PYTHON TUTORIAL

In [1]:
```
import sys
import keyword
import operator
from datetime import datetime
import os
```

Keywords

. keywords are the reserved words in python and can't be use an identifier (variable)

In [2]:
```
print(keyword.kwlist)      # list of all python keywords
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'clas
s', '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 [3]:
```
len(keyword.kwlist)        # python contains 35 keywords
```

Out[3]: 35

Identifiers

An identifier ia a name given to entities like class,function,variable,etc,,,. it helps to differentiate one entity another.

In [4]:
```
1var = 10         # variable can't start with numbers / digits
```

```
  Cell In[4], line 1
    1var = 10         # variable can't start with numbers / digits
    ^
SyntaxError: invalid decimal literal
```

In [5]:
```
var1 = 10
var1                    # this is correct because after vaariable name we can u
```

Out[5]: 10

In [6]:
```
var2 = 36
var2        # this is correct because after vaariable name we can use digits..
```

Out[6]: 36

In [7]:
```
var_1 = 24345
var_1
```

Out[7]: 24345

Comments in Python

. Comments can be useed to explain the code for more readablity.

In [8]:
```
# Single line comment
Val1 = 10
```

In [9]:
```
# Multiple
# line
# Comment
Val1 = 10
```

In [10]:
```
'''Multiple
line
comment
'''
Val1 = 10
```

In [11]:
```
"""Multiple
line
comment
"""
Val1 = 10
print(Val1)
```

```
10
```

In [12]:
```
p = 20                      # creats an integer object with value 20 and assigns
q = 20                      # create new refrance q which will point to the value
q = q                       # Variable r is also point ti the same location where
p, type(p), hex(id(p))      # variable p is pointing to the memory location '0x7f
```

Out[12]:
```
(20, int, '0x7ffe10d5b608')
```

In [13]:
```
q , type(q), hex(id(q))
```

Out[13]:
```
(20, int, '0x7ffe10d5b608')
```

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

Out[19]:
```
30
```

Variable Assignment

In [20]:
```
intvar = 10  # integer Variable
floatvar = 2.27   # float variable
strvar = "PYTHON LANGUAGE"  # string variable

print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.27
PYTHON LANGUAGE
```

Multiple Assignments

In [23]:
```
intvar , floatvar , strvar = 10,2.56,"python language"
print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.56
python language
```

In [24]:
```
p1 = p2 = p3 = p4 = 11 # all variable pointing same vallue
print(p1,p2,p3,p4)
```

```
11 11 11 11
```

DATA TYPES

In [25]:
```
val1 = 10 # Integer data type
print(val1)
print(type(val1)) # type of object
print(sys.getsizeof(val1)) # size of integer object in bytes
print(val1, " is Integer?", isinstance(val1, int)) # val1 is an instance of int
```

```
10
<class 'int'>
28
10  is Integer? True
```

In [26]:
```
val2 = 146.76 # float data type
print(val2)
print(type(val2)) # type of object
print(sys.getsizeof(val2)) # size of float object in bytes
print(val2, " is float?", isinstance(val2, float)) # val1 is an instance of int
```

```
146.76
<class 'float'>
24
146.76  is float? True
```

In [27]:
```
val3 = 25 + 10j   # complex data type
print(val3)
print(type(val3)) # type of object
print(sys.getsizeof(val3)) # size of complex object in bytes
print(val3, " is complex?", isinstance(val3, complex)) # val1 is an instance of
```

```
(25+10j)
<class 'complex'>
32
(25+10j)  is complex? True
```

In [28]:
```
sys.getsizeof(int())   # size of integer object in bytes
```

Out[28]: 28

In [29]:
```
sys.getsizeof(float())   # size of integer object in bytes
```

Out[29]: 24

In [30]:
```
sys.getsizeof(complex())   # size of integer object in bytes
```

Out[30]: 32

In [32]:
```
sys.getsizeof(str())   # size of integer object in bytes
```

Out[32]: 41

In [33]:
```
sys.getsizeof(bool())   # size of integer object in bytes
```

Out[33]: 28

Boolean

. Boolean data type can have only two possible values True or False.

In [34]:
```
bool1 = True
bool1
```

Out[34]: True

In [35]:
```
print(type(bool1))
```

```
<class 'bool'>
```

In [36]:
```
isinstance(bool1, bool)
```

Out[36]: True

In [37]:
```
bool(0)         # Always in memory False store as a zero (0).
```

Out[37]: False

In [38]:
```
bool(1)          # Always in memory True store as a one (1).
```

Out[38]: True

In [39]:
```
bool(None)
```

Out[39]: False

In [40]:
```
bool(False)
```

Out[40]: False

String

- String Creation

In [42]:
```
str1 = "HELLO"
print(str1)
```

```
HELLO
```

In [43]:
```
mystr = 'Good Morning'   # Define String using Single Quotes
print(mystr)
```

```
Good Morning
```

In [44]:
```
mystr = "Good Afternoon"   # Define String using Double Quotes
print(mystr)
```

```
Good Afternoon
```

Srting Indexing

- Forward Indexing

In [45]:
```
str1 = 'Hello'
print(str1)
```

```
Hello
```

In [46]:
```
str1[0]      # First character in string "str1"
```

Out[46]: 'H'

In [47]:
```
str1[2]        # third character in string "str1"
```

Out[47]: 'l'

String Slicing

In [49]:
```
str1[0 : 2]
```

Out[49]: 'He'

In [50]:
```
str1[0:5]
```

Out[50]: 'Hello'

Update & Delete String

In [51]:
```
str1
```

Out[51]: 'Hello'

In [ ]:
```

```