python Variables / Identifires

An identifier is a name given to entities like class, functions, variables, etc. It helps to differentiate one entity from another.

In [1]:
```
1var = 10 # identifier can't start with a digit
```

```
  Cell In[1], line 1
    1var = 10 # identifier can't start with a digit
    ^
SyntaxError: invalid decimal literal
```

In [3]:
```
var@ = 20 # Identifier can't use special symbols
```

```
  Cell In[3], line 1
    var@ = 20 # Identifier can't use special symbols
        ^
SyntaxError: invalid syntax
```

In [4]:
```
import = 125 # keyword can't be used as identifiers
```

```
  Cell In[4], line 1
    import = 125 # keyword can't be used as identifiers
           ^
SyntaxError: invalid syntax
```

In [5]:
```
NIT = 15 # Correct way of defining an identifier
NIT
```

Out[5]:
```
15
```

In [6]:
```
NIT = 20
NIT
```

Out[6]:
```
20
```

In [7]:
```
v = 25
v
```

Out[7]:
```
25
```

In [8]:
```
print(v)
```

```
25
```

Comments in python

Comments can be used to explain the code for more readabilty.

In [11]:
```
# single line coment
val1 = 10
```

In [12]:
```
# Multiple
# Line
# Comment
val1 = 10
```

In [14]:
```
'''
Multiple
line
comment
'''
val1 = 10
```

In [15]:
```
"""
Multiple
line
Comment
"""
val1 = 10
```

Statements

In [16]:
```
p = 20
q = 20
r = q
p, type(p), hex(id(p))
```

Out[16]: (20, int, '0x7fffd7e63c18')

In [17]:
```
q, type(q), hex(id(q))
```

Out[17]: (20, int, '0x7fffd7e63c18')

In [18]:
```
r, type(r), hex(id(r))
```

Out[18]: (20, int, '0x7fffd7e63c18')

In [19]:
```
p= 20
p = p + 10 # variable overwriting
p
```

Out[19]: 30

Variable Assignment

In [20]:
```
intvar = 10 # Integer variable
floatvar = 2.57 # Float Variable
strvar = "Python Language" # String variable
print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.57
Python Language
```

Multiple Assignments

In [21]:
```
intvar , floatvar , strvar = 10,2.57,"Python Language" # Using commas to separat
print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.57
Python Language
```

In [22]:
```
p1 = p2 = p3 = p4 = 44 # All variables pointing to same value
print(p1,p2,p3,p4)
```

```
44 44 44 44
```

Python data types

- INT - value without decimal
  - FLOAT - value with decimal (petrol price, gold price, salary)
  - BOOL - True or False
  - STRING - 'nit ' | '' nit ''
  - COMPLEX - (a +)

Integer

In [2]:
```
i = 30
i
```

Out[2]: 30

In [3]:
```
type(i)
```

Out[3]: int

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

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

In [5]:
```
i
```

Out[5]: 30

In [6]:
```
i1,i2 = 20,30
```

In [7]:
```
i + i1 + i2
```

Out[7]: 80

In [8]:
```
i - i2  + i1
```

Out[8]: 20

In [9]:
```
print(i)
print(i1)
print(i2)
```

```
30
20
30
```

In [10]:
```
i - (i2 +i1)
```

Out[10]: -20

Float

In [11]:
```
f = 110.23
f
```

Out[11]: 110.23

In [12]:
```
type(f)
```

Out[12]: float

In [13]:
```
f1, f2, f3 = 2.3, 3.4, 4.5
```

In [14]:
```
print(f)
print(f1)
print(f2)
print(f3)
```

```
110.23
2.3
3.4
4.5
```

In [16]:
```
f1 = 1e0
f1
```

Out[16]: 1.0

In [17]:
```
f2 = 2e3
f2
```

Out[17]: 2000.0

In [18]:
```
f3 = 3e3
f3
```

Out[18]: 3000.0

In [19]:
```
f4 = 4.2e2
f4
```

Out[19]: 420.0

In [20]:
```
f5 = 5.3e3
f5
```

Out[20]: 5300.0

In [21]:
```
f6 = 7.23e4
f6
```

Out[21]: 72300.0

Bool or Boolean

In [22]:
```
b = True
b
```

Out[22]: True

In [23]:
```
b1 =  False
b1
```

Out[23]: False

In [24]:
```
print(b)
print(b1)
```

```
True
False
```

In [25]:
```
True + False
```

Out[25]: 1

In [26]:
```
True - False
```

Out[26]: 1

In [27]:
```
False - True
```

Out[27]: -1

In [28]:
```
False + True
```

Out[28]: 1

In [29]:
```
True + True + True + False - True
```

Out[29]: 2

In [30]:
```
True * False
```

Out[30]: 0

In [31]:
```
True * True
```

Out[31]: 1

In [32]:
```
True / True
```

Out[32]: 1.0

In [33]:
```
False / True
```

Out[33]: 0.0

Complex

In [51]:
```
c = 1 + 20j
c
```

Out[51]: (1+20j)

In [52]:
```
type(c)
```

Out[52]: complex

In [53]:
```
c
```

Out[53]: (1+20j)

In [54]:
```
c.real
```

Out[54]: 1.0

In [56]:
```
c.imag
```

Out[56]: 20.0

In [57]:
```
c1 = 10 + 20j
c2 = 30 + 40j

print(c1 + c2)
print(c1 - c2)
```

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

String

In [34]:
```
s = 'nit'
s
```

Out[34]: 'nit'

In [35]:
```
type(s)
```

Out[35]: str

In [36]:
```
s1 = 'hello python'
s1
```

Out[36]: 'hello python'

In [37]:
```
s1
```

Out[37]: 'hello python'

In [38]:
```
s = 'nit'
s
```

Out[38]: 'nit'

In [39]:
```
type(s)
```

Out[39]: str

In [40]:
```
s1 = 'hello python'
s1
```

Out[40]: 'hello python'

In [41]:
```
s1[0]
```

Out[41]: 'h'

In [42]:
```
s2 ='''nit
    hello python'''
s2
```

Out[42]: 'nit\n   hello python'

In [43]:
```
s1 [1]
```

Out[43]: 'e'

In [44]:
```
s1[4]
```

Out[44]: 'o'

In [45]:
```
s1[6]
```

Out[45]: 'p'

In [46]:
```
s1[0:13]
```

Out[46]: 'hello python'

In [47]:
```
print(s[0])
print(s[1])
print(s[2])
```

```
n
i
t
```

s2 ='''nit hello python''' s2

In [48]:
```
s1
```

Out[48]:
```
'hello python'
```

In [49]:
```
s2 ='''nit
       hello python'''
s2
```

Out[49]:
```
'nit\n        hello python'
```

In [50]:
```
s3 = 'dataanalyst'
s3
```

Out[50]:
```
'dataanalyst'
```

In [51]:
```
s3[0:10]
```

Out[51]:
```
'dataanalys'
```

In [52]:
```
s3[0:5]
```

Out[52]:
```
'dataa'
```

In [53]:
```
s3 [0:11]
```

Out[53]:
```
'dataanalyst'
```

In [54]:
```
s3
```

Out[54]:
```
'dataanalyst'
```

In [56]:
```
s3[10]
```

Out[56]:
```
't'
```

In [57]:
```
s3[0:11:2]
```

Out[57]:
```
'dtaayt'
```

In [58]:
```
import keyword
keyword.kwlist
```

Out[58]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 'break',
 'class',
 'continue',
 'def',
 'del',
 'elif',
 'else',
 'except',
 'finally',
 'for',
 'from',
 'global',
 'if',
 'import',
 'in',
 'is',
 'lambda',
 'nonlocal',
 'not',
 'or',
 'pass',
 'raise',
 'return',
 'try',
 'while',
 'with',
 'yield']
```

In [59]:
```
for i in s3:
    print(i)
```

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

Python data type completed

Python type casting | type conversion

In [60]:
```
int(2.3) # float to int
```

Out[60]: 2

In [61]:
```
int(True) #bool to int
```

Out[61]: 1

In [62]:
```
int(1+2j) #complex to int not possible
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[62], line 1
----> 1 int(1+2j)

TypeError: int() argument must be a string, a bytes-like object or a real number,
not 'complex'
```

In [63]:
```
int('10')
```

Out[63]: 10

In [64]:
```
s2
```

Out[64]: 'nit\n\thello python'

In [65]:
```
del s2
```

In [67]:
```
float(3)
```

Out[67]: 3.0

In [68]:
```
float(True)
```

Out[68]: 1.0

In [72]:
```
float('10')
```

Out[72]: 10.0

In [86]:
```
complex(10)
```

Out[86]: (10+0j)

In [73]:
```
complex(10, 20)
```

Out[73]: (10+20j)

In [83]:
```
complex(2.3)
```

Out[83]: (2.3+0j)

In [84]:
```
complex(2.3, 10)
```

Out[84]: (2.3+10j)

In [85]:
```
complex(False)
```

Out[85]: 0j

In [78]:
```
complex(True)
```

Out[78]: (1+0j)

In [79]:
```
complex('10')
```

Out[79]: (10+0j)

In [74]:
```
bool(1)
```

Out[74]: True

In [75]:
```
bool(0)
```

Out[75]: False

In [82]:
```
bool()
```

Out[82]: False

In [ ]:
```
bool('nit')
```

In [80]:
```
bool(2.3)
```

Out[80]: True

In [81]:
```
bool(0.0j)
```

Out[81]: False

In [77]:
```
print(str(2))
print(str(2.3))
print(str(True))
print(str(1+2j))
```

```
2
2.3
True
(1+2j)
```

In [ ]:
```

```