Python

- is a high level programming language
- created by **Guido van Rossum**, released in 1991.

It is used for :

- web development
- software development
- mathematics
- system scripting

What can python do ?

- can be used on a server to create web applications.
- can be used alongside software to create workflows.
- can connect to database systems. It can also read and modify files.
- can be used to handle big data and perform complex mathematics.

Why Python ?

1. Python **works on different platforms** (Windows, Mac, Linux, Raspberry pi, etc.)
2. Python has a **simple syntax** similar to the English Language.
3. Allows developers to **write programs with fewer lines** than some other programming languages.
4. Python runs on **an interpreter system**, meaning that code can be executed as soon as it is written. This means that **prototyping can be very quick**.
5. Python can be treated in a **procedural way, an object-oriented way or a fucntional way**.

Python vs Other Languages

- Python was **designed for readability**, and has some similarities to the English language with influence from mathematics
- Uses **new lines to complete** a command as opposed to other programming languages which often use semicolons or parenthesis
- Python **relies on Identation, using whitespace**, to define scope, such as the scope of loops, fucntions and classes.Other programming languages use curly-brackets for this purpose.

Python Comments

'#' - can be used to comment a line in python
"""_""" - under triple quotes you can write a multiline string in your code.

Python Variable & Data Types

Variables ?

- Variables are containers for storing data values
- are created the moment you first assign a value to it.

In [1]:

```
x = 'Aakash'
print(x)
type(x)
```

```
Aakash
```

Out[1]:

```
str
```

In [2]:

```
y = 3
type(y)
```

Out[2]:

```
int
```

In [3]:

```
y
```

Out[3]:

```
3
```

Casting

Casting is used to explicitly specify a data type of particular value or to change its data type.

In [4]:

```
i = 3
type(i)
```

Out[4]:

```
int
```

In [5]:

```
i = str(3)
type(i)
```

Out[5]:

```
str
```

In [6]:

```
i = float(3)
type(i)
```

Out[6]:

```
float
```

type()

type() is used to print the data type of a variable

In [7]:

```
x = 'hello'
type(x)
```

Out[7]:

```
str
```

In [8]:

```
y = char(x)     # to convert in string use "str" instead of 'char'
type(y)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-ca5df98d9631> in <module>
----> 1 y = char(x)     # to convert in string use "str" instead of 'char'
      2 type(y)

NameError: name 'char' is not defined
```

Case senitive

In [9]:

```
a = 10      # a & A are different
A = 11
print(a)
print(A)
```

```
10
11
```

Single or Double Quotes ?

In [10]:

```
print("hello")
print('hello')
print('this is Guddan's phone')     #error with a single quote
```

```
  File "<ipython-input-10-ac9bd940add7>", line 3
    print('this is Guddan's phone')     #error with a single quote
                        ^
SyntaxError: invalid syntax
```

In [11]:

```
print("hello")
print('hello')
print("this is Guddan's phone")     # no error using double quotes
```

```
hello
hello
this is Guddan's phone
```

Variables Name

- **Camel Case, Pascal Case, Snake Case**
- **Variables name are case sensitive. Eg: a = 'apple, A = 'mango'**

Illegal Variable Names

- **2myvar = 'sandeep'**
- **my-var = 'sandeep'**
- **my var = 'sandeep'**

Many Values to Multiple Variables

In [12]:

```
x, y, z = 'orange', 'pineapple','mango'
print(x)
print(y)
print(z)
```

```
orange
pineapple
mango
```

One Value to Multiple Variables

In [13]:

```
p = q = r ='orange'
print(p)
print(q)
print(r)
```

```
orange
orange
orange
```

Packing & Unpacking

In [14]:

```
z = ('hello', 'python', 'is','Awesome',1)      #packing a collection
print(z)
```

```
('hello', 'python', 'is', 'Awesome', 1)
```

In [15]:

```
a,b,c,d,e = z     #unpacking a collection
print(d)
```

```
Awesome
```

Output Variables

- **print : is used to output the variables**
- **(+) to add a variable to another variable and also works as mathmatical operator**

In [16]:

```
st1 = 'awesome'
st2 = 'python'
print(st1 + " " + st2)
```

```
awesome python
```

In [17]:

```
2 + 3
```

Out[17]:

```
5
```

Global Variables

- **are created outside of a function**
- **can be used by everyone , both inside & outside of a function**
- **global keyword can be used create a global variable inside a fucntion**
  - **global x**
- **to change the value of a global variable inside a fucntion use, *global* keyword**

In [18]:

```
x = 2    #global variable
y = 1

def my_func1():
    y = 3    #local variable
    print(y)
    return x

def my_func2():
    print(y)
```

In [19]:

```
print(my_func1())
```

```
3
2
```

In [20]:

```
x = 2               #global variable
def my_func1():
    global d        #using global keyword to create a global variable inside a fucntion
    d = 3
    return
my_func1()

print(d)
```

```
3
```

In [21]:

```
my_func1()
```

DATA TYPES

- **a variable can store data of different types, and different tpes can do differnt things**
- **Built-In data types**
  - **Text type, Numeric Type, Sequence type,Mapping type, Set type, Boolean Type**
- **User-defined data types**
  - **Satck, Queue, Linked List, etc.**

In [22]:

```
x = 9    #int
y = 9.8  #float
z = 1j   #complex

a = int(9.8)    #float type value converting to int type
print(a)

b = int(z)
print(b)    # will throw an error because int & float type can't be converted to complex
values
```

```
9
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-22-4e22a3451e54> in <module>
      6 print(a)
      7
----> 8 b = int(z)
      9 print(b)    # will throw an error because int & float type can't be converted to
complex values

TypeError: can't convert complex to int
```

Random NUmbers

Python does not have a random() function to make random number, but python has a buillt in module called random that can be used to make random numbers.

In [23]:

```
import random   # just import the random module
print(random.randrange(1,10))
```

```
1
```

String

In [24]:

```
multi_line = """lorem ipsum dolor sit amet,
consecteur,
adipiscing elit"""

print(multi_line)
```

```
lorem ipsum dolor sit amet,
consecteur,
adipiscing elit
```

Strings are Arrays:

Like many other popular programming language, strings in Python are arrays of bytes representing unicode characters.

In [25]:

```
x = 'sandeep'
x[1]   # print the character at index 1
```

Out[25]:

```
'a'
```

In [26]:

```
for y in x:   #iterate over all the
    print(x)
```

```
sandeep
sandeep
sandeep
sandeep
sandeep
sandeep
sandeep
```

In [27]:

```
a =""
```

```
for y in x:
    a = a+y
    print(a)
```

```
s
sa
san
sand
sande
sandee
sandeep
```

Looping in Strings:

In [39]:

```
for x in "abcd":
    print(x)
```

```
a
b
c
d
```

Checking in Strings

In [28]:

```
check = "ram is a good boy"
print('ram' in check)
```

```
True
```

In [29]:

```
froz = frozenset(("pthon", "is","aweesome"))    # frozenset is an inbuilt fucntion that takes an iterable object as input and makes them immutable
print(froz)
```

```
frozenset({'is', 'aweesome', 'pthon'})
```

In [30]:

```
sett = set(('python','is','awesome'))
sett
```

Out[30]:

```
{'awesome', 'is', 'python'}
```

In [31]:

```
froz1 = frozenset(('is', 'pthon', 'aweesome'))
froz1.add("good")      #we cannot add a element to frosenset
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-31-cb91d0db5f71> in <module>
      1 froz1 = frozenset(('is', 'pthon', 'aweesome'))
----> 2 froz1.add("good")

AttributeError: 'frozenset' object has no attribute 'add'
```

In [32]:

```
sett1 = set(('awesome', 'is', 'python'))
sett1
```

Out[32]:

```
{'awesome', 'is', 'python'}
```

In [33]:

```
sett1.add("sample")      #sets are mutable, you can check that by adding an element
sett1
```

Out[33]:

```
{'awesome', 'is', 'python', 'sample'}
```

In [34]:

```
set1 = {1,2,3,3,3,3,1,0,9,6}
print(set1)
```

```
{0, 1, 2, 3, 6, 9}
```

In [35]:

```
set2 = set(('python'))
print(set2)
```

```
{'y', 'n', 'p', 'o', 'h', 't'}
```

In [36]:

```
var_bool = bool(5)
#bool is used to store two values i.e True and False .
#Bool is used to test whether the result of an expression is true or false.
var_bool
```

Out[36]:

```
True
```

In [37]:

```
xyz = bytes(5)
xyz
```

Out[37]:

```
b'\x00\x00\x00\x00\x00'
```

In [38]:

```
xx = bytearray(5)
xx
```

Out[38]:

```
bytearray(b'\x00\x00\x00\x00\x00')
```

In [ ]:

```

```

In [ ]:

```

```