

Slincing denotes with coloum (:)

In [95]:
```
s1
```

Out[95]:
```
'hello python'
```

In [97]:
```
s1[:]
```

Out[97]:
```
'hello python'
```

Empty Slice means it print the string as it is.

In [98]:
```
s1[2:7]
```

Out[98]:
```
'llo p'
```

In [ ]:
```
1.The above code states s1[2:7} means print 2 index to 7 index from word hello p
```

In [ ]:
```
hello python
```

In [ ]:
```
01234567891011
```

In [ ]:
```
2:7= llo p( we picked the letter which are fronm 2 to 7 in word hello python, bu
```

In [ ]:
```
2. For the right side index we apply fromula (n-1)
```

In [101…
```
s3= 'dataanalyst'
s3
```

Out[101…
```
'dataanalyst'
```

In [102…
```
s3[0:10]
```

Out[102…
```
'dataanalys'
```

In [103…
```
s3[0:11]
```

Out[103…
```
'dataanalyst'
```

In [105…
```
s3[12]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[105], line 1
----> 1 s3[12]

IndexError: string index out of range
```

in the above code it is hsowing error because the word dataanalyst have only 10 letters and we have entered code as s3[12] which is out of range, that is why it is showing error

In [106…
```
s3[10]
```

Out[106…
```
't'
```

In [107…
```
s3[9]
```

Out[107…
```
's'
```

In [108…
```
s3[9:12]
```

Out[108…
```
'st'
```

STRING ADVANCE SLICING

In [110…
```
s3
```

Out[110… 'dataanalyst'

In [109…
```
s3[0:11:2]
```

Out[109… 'dtaayt'

In [ ]:
```
dataanalyst
```

In [ ]:
```
012345678910
```

In [ ]:
```
1. In the above code 2 means step count
```

In [ ]:
```
2. The step count eliminates 2 letters from index, since we mentioned 2 as step
```

In [ ]:
```
3. Whatever number you mention in step count it eliminates that many letter
```

In [ ]:
```
For example: if you entered step count as 3 it eliminates 3 letters forn the ind
```

In [111…
```
s3[0:11:3]
```

Out[111… 'daas'

In [112…
```
s3[0:11:4]
```

Out[112… 'day'

In [113…
```
s3[2:-2]
```

Out[113… 'taanaly'

In [117…
```
print(s)
print(s1)
print(s2)
print(s3)
```

```
nit
hello python
nit hello python
dataanalyst
```

In [118…
```
for i in s3:
    print(i)
```

```
d
a
t
a
a
n
a
l
y
s
t
```

END OF SESSION