Unit – 01 : Introduction, Variables and Data Types

- History,
- Features,
- Setting up path,
- Installation and Working with Python.
- Basic Syntax,
- Understanding Python variables,
- Numeric data types.
- Using string data type and string operations,
- Basic Operators,
- Understanding coding blocks.
- Defining list and list slicing,
- Other Data Types- Tuples.
- List, Python Dictionary, Arrays, Associative Arrays/Hashes.

Questions to be discussed :

1. Define Python. Explain the basic features of Python programming.
2. Explain in brief arithmetic & relational operator in python.
3. Explain special operators of python in details.
4. What is numeric data types ? Explain in details numeric data types in python.
5. Write a simple program in python to display the name of your college.
6. Write a python program to find the addition/subtraction/multiplication & division.
7. Explain sequence data types in details.
8. Write short notes :
    a. Python variable
    b. Tuple
    c. Dictionary
    d. List slicing
    e. Membership operator
    f. Identity operator

What is Python ?

- Python is an open source & very powerful high level programming language.
- It is a platform independent programming language which is very easy to learn and use.
- It is an interpreted, object-oriented, general-purpose programming language.
- It is used to develop any application very easy & quickly.
- It can be used as a scripting language.
- It supports automatic garbage collection.
- Python has an extension called (.py).
- Python was developed by Guido van Rossum.
- Python is a case-sensitive programming language.

- Python is used to develop system software, mobile app, GUI, website & web application.
- It is also used in Data science, machine learning, AI & Big data applications.
- It provides dynamic type system & automatic memory management, so the development & maintenance cost is very low.
- It can be easily integrated with C, C++, CORBA, and Java etc.
- Now a day it use by youtube, Instagram, google, Netflix, NASA etc.

History of Python :

- Python was developed by Guido van Rossum in 1980, at the National Research Institute for Mathematics and Computer Science in the Netherlands.
- The name python is taken from the comedy group monty python.
- First time it is lunched in 1991.
- The first version of python 1.0 is lunched in 1994.
- The second version of python 2.0 is lunched in October 2000.
- After a long time the third version of python 3.0 is lunched in 2008.
- The latest version of python 3.10.5 is released in march 2017.

Python Features

- Python is free
- It is simple & easy-to-learn
- Easy-to-maintain
- Large standard library (Numpy & Pandas)
- It is Portable
- It is very fast & powerful.
- Extendable Databases – Python provides interfaces to all major commercial databases.
- Python supports GUI applications.

Application of python :

- Data analysis
- Robotics
- Website & Web application
- Games development
- Desktop application
- Data visualization
- Scientific calculation
- Machine learning & AI
- 3D application development
- Audio & video software development

The Best Python IDEs and Code Editors :

- PyCharm.
- Visual Studio Code.
- Sublime Text.
- Vim.
- Atom.
- Jupyter Notebook etc.

Steps to install Python in Windows :

Step – 1 : Visit Official Site to Download Python Installer(www.python.org).

Step – 2 : We can download the latest Python installer by clicking on the Download button.

Step – 3 : Once the download is completed, then run the installer by double-clicking on the download file.

Step – 4 : It will ask for Add Python 3.10.5 to PATH.

Step – 5 : Click on the checkbox and click on Install Now navigator.

Step – 6 : Now Python is successfully installed in Windows, and you are ready to work with Python.

How to set Path for executing Python programs.

1. Right click on My Computer and click on properties.
2. Click on Advanced System settings.
3. Click on Environment Variable tab.
4. Click on new tab of user variables.
5. Write path in variable name.
6. Copy the path of Python folder.
7. Paste path of Python in variable value.
8. Click on Ok button.
9. Click on Ok button.

Basic Syntax :

- The syntax of the Python programming is the set of rules which defines how a Python program will be written.
- A Python program is read by a parser.
- Python was designed to be a highly readable language.

Syntax :

```
print("SBTE, Patna")
```

Output : SBTE, Patna.

Example :

```
a=10
b=20
print(a+b)
```

Output : 30

Comments in Python :

- A comment begins with a hash character(#).
- All characters after the # character up to the end of the line are part of the comment and the Python interpreter ignores them.
- Python has no multi-lines or block comments facility.
- Example :

```
#This is my first program.
```

Python Variables :

- A variable is a name that is used to store the value.
- Python variable is also known as an identifier and used to hold value.
- In Python, we don't need to specify the type of variable.
- Variable names can be a group of both the letters and digits.
- A variable name always start with a letter or an underscore.

Rules to create variable in python :

Variable names in Python must follows the following rules :

- variable names must start with a letter
- variable names can only contain letters, numbers and the underscore character _
- variable names can not contain spaces or punctuation
- variable names are not enclosed in quotes or brackets

Examples :

Abc, _ab, a1, x123, AB_CD, AB_C etc are valid.

1a, n%4, ABC 9, 123abc etc are invalid.

What are operators in python?

- An operator is a symbol which operates on a value or a variable.
- It tells to the compiler to perform certain arithmetical or logical calculation.
- The value that the operator operates on is called the operand.

Example:

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

Here, '+' is an operator to perform addition while 10 & 20 are operands.

Types of Operators in Python :

Python language supports the following types of operators.

1. Arithmetic Operators
2. Comparison (Relational) Operators
3. Assignment Operators
4. Logical Operators
5. Bitwise Operators
6. Membership Operators
7. Identity Operators

Arithmetic operators

- Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication, etc.

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
//	Floor division	x // y
**	Exponent	x**y (x to the power y)

Comparison operators

- Comparison operators are used to compare values. It returns either TRUE or FALSE according to the condition.

Operator	Name	Example
>	Greater than	x > y
<	Less than	x < y
==	Equal to	x == y
!=	Not equal to	x != y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical operators

- Logical operators are the AND, OR, NOT operators.

Operator	Meaning	Example
and	True if both the operands are true	x>0 and y<10
or	True if either of the operands is true	x>45 or y<60
not	True if operand is false	not x

Bitwise operators

- Bitwise operators act on operands as if they were strings of binary digits.
- They operate bit by bit, hence the name bitwise.

Operator	Meaning	Example
&	Bitwise AND	x & y = 0 (0000 0000)
|	Bitwise OR	x | y = 14 (0000 1110)
~	Bitwise NOT	~x = -11 (1111 0101)
^	Bitwise XOR	x ^ y = 14 (0000 1110)
>>	Bitwise right shift	x >> 2 = 2 (0000 0010)
<<	Bitwise left shift	x << 2 = 40 (0010 1000)

Assignment operators

- Assignment operators are used in Python to assign values to variables.
- a=5 is a simple assignment operator that assigns the value 5 to the variable a.
- There are various compound operators in Python like a+=5 that adds to the variable and later assigns the same. It is equivalent to a=a+5

Operator	Example	Equivalent to
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5

Special operators

- Python language offers some special types of operators like the identity operator or the membership operator.
- They are described below with examples.

Identity operators

- **is** and is **not** are the identity operators in Python.
- They are used to check if two values are located on the same part of the memory.

Operator	Meaning	Example
is	True if the operands are identical	x is True
is not	True if the operands are not identical	x is not True

Membership operators

- **in** and **not in** are the membership operators in Python.
- They are used to test whether a value or variable is found in a sequence (string, list, tuple, set and dictionary).
- In a dictionary we can only test for presence of key, not the value.

Operator	Meaning	Example
in	True if value is found in the sequence	5 in x
not in	True if value is not found in the sequence	5 not in x

Basically in Python data type is divided into two categories :

1. Mutable Data Types
2. Immutable Data Types

Mutable Data Types :

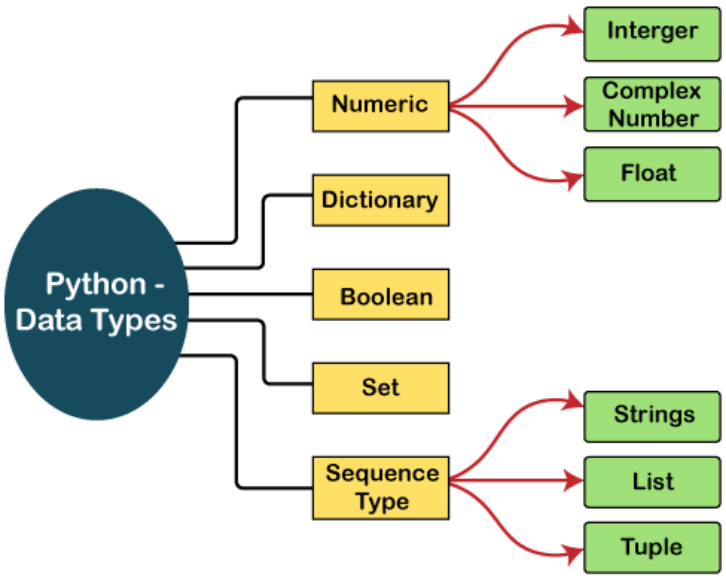
- Those data type whose value can be changed after initialization.
- We can change the object's values, such as field and states, after the object is created.
- Example : list, dictionary etc.

Immutable Data Types :

- Those data types whose value can not be changed after initialization.
- We can not change anything once the object is created.
- Example : int, float, complex, string, tuple, set etc.

Data types in Python :

1. Numeric type
   a. Integer
   b. Float
   c. Complex number
2. Dictionary
3. Boolean
4. Set
5. Sequence type
   a. String
   b. List
   c. Tuple



Numeric data types :

- In Python, numeric data type represent the data which has numeric value.
- Numeric value can be integer, floating number or even complex numbers.
- Python provides the **type()** function to know the data-type of the variable.
- Python supports three types of numeric data.

Integers

- An integer is a whole number, negative, positive or zero.
- Python has no restriction on the length of an integer.
- Example : 10, 2, 13, 29, -20, -150 etc.

Floating Point Numbers

- Floats are decimals, positive, negative and zero.
- Floats can also be represented by numbers in scientific notation which contain exponents.
- It is accurate upto 15 decimal points.
- Example : 1.9, 9.902, 15.2, etc.

Complex Numbers

- A complex number contains an ordered pair, i.e., x + iy where x and y denote the real and imaginary parts, respectively.
- The complex numbers like 2.14j, 2.0 + 2.3j, etc.

Sequence Type

- In Python, sequence is the ordered collection of similar or different data types.
- Sequences allows to store multiple values in an organized and efficient manner.
- There are several sequence types in Python :
  - String
  - List
  - Tuple

String

- In Python, Strings are arrays of bytes.
- It is a collection of one or more characters put in a single quote, double-quote or triple quote.
- In python there is no character data type, a character is a string of length one.
- The operator + is used to concatenate & the operator * is known as a repetition operator.
- Example :

  "CTH" + "EDUCATION" returns "CTH EDUCATION".

  "CTH" *2 returns 'CTH CTH'.

List :

- Lists are similar to arrays in C.
- The list can contain data of different types.
- The items stored in the list are separated with a comma and enclosed within square brackets [ ].
- We can use slice [:] operators to access the data of the list.
- The concatenation operator (+) and repetition operator (*) works with the list in the same way as they were working with the strings.

Example : list1 = [1, "hi", "Python", 2]

Tuple

- A tuple is similar to the list.
- It is also contain the collection of the items of different data types.
- The items of the tuple are separated with a comma and enclosed in parentheses ( ).
- The only difference between tuple and list is that tuples are immutable.
- Tuples cannot be modified after it is created.

Example : tup = ("hi", "Python", 2)

Dictionary :

- In Python dictionary is an unordered collection of data values.
- It is like an associative array or a hash table where each key stores a specific value.
- Key can hold any primitive data type, whereas value is an arbitrary Python object.
- The items in the dictionary are separated with the comma and enclosed in the curly braces { }.
- It is a mutable data type.

  Example : d= {1: 'CTH', 2: 'EDUCATION'}

  d={ 'Course_name' : 'Python', Course_duration : '2 Month'}

  Print(d['Course_name'])

Boolean :

- It provides two values, True and False.
- These values are used to determine the given statement true or false.
- It denotes by the class bool.
- True can be represented by any non-zero value or 'T' whereas false can be represented by the 0 or 'F'.

Set :

- Set is the unordered collection of the data type.
- It is mutable and has unique elements.
- In set, the order of the elements is undefined, it may return the changed sequence of the element.
- The set is created by using a function **set( ),** or a sequence of elements is passed in the curly braces and separated by the comma.

Example :

```
set1 = {'apple', 2, 3,'Python'}
```

List Slicing :

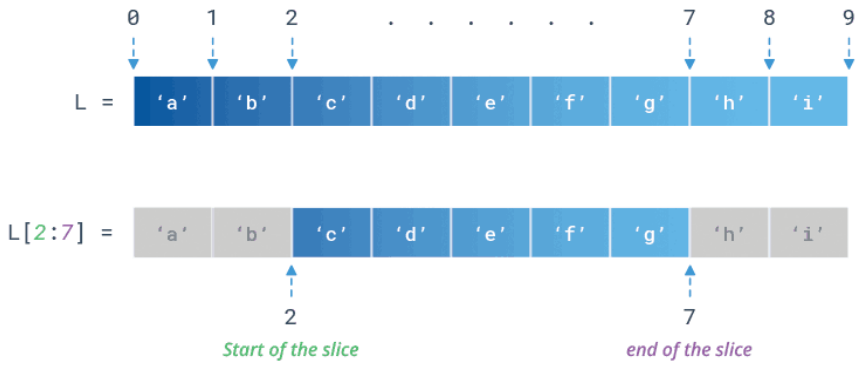
- To access a range of items in a list, you need to slice a list.
- Colon(:) is used to repersents the simple slicing operator :
- With this operator you can specify where to start the slicing, where to end and specify the step.

Syntax :

L [start : stop : step]

Example :

```
L = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
print(L[2:7])
```



Hash( ) :

- Hash method in Python is a module that is used to return the hash value of an object.
- Hash values are just integers that are used to compare dictionary keys during a dictionary look quickly.
- The hash() method takes a single parameter:
- Hash() method only works for immutable objects as tuple.

Syntax :

hash(object)

Associative array :

- The dictionaries are the Python implementation of an abstract data type, known as an associative array.
- Associative arrays consist dictionaries of (key, value) pairs, such that each possible key appears at most once in the collection.
- Any key of the dictionary is associated to a value.
- The values of a dictionary can be any type of Python data.
- So, dictionaries are unordered key-value-pairs.
- Dictionaries are implemented as hash tables, and that is the reason why they are known as "Hashes".

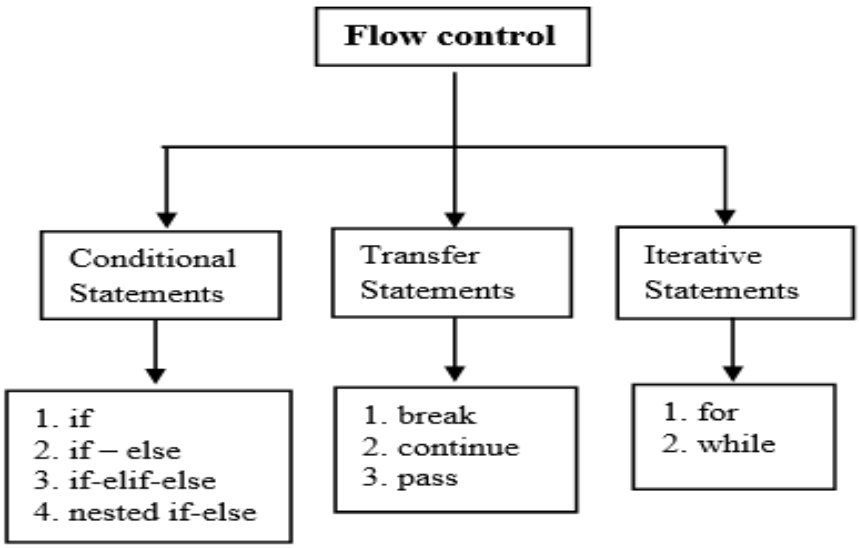
Unit – 02 : Control Structures

- Conditional blocks using if, else and else if.
- For loops and iterations
- while loops, Loop manipulation using continue break and else (and pass in Python).
- Programming using conditional and loops block.

1. Define control structures in python. Explain its type in details.
2. Explain if statement and if-else statements in python with example.
3. Define for loop and while loop in python. Write a program to elaborate it.
4. Explain transfer control statements(continue, break & pass) with example.
5. Write a program to find greater between two numbers in python.
6. Write a program to check whether the given number is even or odd.
7. Write a program to check whether the given number is even or odd.
8. Write a Python Program to Find the Sum of Natural Numbers.
9. Write a Python Program to Find the Factorial of a Number.
10. Write a Python Program to Display the multiplication Table.
11. Write short notes on :
    a. Indentation
    b. Range function
    c. Input function

What is control structure in python ?

- It is a structure which controls the flow of program code execution.
- Control structures are responsible for deciding the flow of program to generate desired output.
- They are also known as Python control flow statements.
- The control flow of a Python program is controlled by conditional statements, loops, and function calls.
- Python has three types of control structures :
  1. **Conditional(Selection)** - Used for decisions and branching
  2. **Transfer statements –** Transfer the flow control
  3. **Repetition(Itration)** - Used for looping.



What is Indentation Python ?

- Indentation is the white spaces at the beginning of a code line.
- Without indentation, Python does not know which statement to execute next or which statement belongs to which block.
- In Python, the indentation is mandatory concept when writing a python code; otherwise, the python interpreter throws Indentation Error.

Example :

```
n = 10
if n>5:
   print "n is greater than 5"
else:
   print "n is not greater than 5"
```

Conditional statements :

- The selection statement allows a program to test the conditions and execute instructions based on which condition is true.
- The selection statements are also known as Decision control or branching statements.
- Some selection statements are :
  - Simple if
  - if-else
  - nested if
  - if-elif-else

if statement :

- If statements are conditional or selection statements.
- It help us to run a particular code, but only when a certain condition is true.
- If gives the output only when condition is true otherwise no output.

Syntax :

```
if condition:
    statement(s)
```

Example :

```
n = 10
if n % 2 == 0:
    print("n is an even number")
```

Output : n is an even number

if-else :

- If-else statement check the condition and execute the body of if when the condition is True.
- If the condition is false, then the body of else is executed.

Syntax :

```
if condition:
    Body of if
else:
    Body of else
```

Example :

```
n = 5
if n % 2 == 0:
        print("n is even")
else:
        print("n is odd")
```

Output : n is odd

if-elif-else :

- The elif is short form of else if.
- It allows us to check for multiple expressions.
- If the condition of if is false, it checks the condition of the next elif block and so on.
- If all the conditions are false, then the body of else is executed.
- Only one block among the several if...elif...else blocks is executed according to the condition.
- The if block can have only one else block but it can have multiple elif blocks.

Syntax :

```
if condition:
    Body of if
elif condition1:
    Body of elif
else:
   Body of else
```

Example :

```
per = 55
if per>=60:
  print(“First Division")
elif per>=45:
   print(“Second Division")
elif per>=30:
   print(“Thord Division")
else:
   print(“Fail")
```

nested if :

- Nested if statements are if...elif...else statement inside another if...elif...else statement.
- This is called nesting in computer programming.
- Any number of these statements can be nested inside one another.
- Indentation is the only way to figure out the level of nesting.

Example :

```
a = 5
b = 10
c = 15
if a > b:
  if a > c:
    print("a is greater")
  else:
     print("c is greater")
elif b > c:
   print("b is graeter")
else:
    print("c is grater")
```

Output : c is graeter

range() function in python :

- The range( ) function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.
- The range( ) function can take a maximum of three arguments: start, stop and step.
- The start and step parameters in range( ) are optional.

Syntax :

range(start, stop, step)

Example1 :

range(5)

start=0

stop=4(condition<5)

step= increment by 1

Example2 :

range(1, 5)

start=1

stop=4(condition<5)

step= increment by 1

Example3 :

range(1, 5, 2)

start=1

stop=4(condition<5)

step= increment by 2

Parameter	Description
start	Optional. An integer number specifying at which position to start. Default is 0
stop	Required. An integer number specifying at which position to stop(not included).
step	Optional. An integer number specifying the incrementation. Default is 1

Transfer statements in Python :

- It is used to control the order of execution of the program based on the logic.
- There are 3 types of transfer statements in Python :
  1. Continue
  2. Break
  3. Pass

Continue Statement :

- When the python interpreter find the continue statement in a program, then it will skip the statements which are present and proceed with the next iterations.
- The continue keyword is used for continue statement.
- Continue statement is used in a for loop or a while loop.

Syntax:

continue

Example :

```
for i in range(10):
if(i==3):
        continue
print("Value of i=", i)
```

Output :

Value of i= 0
Value of i= 1
Value of i= 2
Value of i= 4
Value of i= 5
Value of i= 6
Value of i= 7
Value of i= 8
Value of i= 9

- In the above example during the second iteration if the condition evaluates to true, then it will execute the continue statement.
- So whatever statements are present below, it will be skipped, hence value 3 is not printed.

Break Statement :

- The break statement is used to terminate the loop.
- The control of the program will come out from the loop when break statement encounters.

Syntax:

break

Example :

```
for i in range(10):
if(i==3):
    break
print("Value of i=", i)
```

Output :

Value of i=0
Value of i=1
Value of i=2

Pass Statement :

- When the pass statement is executed, nothing happens.
- It can be used inside loop or in if statement to represent no operation.
- Pass is useful when we need statement is syntactically correct but no operation can performed.
- Pass keyword is used for pass statement.

Syntax :

pass

Example :

```
for i in range(10):
if(i==3):
    pass
print("Value of i=", i)
```

Output :

Value of i= 0
Value of i= 1
Value of i= 2
Value of i= 3
Value of i= 4
Value of i= 5
Value of i= 6
Value of i= 7
Value of i= 8
Value of i= 9

Itrative statements :

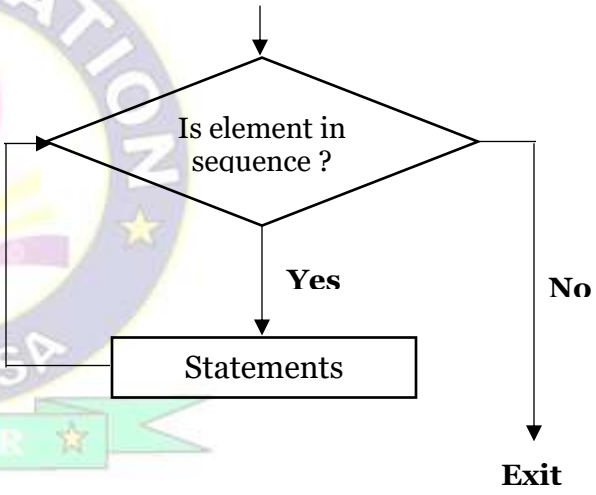
- An iterative statements are also known as repetative statement.
- It is used to repeat a group of instructions multiple times.
- We can repeat a single instruction multiple times using loops.
- There are two types of loop in Python
  1. for-loop(Definite loop)
  2. while-loop(Indefinite loop)

for loop :

- for loops are used when you want to repeat a block of code fixed number of times.
- For loop is also known as definite loop.
- The for loop in Python is used to iterate over a sequence (list, tuple, string).

Syntax of for Loop :

for variable_name in **sequence**:
    statements(s)



Example :

```
String=”CTHEDUCATION”
for char in String:
    print(char)
```

Write a program in python to find the sum of all numbers stored in a list :

```
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]
sum = 0
for i in numbers:
    sum = sum + i
print("The sum is", sum)
```

Syntax of for Loop using range() :

```
for variable_name in range(parameters):
        statements(s)
```

Example :

```
for n in range(5)
    print(n)
```

Write a program in python to print the table of n using for loop :

```
n=int(input("Enter the number"))
for i in range(1,11):
        print(n,"*",i,"=",n*i)
```

while loop :

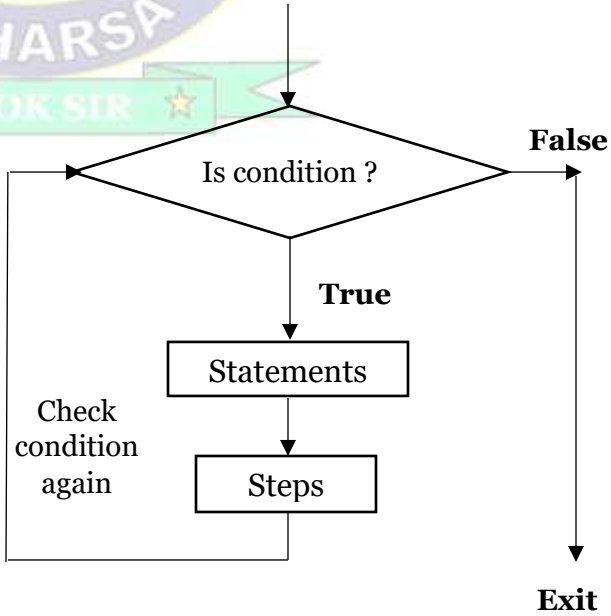
- The while loop is used when we did not know how many iterations are required.
- While loop is also known as indefinite loop.
- While loop is used to execute a block of statements repeatedly until a given condition is true.
- If the condition becomes false, the control of loop immediately come out from the loop.

Syntax :

```
while condition:
        statement(s)
```

Example :

```
m = 5
i = 0
while i < m:
   print(i)
   i = i + 1
```



Output :

```
0
1
2
3
4
```

A python program to print the name of your college 10 times using while loop :

```
i=1
While i<=10
        Print("College Name")
        i=i+1
```

Difference between for loop & while loop :

For loop	While loop
The for loop is used only when we already know the number of iteration.	The while loop is used when we did not know how many iterations are required.
For loop is also known as definite loop.	While loop is also known as indefinite loop.
Syntax : for value in range: <statement>	Syntax : while condition: <statement>
If there is no condition then the loop iterates infinite time.	But in the case of while if the condition is missing then it shows an error.
In the loop statement, it is always written at the top.	But in the case of while we can store statements anywhere in the loop body.
Example : for n in range(10): print(n) Output : 0 to 9	Example : n = 2 while n < 6: print(n) n = n + 1 Output : 2, 3, 4, 5

Write a program in python to find the sum of first 15 natural numbers.

```
s=0
for i in range(1, 16):
        s=s+i
print("Sum is = ", s)
```

Write a program in python to print table.

```
n=int(input("Enter the no which you want to print the table="))
for i in range(1, 11):
        print(n,"*", i "= ", n*i)
```

Write a program in python to calculate factorial.

```
n=int(input("Enter the no which you want to print the table="))
f=1
for i in range(1, n+1):
        f = f * i
print("Factorial is = ", f)
```

Practice questions :

1. A python program to print the name of your college 10 times using while loop .
2. Write a program to print the first 15 natural numbers using while loop.
3. Write a program to print the first 50 natural numbers in reverse order using while loop.

Unit – 03 : Functions, Modules and Packages

- Organizing Python codes using functions
- Organizing Python projects into modules
- Importing own module as well as external modules
- Understanding Packages

Questions to be discussed :

1. Define function in python. What are the advantages of using function ?
2. How to create a function in python explain with example ?
3. Explain different types of function in python.
4. What is module ? How to create module in python ?
5. Define the term package in python.
6. Write a python program to find the GCD(HCF).
7. Write a Python program to convert decimal to binary number.
8. Write a Python Program To Find ASCII value of any character.
9. Write a Python program to Display Calendar.
10. Write a python program for Simple Calculator.

What is Function in Python ?

- A function is a group of statements that together perform a specific task.
- In other words, a group of lines with some name is called a function.
- A function is a block of code which executes only when it is called.
- A function can be create in python using the **def** keyword.
- We can pass data into a function & these data is known as parameters.
- It is used to perform certain actions, and they are important for reusing code.
- If we call a function, the statements inside the function body are executed.
- They are not executed until the function is called.

How to create a function in python ?

- We can create a function using **def** keyword followed by the function name, parenthesis & colon.
- Any input parameters or arguments should be placed within these parentheses.
- Parameters through which we pass values to a function are optional.
- A colon (:) indicate the end of the function header.
- The statements in the block of the function must be indented.
- An optional return statement to return a value from the function.

Syntax :

```
def function_name(parameters):
    statements
    return expression
```

Example :

```
def  xyz( ) :
    Print("Welcome to CTH EDUCATION")
```

How to call a function in python ?

- Without calling a function can't gives any result.
- Once we have defined a function, we can call it from another function or program.
- To call a function we simply type the function name with appropriate parameters.

Syntax :

```
function_name()
function_name(arg1, arg2,.....)
```

Example :

```
Add()
Sum(20)
Add(10, 20)
Item("CTH EDUCATION")
```

Example :

```
def xyz( ) :
        Print("Welcome to CTH EDUCATION")
xyz()
```

Output : CTH EDUCATION

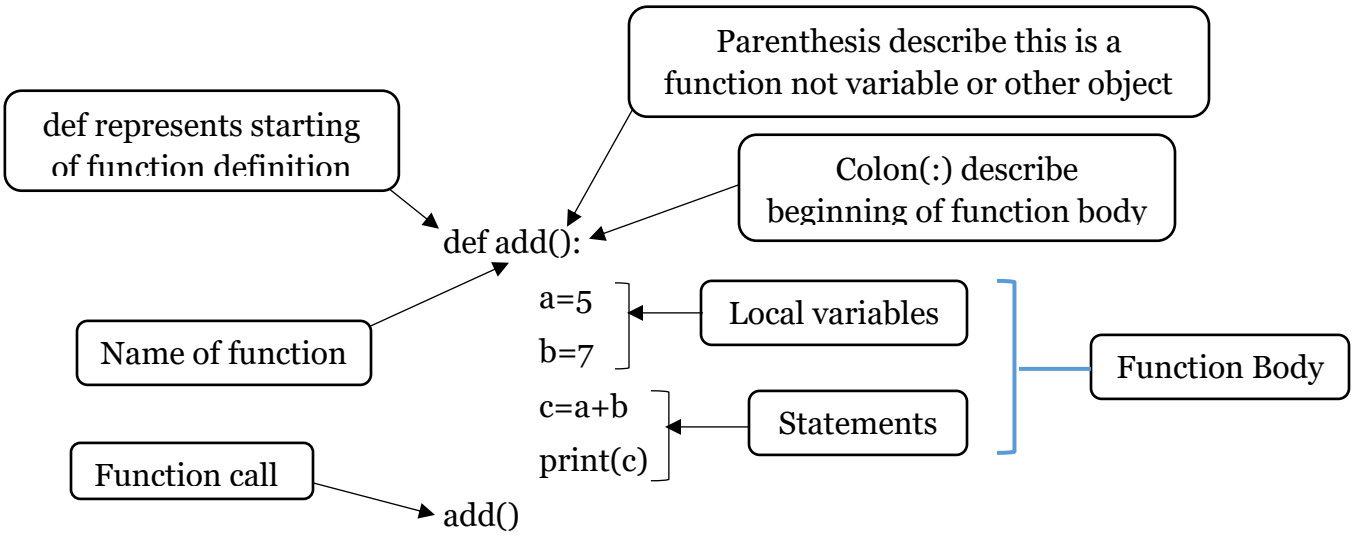
Example1 :

```
Def add():
    a=5
    b=7
    c=a+b
    print(c)
add()
```

Example2 :

```
Def add(b):
    a=5
    c=a+b
    print(c)
add(7)
```

Function explanation :



How Function works in Python ?

```
def add():
    a=5
    b=20
    c=a+b
    print(c)
add()
```

```
def add(b):
    a=5
    c=a+b
    print(c)
add(10)
```

Types of functions in python :

- Python has many built-in functions like print(), upper(), input() etc. but you can also create your own functions.
- So, there are two types of functions :
  1. Built-in function(Pre-defined functions)
  2. User-defined functions.

Built-in functions :

- Built-in functions are the functions that are already defined in python.
- We only need to remember the names of functions and use frequently.
- As these functions are already defined so we do not need to define only use.
- Below are some **built-in** functions of Python.

Function name	Description
len()	It returns the length of an object/value.
list()	It returns a list.
max()	It is used to return maximum value from a sequence (list,sets) etc.
min()	It is used to return minimum value from a sequence (list,sets) etc.
open()	It is used to open a file.
print()	It is used to print statement.
str()	It is used to return string object/value.
sum()	It is used to sum the values inside sequence.
type()	It is used to return the type of object.
tuple()	It is used to return a tuple.

Example :

```
x = [1, 2, 3, 4, 5]
print(len(x))
print(type(x))
```

Output:

```
5
<class 'list'>
```

User-Defined Functions :

- Those functions which are created by user according to their need is called user defined function.
- It is used to reduce the complexity of big problems.
- def keyword is used to create user defined function.

Example :

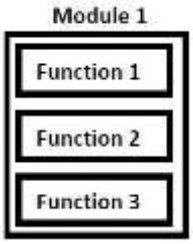
```
x = 7
y = 5
def add():
    print(x+y)

add()
```

Output : 12

Advantages of Functions in Python :

- Reuse of code(write the code once & use it many times as you want).
- Divide large program into smaller program so, easy to debug.
- User can remove or add new features to a function any time.
- Reducing duplication of code.
- Decomposing complex problems into simpler pieces.
- Python is dynamic in nature so, It is possible to redefine an already defined function.

What are Python Modules ?



- A group of functions saved to a file , is called Module.
- A module is a collection of function, variable or classes.
- It just like a library which can reuse multiple times while creating once.
- In Python, Modules are simply files with the ".py" extension.
- A module contains python code that can be imported inside another python program.
- Here the core idea is to minimize the code, and if we create modules, it doesn't mean we can only use it for this program, but we can even call these modules for other programs as well.

How to create Python Modules ?

- To create a module, we have to save the code with the file extension **".py"**.
- Then, the name of the Python file becomes the name of the module.

Example :

addition.py

```
def sum(a, b):
        print("The sum is=", a+b)
```

How to use Python Modules ?

- If you create a module then we can use this module using import keyword in another program.
- Here, create another file with the name my.py and use module in it.

Syntax :

my.py

```
import module_name
module_name.function_name
```

Example :

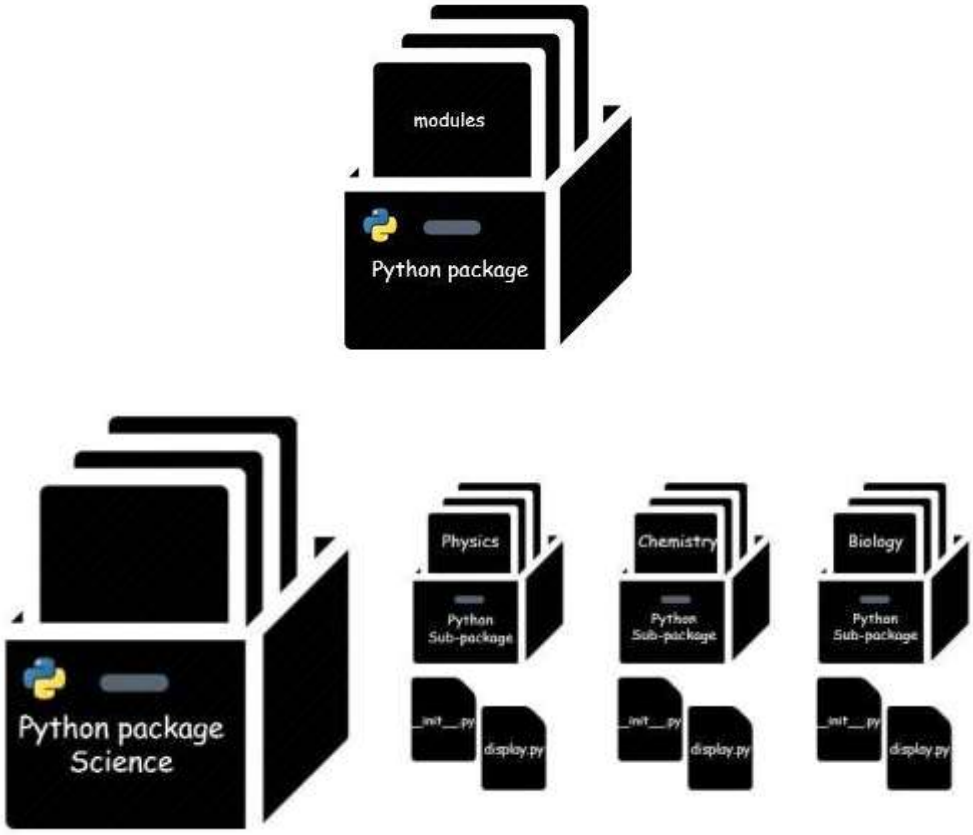
my.py

```
import addition
addition.sum(12, 13)
```

Output : The sum is = 25

What is package Package in python ?

- A package is a collections of modules.
- In other words, it is called a directories or folder.
- This directory contains Python modules and also having ___init___.py file by which the interpreter interprets it as a Package.
- The package also contains sub-packages inside it.
- The modules can also be arranged in hierarchy of folders inside a package.
- A Python **package** is nothing but a collection of modules along with a ___init___.py file.



ord() function in Python :

- Python ord() function returns the Unicode code from a given character.
- This function accepts a string of unit length as an argument and returns the ASCII value of given character.
- In other words, given a string of length 1, the ord() function returns an integer representing the Unicode code point of the character

Syntax:

ord(character)

Write a python program to find the GCD(HCF) of a number :

```
def hcf():
   a=int(input("Enter the value of a="))
   b= int(input("Enter the value of b="))
   while a!=b:
      if a>b:
         a=a-b
      else:
         b=b-a
   print("The hcf is=",a)
hcf()
```

Write a Python program to convert decimal to binary number.

```
def Decimal_to_Binary():
    n=int(input("Enter the decimal value of n="))
      while n>0:
        r=n%2
        print(r)
        n = n // 2
Decimal_to_Binary ()
```

Python Program To Find ASCII value of a character

```
char = input("Enter a character : ")
        print("The ASCII value of '" char "' is ", ord(char))
```

Python Function to Display Calendar

```
# First import the calendar module
import calendar
year = int(input("Enter the year : "))
month = int(input("Enter the month : "))
mycal=calendar.month(year, month)
print(mycal)
```

Write a python program for Simple Calculator :

```
def add(P, Q):
        return P + Q
def subtract(P, Q):
        return P - Q
def multiply(P, Q):
        return P * Q
def divide(P, Q):
        return P / Q
print ("Please select the operation.")
print ("a. Add")
print ("b. Subtract")
print ("c. Multiply")
print ("d. Divide")

choice = input("Please enter choice (a/ b/ c/ d) : ")

n1 = int (input ("Please enter the first number : "))
n2 = int (input ("Please enter the second number : "))

if choice == 'a':
        print (n1, " + ", n2, " = ", add(n1, n2))

elif choice == 'b':
        print (n1, " - ", n2, " = ", subtract(n1, n2))

elif choice == 'c':
        print (n1, " * ", n2, " = ", multiply(n1, n2))
elif choice == 'd':
        print (n1, " / ", n2, " = ", divide(n1, n2))
else:
        print ("This is an invalid input")
```

Unit – 04 : File I/O, Text Processing, Regular Expressions

- Understanding read functions
- Understanding write functions
- Powerful pattern matching and searching Power of pattern searching using regex.

Questions to be discussed :

1. What is files in python? Explain types of files.
2. Discuss about file handling. Explain operations performed on files.
3. Explain different mode of operations supports in python.
4. What is Regular Expression in Python?

What is file in python ?

- File is the collection of data that is available to a program.
- We can retrieve and stored in a file whenever we required.

Advantage of files :

- Data stored in file permanently unless someone remove it.
- Stored data can be shared.
- It is possible to update and remove the data in file.

There are two types of files :

1. Text file
2. Binary file

Text file :

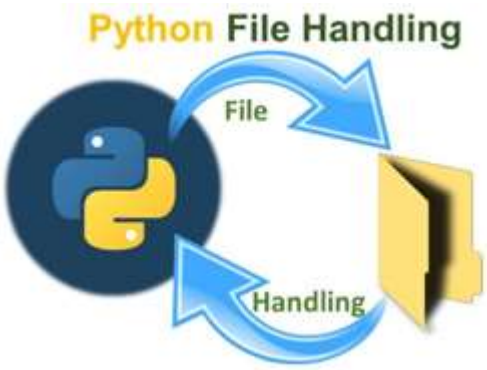
- It stores data in the form of characters.
- It is used to store characters and string.

Binary file :

- It stores data in the form of bytes.
- It is used to store text, images, pdf, csv, audio, video etc.

File Handling in Python :

- Python supports file handling and allows users to handle files.
- To read and write files, along with many other file handling options, to operate on files.
- Python treats files as a text or binary and this is important.
- Each line of code includes a sequence of characters and they form a text file.
- Each line of a file is terminated with a special character, called the EOL or End of Line {,}.



File operations :

- When we want to read or write a file, we need to open it first.
- When we are done, it needs to be closed.
- In Python, a file operation is :
  1. Open a file
  2. Read
  3. Write
  4. Close the file

Working of open() function :

- If we want to use a file or its data first, we have to open that file.
- Open() function is used to open a file.
- For this, we should use python's inbuilt function open() but at the time of opening, we have to specify the mode, which represents the purpose of the opening file.

Syntax :

f = open(filename, mode)

Example :

f = open('student.txt', 'w')

Where the following mode is supported:

1. **r :** Read only
2. **w :** Write only
3. **a :** Append and it won't override existing data.
4. **r+ :** Read as well as write
5. **w+ :** Write as well as read and it will override existing data.
6. **a+ :** Append & read data from file and it won't override existing data.

Reading a file :

- There are three ways to read data from a file :
  1. read()
  2. readline()
  3. readlines()

read() :

- The read() method in Python is a pre-defined function.
- Read() function returns the read data in the form of a string.

  Syntax :

  file.read(n)

  - Where 'n' is the number of bytes to be read from the file.
  - In the case where n is not specified, the read() function reads the whole file.

readline() :

- readline() is yet another pre-defined method in Python.
- It returns a read line in the form of a string.

  Syntax :

  file.readline(n)

readlines() :

- readlines() reads all the lines present inside a specified file and returns a list containing the string forms of the read lines.
- Syntax,

  file.readlines()

Writing a file :

- There are two ways to write in a file.
  4. Write()
  5. Writelines()

write() :

- Inserts the string str1 in a single line in the text file.

  Syntax :

  File_object.write(str1)

writelines() :

- Used to insert multiple strings at a single time.

  Syntax :

  File_object.writelines(L) for L = [str1, str2, str3]

What is Regular Expression in Python?

- A regular expression, is a sequence of characters that forms a search pattern.
- In python regular expression is denoted as RE or RegEx.
- RegEx can be used to check if a string contains the specified search pattern.
- It is a collection of pre-defined functions.
- It is used to process the input text.
- It is imported through **re** module.
- Python provides some function for pattern matching :
  1. re.match()
  2. re.search()
  3. re.findall()
  4. re.split()

match() : To test the input string start with the specified pattern or not.

search() : To test the specified pattern is present or not in the given string.

findall() : To find duplicates for specified pattern.

Split() : To split the input string

Some regex pattern with description :

Pattern	Description
^	Matches beginning of line.
$	Matches end of line.
.	Matches any single character except newline. Using m option allows it to match newline as well.
[...]	Matches any single character in brackets.
[^...]	Matches any single character not in brackets

Example :

^a...s$

- The above code defines a RegEx pattern.
- The pattern is: any five letter string starting with ‘a’ and ending with ‘s’.

Expression	String	Matched?
^a...s$	abs	No match
	alias	Match
	abyss	Match
	Alias	No match
	An abacus	No match

Unit – 05 : Frameworks

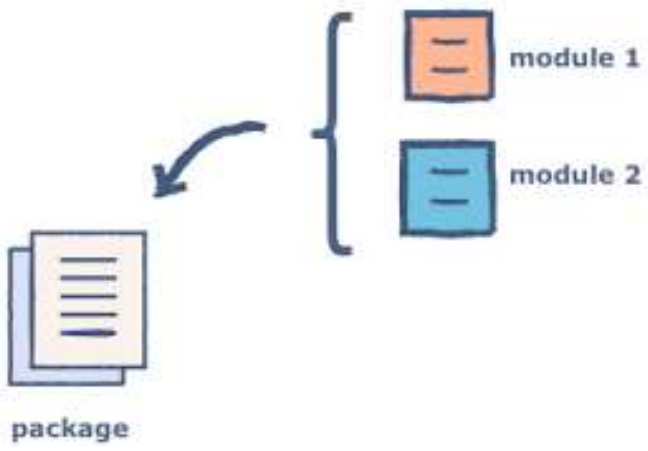
- Frameworks –
  - Web2Py,
  - Django (any one of these or any other).

Questions to be discussed :

1. What are frameworks ? What are the advantage of using frameworks in python ?
2. Differentiate between micro framework and full stack framework.
3. Write short notes :
   a. Web2Py
   b. Django

Python module :

- In Python, Modules are simply files with the ". py" extension containing Python code that can be imported inside another Python Program.



What are Frameworks in Python?

- A framework is a collection of modules or packages.
- It helps in writing web applications or web development.
- Framework makes the developers life easier by giving them a structure for app development.
- They provide common patterns for faster, scalable reliable & easily maintained web application.

Advantages Of Frameworks :

1. Open-source
2. Good documentation
3. Efficient
4. Secure
5. Integration

Important Frameworks In Python :

1. Django
2. Web2Py
3. Flask
4. Bottle
5. CherryPy
6. TurboGears
7. Zope2
8. Grok
9. Bobo
10. Web.py

There are two types of Python Frameworks :

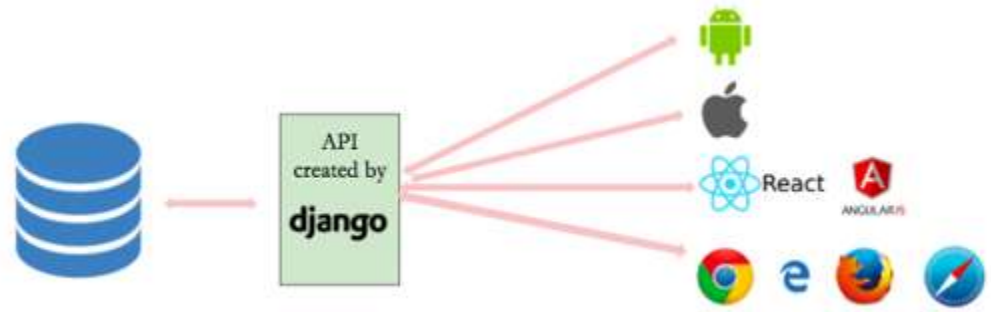
1. Full Stack Frameworks
2. Micro-frameworks

Difference between a micro-framework and a full-stack framework?

Micro-framework	Full-stack framework
Simple and easy to use.	Complex and does the heavy lifting
URL routing is RESTful often	Need not be RESTful
Prefer to develop small application.	Prefer to develop large application.
A good choice for small applications.	Can be used to make any applications.
Use WSGI(Web Server Gateway Interface) and work through HTTP request/response.	Provide libraries, template engines, database management etc.
Examples: Flask, Bottle, CherryPy, Bobo, Web.py	Examples: Django, TurboGears, Web2py, Zope2, Grok.

What does REST stand for in Python?

- REST stands for representational state transfer.
- A REST API is an application programming interface.
- It is also known as RESTful API.
- It was created by computer scientist Roy Fielding.
- REST API is a way of accessing web services in a simple and flexible way without having any processing.



Django :

- Django is a free and open-source full-stack python framework.
- It includes all the necessary features by default.
- Django is used for the fast development of APIs and suave design of web applications?
- Django apps are fast, secure, and scalable.
- Django is favorable for web apps that are strong in handling back-ends.
- Also, when you have a lot of traffic/transactions taking place, Django is a good choice.
- You can choose Django when building inventory management, CRM systems, and social networking sites.

Features of django web framework:

- Authentication
- URL routing
- Template engine
- ORM(Object Oriented Mappers)
- Database Schema migrations

Web2Py :

- Web2Py is open source, scalable and a full-stack framework.
- It does not support python3 and comes with its own web based IDE which also includes a separate code editor, debugger and one click deployment.

Features of Web2Py framework:

- It does not have any prerequisites for installation and configuration.
- It has the ability to run on different platforms.
  - Example- windows, mac, linux etc.
- Comes with an ability to read multiple protocols
- Web2Py provides data security against vulnerabilities and other malicious attacks.