In [1]:

```
print("Hello everyone")
```

```
Hello everyone
```

In [2]:

```
print("hi")
```

```
hi
```

In [ ]:

```

```

Heading1

Heading2

Heading3

Head6

Today **learning** python

I am in *italic*

- Java
- Python
  - v1.0
  - v3.8

clickMe (https://www.google.com)


```
function fancyAlert(arg) {
  if(arg) {
    $.facebox({div:'#foo'})
  }
}
```

Python Keywords

In [3]:

```
import keyword
keyword.kwlist
```

Out[3]:

```
['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']
```

Identifiers

Identifier is a name given to the variable,classes ,functions.

Rules

- identifier should start with either alphabets or _(underscore)
- identifier accepts numbers also
- it can not contain spaces and special symbols

In [4]:

```
var1 = 20
first name = "python"
```

```
  File "<ipython-input-4-e90a51bded86>", line 2
    first name = "python"
               ^
SyntaxError: invalid syntax
```

In [5]:

```
_firstName = "python"
```

Variables and data types

named memory location to store the value

In [6]:

```
firstName = 10

print(firstName)
```

```
10
```

In [7]:

```
type(firstName)
```

Out[7]:

```
int
```

Datatypes

1. Numbers (int,float,complex)
2. Characters (Str)
3. True,False (bool)

In [8]:

```
number = 10
print(number, type(number))
number = 10.5

print(number,type(number))
```

```
10 <class 'int'>
10.5 <class 'float'>
```

Operators in python

special symbols to complete special task

Arthematic operators

+ , -, *, / ,// , %, **

In [9]:

```
print(10 + 20)
```

30

In [10]:

```
print(10.50-9)
```

1.5

In [11]:

```
print(10*2)
```

20

In [12]:

```
print(9/2)
```

4.5

2)9(4

8

1

In [13]:

```
print(9//2)
```

4

In [14]:

```
print(9%2)
```

1

In [15]:

```
print(2**3) # 2 power 3
```

```
8
```

In [ ]:

```

```

Assignment Operators

= , += , -=, *= , /= ,//=, %= , **=

In [16]:

```
number = 10
print(number)
```

```
10
```

In [17]:

```
print(number) # 10
number += 2 # number = number + 2

print(number)
```

```
10
12
```

In [18]:

```
print(number) # 12
number -= 4  # number = number- 4
print(number) # 8
```

```
12
8
```

In [19]:

```
number = 10
print(number) # 10
number *= 5  # number = number * 5
print(number) #50
```

```
10
50
```

In [ ]:

```

```

Comparision Operators

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

In [20]:

```
print(2 == 3)
```

```
False
```

In [21]:

```
print(2!=3)
```

```
True
```

In [22]:

```
print(2 < 3)
```

```
True
```

In [23]:

```
print(2 <= 3)
```

```
True
```

In [24]:

```
print(3>=2)
```

```
True
```

In [ ]:

```

```

Logical operators

and , or, not

In [25]:

```
condition1 = 2 < 3  # True
condition2 = 2 > 3  # False
var = None
print(condition1 and condition2,var)
```

```
False None
```

In [26]:

```
print(10,20,30)
```

```
10 20 30
```

In [27]:

```
print(condition1 or condition2 )
```

```
True
```

In [28]:

```
print(not True)
```

```
False
```

In [29]:

```
print(not False)
```

```
True
```

In [ ]:

```

```

Bitwise operators

- & (bitwise and)
- | (bitwise or)
- ~ (bitwise not)
- ^ (bitwise xor)
- **>> (bitwise right shift)**
- << (bitwise left shit)

In [30]:

```
a = 10
b = 11

print(a & b)
```

```
10
```

In [31]:

```
print( a | b )
```

```
11
```

In [32]:

```
print(~ 10)
```

```
-11
```

~ a a = 10

result = -a -1 = -10-1 =-11

In [33]:

```
print(~ 20)
```

```
-21
```

In [34]:

```
print( a ^ b)
```

```
1
```

In [35]:

```
print(10 >> 2)
```

```
2
```

In [36]:

```
print(10 << 2)
```

```
40
```

In [ ]:

```

```

special operators

1. membership operators (in , not in)
   to check particular member available in group of members.
2. identity operators (is , is not)

In [37]:

```
name = "python"


print('h' in name)
```

```
True
```

In [38]:

```
print('H' in name)
```

```
False
```

In [39]:

```
print('H' not in name) # we get True if 'H' not in names else False
```

```
True
```

In [ ]:

```
```

Identity operators

to check both objects are same or not

```
    is , not
```

In [40]:

```
marks_student1 = [10,20,30]
marks_student2 = [10,20,30]

print(marks_student1 == marks_student2)
```

```
True
```

In [41]:

```
print(marks_student1 is marks_student2 )
```

```
False
```

In [42]:

```
marks_student1 = [10,20,30]
marks_student2 = marks_student1

print(marks_student1 is marks_student2)
```

```
True
```

In [ ]:

```
```

Data type conversion

to convert one type from another type. int(), float(), str() , bool()

In [43]:

```
number = 10
print(float(number))
```

```
10.0
```

In [44]:

```
number = 123

print(str(number)) # str(123)  return after conversion result
number = str(number)

print(type(number))
```

```
123
<class 'str'>
```

In [45]:

```
number = '123'


print(int(number))
```

```
123
```

In [46]:

```
name = "123python"
print(int(name))
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
<ipython-input-46-30ac0d1d46dc> in <module>
      1 name = "123python"
----> 2 print(int(name))

ValueError: invalid literal for int() with base 10: '123python'
```

In [47]:

```
number = 123

print(str(number))

number = '123'

print(type(number))
```

```
123
<class 'str'>
```

http://bit.ly/apssdc-python-fdp1 (http://bit.ly/apssdc-python-fdp1)