In [1]:
```
# Day 01

# Terminologies.
# Variables
# Operators

# Day 02

# Datatypes
# Conditional Statements
# Looping Statements
# Functions

# Day 03

# Python Basics Hands-On
```

In [2]:
```
# Terminologies :

# Syntax : it is the grammer of any programming language.It is the format that is to be followed while writing a program.
# Keywords : certain words that has some predefined meaning in python.
# Inbuilt Functions : The block of code that has some predefined task to perform. Eg: print()
# Case Sensitivity : Uppercase letter and Lowercase letter are treated as two different entities. Eg: 'A' and 'a' are different.
# Comments : Comments are not the part of the code, but they are used to explain the code. CTRL + / is the shortcut to comment.
# Shift + Enter to Execute a line of code.
```

In [3]:
```
# Variables :

# -> a storage container that holds some data.
# -> a memory location that will be assigned with some values or information.

# SYNTAX for defining a variable is :
#     variable_name = value

a = 10
print(a)

# a -> variable_name
# = -> assignment operator
# 10 -> value -> literal
```

```
10
```

In [4]:
```
# Defining a variable :

a = 10 # int : any numeric value without decimal point
b = 20 # int : any numeric value without decimal point
c = 24.56 # float : any numeric value with a decimal point
d = "Avinash" # str : anything within "" or ''
e = 'Avinash' # str : anything within "" or ''
f = "1234" # str : anything within "" or ''
g = "Avinash20" # str : anything within "" or ''

# print() : inbuilt function that is used to display the output.

print(a)
print(b)
print(c)
print(d)
print(e)
print(f)
print(g)

# type() : inbuilt function that is used to determine the type of value stored in a variable.

print(type(a))
print(type(b))
print(type(c))
print(type(d))
print(type(e))
print(type(f))
print(type(g))
```

```
10
20
24.56
Avinash
Avinash
1234
Avinash20
<class 'int'>
<class 'int'>
<class 'float'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
```

In [5]:
```
# Rules to be followed while naming a variable :
# 1. Keywords can't be used as a variable name.
# 2. Variable name can start with a letter or an underscore(_) but never with a number.
# 3. No special symbols or characters are allowed in a variable name except for underscore(_).

# def = 123
# print = 1234
# print(print)

avinash = 1234
_aviansh = 1234
# 1avinash = 1234
avinash1 = 1234

_aviansh = 1234
avinash_ = 1234
# &avinash = 1234
# avi&nash = 1234
```

In [6]:
```
# Assigning a value to a variable:

# 1. Static Way -> value is assigned to the varaible while writing the program.

a = 10 # integer : any numeric value without decimal point
b = 20 # integer : any numeric value without decimal point
c = 24.56 # float : any numeric value with a decimal point
d = "Avinash" # str : anything within "" or ''
e = 'Avinash' # str : anything within "" or ''
f = "1234" # str : anything within "" or ''
g = "Avinash20" # str : anything within "" or ''
```

In [12]:
```
# 2. Dynamic Way -> value is assigned to the variable during the run time by the user.
#     input() : inbuilt function that is used to fetch data from the user.
#     SYNTAX:
#         variable_name = input("message")

# Write a program to add two numbers :

a = int(input("Enter the value of a :"))
b = int(input("Enter the value of b :"))
print(a+b)
```

```
Enter the value of a :23
Enter the value of b :77
100
```

In [ ]:

```python
# Operators

# -> These are some symbols that perform some specific tasks.

# There are seven types of operators in python :
# 1. Arithmatic Operator : +, -, *, /, %, **
# 2. Assignment Operator : =, +=, -=, *=, /=
# 3. Comparison Operator : >, <, <=, >=, ==, !=
# 4. Logical Operators : and, or, not
# 5. Bitwise Operators : &, |, <<, >>
# 6. Identity Operator : is, is not
# 7. Membership Operator : in, not in
```

In [1]:

```python
# 1. Arithmatic Operator :

# + -> Addition
# - -> Substraction
# * -> Multiplication
# / -> Division
# % -> Modulus : Return the remainder of the division.
# ** -> Exponential : a to the power of b

x = 100
y = 50

print(x + y) # 150
print(x - y) # 50
print(x * y) # 5000
print(x / y) # 2.0 because division always return value in float
print(x % y) # 0
print(x ** y) # 100 power of 50
```

```
150
50
5000
2.0
0
10000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

In [3]:

```python
# 2. Assignment Operator : =, +=, -=, *=, /=

a = 10
print(a)

a += 10 # a = a + 10
```

```
print(a) # 20
a -= 10 # a = a - 10
print(a) # 10
a *= 10 # a = a * 10
print(a) # 100
a /= 10 # a = a / 10
print(a) # 10.0
```

```
10
20
10
100
10.0
```

In [7]:
```
# 3. Comparison Operator :

# -> The comparison operator is used to compare between two entities.
# -> The comparison operator always returns either True or False

# >, <, <=, >=, ==, !=

p = 50
q = 50

print(p>q) # False
print(p<q) # False
print(p>=q) # true
print(p<=q) # true
print(p==q) # True
print(p!=q) # False
```

```
False
False
True
True
True
False
```

In [ ]:
```

```