Variables

In [1]:
```
x = 10
y = "Python"
z = 'Python'
```

In [2]:
```
print(x)
print(y)
print(z)
```

```
10
Python
Python
```

In [7]:
```
myvar = 'Shyam'
my_var = 'Shyam'
_my_var = 'Python'
myVar = "Python"
MYVAR = "Data Science"
myvar10 = 'Python'
```

In [8]:
```
2myvar = 'Python'
```

```
  File "<ipython-input-8-c52fec124df2>", line 1
    2myvar = 'Python'
     ^
SyntaxError: invalid syntax
```

In [9]:
```
my-var = 'Python'
```

```
  File "<ipython-input-9-c3f5943790e6>", line 1
    my-var = 'Python'
    ^
SyntaxError: cannot assign to operator
```

In [12]:
```
my var = 'Shyam'
```

```
  File "<ipython-input-12-5b0786cece3a>", line 1
    my var = 'Shyam'
       ^
SyntaxError: invalid syntax
```

Multiple LIne variable:

```
1. Camel Case
2. Pascal Case
3. Snake Case
```

In [14]:
```
# camel case
myVariableName = 'python'
```

In [15]:
```
#Pascal case
MyVariableName = 'python'
```

In [16]:
```
#snake case

my_variable_name = 'Python'
```

Many Values to Multiple Variables

In [17]:
```
x, y, z = "Mango", "Apple","Papaya"
```

In [18]:
```
print(x)
```

```
Mango
```

In [19]:
```
print(y)
```

```
Apple
```

In [20]:
```
print(z)
```

```
Papaya
```

In [21]:
```
#x, y, z single value assign

x = y = z = 'Orange'
```

In [22]:
```
print(x)
print(y)
print(z)
```

```
Orange
Orange
Orange
```

In [23]:

```
fruits = ['Apple', 'Mango',"Cherry","Banana"]

a, b, c, d = fruits


print(a)
print(b)
print(c)
print(d)
```

```
Apple
Mango
Cherry
Banana
```

Output Variable:

1. print()
2. to combine both text and variable python use + character

In [25]:

```
x = "Apple"
y = 'Mango'
print("My Fruit Name is "+ x )
print("My Fruit Name is "+ y )
```

```
My Fruit Name is Apple
My Fruit Name is Mango
```

In [26]:

```
x = 'Awesome'
y = 'Python is'

z = y + x
print(z)
```

```
Python isAwesome
```

In [30]:

```
x = 'Awesome  '
y = 'Python is'

z = x + y
print(z)
```

```
Awesome Python is
```

In [31]:

```
x = 100
y = 200

z= x + y
print(z)
```

```
300
```

In [32]:
```
x = 100
y = 200

z= x - y
print(z)
```

```
-100
```

In [33]:
```
x = 10
y = 20

z= x * y
print(z)
```

```
200
```

In [36]:
```
x = 100
y = "python"

z= x + y
print(z)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-36-1599c4de535c> in <module>
      2 y = "python"
      3 
----> 4 z= x + y
      5 print(z)

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Gloabal Variable

In [40]:
```
x = 'Awesome'

def myFunc():
print("Python is "+ x)
myFunc()

print("PythonData Scince is "+x)
```

```
  File "<ipython-input-40-f5ae293bf008>", line 4
    print("Python is "+ x)
    ^
IndentationError: expected an indented block
```

In [44]:

```
x = 'Awesome'

def myFunc():
    print("Python is "+ x)
    print("PythonData Scince is "+x)
myFunc()

print("PythonData Scince is "+x)
```

```
Python is Awesome
PythonData Scince is Awesome
PythonData Scince is Awesome
```

Local Variable

In [51]:

```
x = 'Awesome' # Global Variable

def myFunc():
    global x
    x = 'Fantastic ' #Local Variable
    print('Python is '+ x)
myFunc()

print("PythonData Scince is "+x)
```

```
Python is Fantastic
PythonData Scince is Fantastic
```

Python Data Types:

Built-in Data Types:

1. `variable can store different types values or different types things`
2. `There are different built-in datatypes in Python`

1. Text Data [str]
2. Numeric Types [int, float, complex]
3. Sequence Types [list, tuple, range]
4. Mapping Type [dict]
5. Set [Set, Frozenset]
6. Bolean Type [bool]
7. Binary Types [bytes, bytearray, memoryview]

In [54]:

```
#string
x = 'Hello World'

#Numeric Data
x = 10
x = 45.20
x = 1j

#Sequence Type
x = ['apple','mango','cherry']
x = ('apple', 'mango')
x = range(6)

#Mapping Type
x = {'name':'Shyam','age':28}

#Set
x= {'apple','mango','cherry'}
x = frozenset({'apple','mango','cherry'})

# Boolean Value
x = True, False

#Binary Types
x = b'Python'
x = bytearray(5)
x =  memoryview(bytes(5))
```

Python Numbers:

1. int
2. float
2. complex

In [55]:

```
x = 10 #int
y = 12.12 # float
z = 1j # complex
```

In [56]:

```
print(x)
print(y)
print(z)
```

```
10
12.12
1j
```

In [57]:

```
print(type(x))
print(type(y))
print(type(z))
```

```
<class 'int'>
<class 'float'>
<class 'complex'>
```

In [58]:

```
x = -120
y = 13456789
z = 1
```

In [59]:

```
print(x)
print(y)
print(z)
```

```
-120
13456789
1
```

In [60]:

```
print(type(x))
print(type(y))
print(type(z))
```

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

In [61]:

```
x = -120.23
y = 13456789.25
z = 1.12
```

In [62]:

```
print(x)
print(y)
print(z)
```

```
-120.23
13456789.25
1.12
```

In [63]:

```
print(type(x))
print(type(y))
print(type(z))
```

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

Complex

In [64]:

```
x = 3+5j
y = 5j
z = -5j
```

In [65]:

```
print(x)
print(y)
print(z)
```

```
(3+5j)
5j
(-0-5j)
```

In [66]:

```
print(type(x))
print(type(y))
print(type(z))
```

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

Type Conversion

In [70]:

```
x = 1
y = 20.45
z = 1j

print(x)
print(y)
print(z)

#Convert int into float value
a = float(x)

# Convert float value into int
b = int(y)

#Convert from int to complex
c =  complex(x)

print(a)
print(b)
print(c)
```

```
1
20.45
1j
1.0
20
(1+0j)
```

In [71]:

```
x  = int(10)
y = float(2.8)
```

In [72]:

```
print(x)
print(y)
```

```
10
2.8
```

Python Strings

In [73]:

```
print("Welcome to Data science batch")
```

```
Welcome to Data science batch
```

In [74]:

```
print('Welcome to Data science batch')
```

```
Welcome to Data science batch
```

In [76]:

```
x = 'Python'
y = "Python"
```

In [77]:

```
print(x)
print(y)
```

```
Python
Python
```

In [78]:

```
x = """ python is good welcome to data science batch
Python is Awesome
Python is Fantastic"""
```

In [80]:

```
print(x)
```

```
 python is good welcome to data science batch
Python is Awesome
Python is Fantastic
```

In [82]:

```
x = 'Python'
y = "Python"
```

In [83]:

```
x = ''' python is good welcome to data science batch
Python is Awesome
Python is Fantastic'''
```

In [84]:

```
print(x)
```

```
 python is good welcome to data science batch
Python is Awesome
Python is Fantastic
```

In [89]:

```
x = 'Hello, World'

print(x[0:4])
```

```
Hell
```

In [ ]:

```

```