**Python Introduction**

Python is a computer language which is built on C but it is an interpreted language

Python is a multi-paradigm language (which has different types of uses)

Python has different types of modules for our different needs.

They can be downloaded with the help of pip command in the PowerShell.

Note. While importing the modules from the Python interpreter you need to type the module name. function in module.

Features of Python:

Python is very simple to use and it does not need extreme brain power to learn it

Python is open source

Python is portable. It is embeddable and extensible meaning any language can be embedded into python

python is interpreted language

Python is a vast language and can handle different types of files and one of them is csv file

CSV file is nothing but a file in which the data is separated by commas and the data can be of any form either lists, tuples, dictionaries or any other things.

To handle CSV files python has a module named CSV to handle all the csv file functions.

CSV files give the programmer ability to parse CSV files.

Note: parsing is nothing but splitting a sequence of data into different formats In the above scenario CSV files are separated by commas and the programmer can use CSV module to parse the file by taking into consideration of commas

**Python keywords:**

|  |  |
| --- | --- |
| Not in | True |
| No | False |
| Or | Break |
| And | Continue |
| Input | def |
| Print | If |
| For | Else |
| Elif | \_\_doc\_\_ |
| In | Try |
| Except | Exception |
| as | with |
|  |  |

Python modules

Python has some package managers the package managers in python are pip and these package managers install modules which help us to write code effectively modules are installed from the internet

**Input:** input in Python is the read function which the programmer uses to get input from the user.

Note: while giving input Python takes input as a string so typecasting is necessary.

Note: We don’t need to use semicolons to terminate the statement like in C.

**Python Operators**

**Operators:** In python there are different types of operators

1. Arithmetic operators
2. Assignment operators
3. Comparison operators
4. Logical operators
5. Identity operators
6. Membership operators
7. Bitwise operators

**Arithmetic operators:** Arithmetic operators help in numeric calculations

1. + add
2. – sub
3. \* mul
4. / div in float
5. // div in int
6. \*\* exponent
7. % modulus

**Assignment operators:**

1. =
2. +=
3. /=
4. -=
5. \*=
6. %=
7. \*\*=

**Comparison operators:**

1. == comparison operators
2. != not equals to
3. >= greater than or equal
4. <= less than or equal
5. < less than
6. > greater than

**Logical operator:**

1. and
2. or

**Identity operator:**

1. is
2. is not

**Membership operator:**

1. in
2. not in

**Bitwise operator:**

1. &
2. |

**Comments**

**Comments:** Comments in Python language are written by using (***#)*** and by (”***”” code “””)*** symbol.

In python generally when we write something after a line it automatically jumps the cursor to the next line to avoid this we write (“code", end=””).

In python to print the escape sequence character then we need to give two (“\\”) to print.

**Variables**: variable is like a container which can store a value in it

**Typecasting**: Changing from one data type to another is called typecasting.

Syntax:

Newdatatype(var)

Ex : int(var)

To know the data type of the variable

type(var)

**Strings**

**String:** string is a datatype in Python which can hold many characters along with numbers in it.

**String slicing**: printing a particular thing in the string is called as string slicing.

Syntax:

Mystring=”hello”

print(mystr[0:2])

**output**

hel

**String functions:**

**Note:** In all these functions the str is referred to the variable name

1. Str.len() to find length of a string
2. Str[0:4:2] to print the elements in the string leaving 2 elements
3. Str[0:4:-1] to print the reverse of the string
4. str.isalnum() to check if string contains alpha numeric characters
5. str.endswith() to check if string ends with so and so element
6. str.count() to
7. str.captilize() to captilize the first element of the string
8. str.find() to find a particular character in the string
9. str.upper() to convert string into upper case
10. str.lower() to convert string into lower case
11. str.replace(old,new) to replace the old word with new one in the string
12. str.strip() to remove white spaces in the string
13. str.replace() to replace values in string with another.
14. Str.split() to split the sting into multiple strings and will be saved as a list.

**Lists**

**Lists:** lists in python are collection of heterogenous elements together.

Syntax:

listname[“element1”,”element2”……”elementn”]

**List functions:**

1. listname.sort() to sort the elements in the list
2. listname.reverse() to reverse the elements in the list

**list slicing:**

1. listname[0:3] to print the elements from 0 to 3

Note: the old list will not be changed.

By default, the the two parameters in the slicing are 0:max

**Extended slicing:**

1. Listname[::1] to leave elements while printing the list
2. Listname.append(no.) To append the list
3. Listname.insert{1,67} To insert new elements in the list

Note: first parameter is index and second parameter is the element.

1. Listname.isnumeric() To check the elements in the items are numeric

Note: will only work for strings.

Note: When two lists are added then two of them concatenate.

Note: when a list is multiplied by a value then the list will be repeated that many times

tuple

**Tuple:** Tuple in Python is heterogenous data elements which cannot be changed (immutable).

sets

**Sets**: sets are also like lists which can hold many values in a single variable but there are not indexable and not duplicable.

Syntax:

Tuplename{(element1, element2, element3…..elementn)

Note: If you are writing only one element in the tuple then put a comma after the element else python will take tuple as ordinary variable

arrays

arrays are created by using array module.

Syntax:

import array

arrayname = array.array(‘i’,[1,2,3,4,5]) from array module use array function along with type ( here i have given ‘i’ as type which means integer and then type the elements of array

to access elements from an array we use indexing and indexing starts from 0 to n-1 and there also exists negative indexing which indexes from right to left (last to first)

Just like functions in list there are functions for array also.

Functions in array:

pop() to remove elements from array

remove() to remove elements

insert() to insert data to array

extend() to extend array ( note: use [] while using extend )

append() to append info to array

We can also slice array like strings,list etc

Hash table and Hash map

hello

IF Else Elif

**If, else, elif control statements:** if,else,elif control statements in Python check for a condition in code if the condition is true then the code in the if section will get executed else then the code in the else condition will be executed.

If, else Syntax:

if condition: or x in list:

code

else:

code

elif syntax:

if condition: or x in list:

code

elif condition:

code

else:

code

Loops

**For Loops:** for loops in Python are the iterative statements which help the programmer to iterate code again and again until and unless the condition fails

Syntax:

For item in list:

Print(item)

**While Loops**:while loops in Python are the iterative statements which help the programmer to iterate code again and again until and unless the condition fails

Syntax:

While condition:

code

**Break and continue:** break is used to break the loop and continue is used to skip the iteration of the loop (the code will not be executed).

Syntax:

Break

Syntax:

Continue

Functions

**Functions:** Functions in Python enable user to create own code snippets which help to split up the code

Syntax:

def function\_name( arg1, arg2, arg3…..argn):

code function definition

function\_name(args) function call

Note: In python we don’t need to specify th return type as the python automatically takes the return type as none (null in java) if we didn’t return anything.

Note: If you want a function to give 2 arguments to a function but by default it takes 3 arguments then you can assign one argument as none and then you can pass 2 arguments to a function

example:

def add\_numbers(x, y, z=None):

if(z == None):

return x+y

else:

return x+y+z

add\_numbers(1,3)

output: 4

**Global and local variables:** In python while using functions there are both local and global variables and in function python gives more precedence to local variables and you cannot change the global variables in function

Syntax to access and change global variables:

Global var

**Docstrings:** docstrings are the first line of comments which are written in the function definition which enable programmer to know about the function without much time.

Syntax:

def function\_name( arg1, arg2, arg3…..argn):

“”” doc string “”” function definition

Function\_name( args ). \_\_doc\_\_ function call

Try and Except

**Try and except:**  try and except in Python is used to skip a piece of code which throws an error and which caused the code to discontinue by using try and except we can skip the code which cause the error and continue.

Syntax:

Try:

Error code

except Exception as (e): error message will go into e

print(e) printing of error message

code continuation….

Files

**File I/O:** Files in python are used by programmers to save a file which they want to reuse again and again

Note: for creating a file you need to create a file pointer and then you need to open the file in the file pointer

(You need to return your file open function to file pointer then you can use file functions)

**File modes:**

1. Read (“r”) Open file for reading
2. Write (“w”) Open file for writing
3. Exclusive creation (“x”) Open file if there is no file exists
4. Append (“a”) open file for appending
5. T mode (“t”) Text mode
6. B mode (“b”) open file for binary mode
7. Plus, mode (“+”) open file for update (read + write mode)

**File functions (for reading):**

1. Open for opening

Syntax:

Filepointer = open(“file\_name”,”mode”)

1. Read for reading the file

Syntax:

Filepointer.read()

1. Readline for reading line

Syntax:

Filepointer.readline()

1. Close for closing the file

Syntax:

Filepointer.close()

Note: whenever we open a file, we need to close the file to free up the resourced used by the file.

**File functions (for writing and appending):**

1. Write for writing to a file

Note: while writing if a file is not created then python will create a new file if a file exists then all the info will be deleted then the new info will be written on to it.

Syntax:

Filepointer.write(“ the matter you want to write to that file “)

Note: filepointer.write returns the len of the string you are writing to the file. For appending info to the file we use write function to append it to the file.

1. tell for printing control on which line it is

syntax:

filepointer.tell()

1. seek for printing the line in the file which we want to instead of printing the whole file

syntax:

filepointer.seek( line number)

**With Block:** we can open the file using with block also

Syntax:

With open(“filename.txt”,”mode”) as filepointer:

Code

Note: you don’t need to close the file after you opened with the with block

Functions (scope and global variables)

**Scope and Global variables:** In python while using the function a function can make its own local variables and these local variables cannot be used outside the function and even global variables cannot be used in the function to use them then the “global “key word must be used.

Syntax:

global global\_variable

Note: if the global variable doesn’t exist then the python will create the global variable. If you are creating a global variable and in the sub function of another function then the global key word won’t search for the variable in the parent function but it will go directly outside all the functions.

Recursions

**Recursions:** Recursions are nothing but solving problems by calling the function again and again inside it’s own definition. While doing recursions python will make frameworks in the memory ( stack ) and while calling the function in the definition the first iteration of the function won’t get deleted but instead the function will be called again and the memory will be allocated to it and it will go on till there is a stop then after stopping a function will return a value and the returned value will go on circulating till it reaches the first iteration of the recursion.

Note: To stop the iterations in the loops there are continue and break but to stop the recursions there is return statement if you want to stop the recursion you need to use the return statement.

Lambda

**Anonymous or Lambda function:** Anonymous or lambda functions are the function which have only one line in the definition and these are mostly used when there is need to create a function but not with full functions then the lambda functions are used.

Syntax:

variable = lambda arg1, arg2, argn…: function is created with the name variable.

Modules

**Modules:** Modules in Python are the pre-written code by the python developers which help us to create efficient programs**.**

Modules in python have inner modules to get more info about the modules search in internet

To use module:

Syntax method 1:

Import module\_name

To use functions in the module:

Syntax:

module\_name.function\_in\_module

Note: if you want to use functions in the module then you need to write module name. function in the module as the python will search the function in the main function if you write the module function directly.

Syntax method 2:

import module\_name as mod module will be imported as alias name as mod

syntax method 3:

from module\_name import \* from module all the functions will be imported

syntax method 4:

from module\_name import function\_in\_module from module only one module will be imported

string interpelation

**string interpretation:** string interpretation in Python are the string formatting tools in Python which help us to edit the strings.

Syntax:

f” strings { code to be added to the string} {} {} “

arbitrary and keyword arbitrary arguments

**\*args and \*\*kwargs:** \*args and \*\*kwargs in python are the new type of arguments.

\*args are called arbitrary arguments which help the programmer to send any number of arguments to the function without specifying the correct number of arguments to be sent to the function.

Note: arbritrary argument always merges the arguments into a tuple whenever you are sending the arguments to the function.

Syntax:

Def function(\*args)

Function(list) no of arguments in the list will be passed to the function as only one argument.

\*\*kwargs are called keyword arbitrary arguments which help the programmer to customly send the argument to the function without knowing the no. of argument of the function so the programmer can make a dictionary and create a keyword to send the argument to the function.

Syntax:

Def function(\*\*kwargs)

Function (dict )

Time module

**Time Module:** Time module in the Python is a module which has different functions in the module which helps us to calculate time taken to execute a code and much much more function available in the Time module.

Syntax:

Import time

Functions in time module:

1. time() counts ticks which are equal to one second.
2. Localtime() prints the time in the form of a tuple.
3. Asctime converts the time to a presentable format.
4. Sleep() if executed then the function will make to code to sleep for x seconds (x is taken as the argument in the function).

Note: whenever you are using the function in the module then it is mandatory to type the module\_name.funtion\_name else the function in the module will not be executed.

Pygame module

**Pygame module:** Pygame module in python is used to make games add sound effects to a game and much much more functions it is also used to make images, videos, play sound in python files.

Syntax:

Import pygame

Functions in pygame module:

Mixer() mixer is used to play sound

To play sound in the python file:

Mixer.init() to initialize the mixer

Mixer.music.load(“musicfile.mp3”) to load the music file

Mixer.music.play() to play the music file

Note: To play the music file you need to give it some time to play else music file will be executed and the program will stop working.

Python Numpy module

Python Numpy module is used to make arrays of different sizes. Their usage is extensively seen in scientific computations.

Numpy arrays are much more faster than regular lists and even take less space to compute.

Numpy module also has some special functions to work with arrays

Python Pandas Module

Python pandas module is used for data science

Python extensively uses Numpy arrays for faster computation.

Python has two primary data types 1) series 2) data frames

Series are nothing but similar to arrays in python they are contigous and hold homogenous elements in them but they are not horizontal but vertical

Data frames are 2 dimension dataframes which can hold heterogenous elements in them

Dataframes have their own index which generally start from 0 to 1 and are at the left side of the matrix

Dataframes is made using different types and amount of series attached to each other and these act as coloumns of data frame

Dataframe can be visualized as excel spreadsheets

We can also import and export to and from excel spreadsheets

Dataframes cannot be simple assigned to another variables like in python code in fact if you try to do then it will create a pointer to the original dataframe and whenever you try to change the pointer in turn you will be changing the original dataframe which is bummer because it causes dataloss

To stay away from this hassle we have .copy() function to copy one dataframe to another

And don’t try to change elements of dataframe as if we would do in arrays cause pandas module in python makes pointers to the newarrays which will be another hassle to stay away from this we have .loc function in pandas module.

Enumerate

**Enumerate:** Enumerate function in Python is used to select some pattern of numbers from the list to be printed or to do some other work.

Syntax:

Enumerate(index\_value,list) enumerate function takes both index value and list to filter the elements in the list to be printed or to be executed.

Note: Enumerate function is mostly used in the For loops.

Import: Import in Python is used to import different modules from the Python Interpreter to execute a particular type of code

Import first searches in the local directory in which all the python files are kept then import goes on searching all the directories.

If you only want to import only one function from the module then we can use

Form module\_name import Function\_name

If \_\_name\_\_ == \_\_main:

If \_\_name\_\_==\_\_main\_\_ : In Python is used in situations where we are importing our own python file for some purposes and while importing the file the whole of the file will be executed while importing the output of that file also will be imported along with the code so if you want to hide some functions while importing, the above function can be used.

Syntax:

If \_\_name\_\_== ‘\_\_main\_\_’:

Code to be hidden from importing

Join, Map, and Filter functions

**Join function:** Join function is an inbuilt function in Python which helps to randomly add to a list or any other thing.

Syntax:

Var = “thing”.join(list).

**Map:** Map function is a function is an inbuilt function which applies a function to total of the list.

Syntax:

Var = list(map(function\_to\_be\_executed\_to\_the\_whole\_of\_the\_list,list)

Print(var)

Note: If you are directly trying to print the map then the var will be in map class form so you need to type cast it into list. If you want to use a function in the map then you only need to specify the function name you don’t need to call the function.

**Filter:** Filter function is used to filter the elements in the list, filter makes a list on which a function returns true.

Syntax:

Var = list(filter(function\_name,list))

Decorators

**Decorators:** Decorators in Python help to modify the functionality of the function.

Syntax:

@dec1

More in the python file.

Object oriented programming

**Object oriented programming (OOPS ):** Object Oriented programming is a method of writing code to organize the code. Object oriented programming contains some classes which are literally equivalent to templates in the real world which are used to derive or make objects. Objects are also called as instances of that class.

Object oriented programming is done in the concept of DRY (do not repeat yourself) It means no code is re written again and again.

Class contains some code which helps the programmer to help or to make his work easy.

We can also make some instance variables.

**Pass:** Pass in Python refers to nothing in Python. If you want to skip a class in Python ( that means if you don’t want to write anything in the class) then you can write Pass to pass the code to the next snippet.

**To make a class and object of that class:**

class class\_name:

code in the class

pass if you don’t want to write anything in that class.

**To make some instance variables:**

object\_name.variable\_name = value

print(object\_name.variable\_name)

Self: Self is an argument taken in class methods to take instance name (self is taken by default). It is just like an ordinary argument but python prefers to use self as an argument when dealing with oops.

**To make a template in the class:** In the class you can make any code which can help while doing the object-oriented programming but the first argument you take in the method in the class should be self.

**\_\_init\_\_ ()**: init in Python classes is nothing but to initialize instant variables of the objects in the classes.

Note: while using this method we need to use self as the first argument in init() but not necessay explicitly as it will only throw warnings but the code will run as usual.

Class methods: class methods are methods which we make in class to alter/access class variables with the help of instance or even class itself.

Syntax:

@classmethod

def somefunction(cls,argument):

code in the function to alter class variables

Note: classmethod is a decorator which we use to change normal function to class method and this method will work if and only if it is places after init()

**Static Method**: In oops if you want to create any function/method which does not refer to object or class then we can use static method to make functions.

This is a decorator and this is used when we want to make simple function to save some time.

Syntax:

@staticmethod

def somefunction():

code in the function

**Inheritance:** Inheritance in python oops is similar to the general science inheritance as it is used to inherit methods written in parent class or also called as base class to child class. In this way child class can have all the methods of parent class along with its own methods without writing all the methods again in child class.

**Multiple Inheritance:** Multiple Inheritance in python is used to inherit two or more class methods into one.

Note: multiple inheritance depends on order of inheritance the first inherited class will be given more precedence (this happens if there is a similar function in both the class so the first inherited class method will be revoked)

Note: If you are accessing an instance variable from class which is inherited from another class then python will look for instance variable in local class then python will go to parent class if there is no instance variable then python will look for class variable in local class if not available then it will go to parent class.

**Multilevel Inheritance:** Multilevel inheritance is nothing but making a class which is inherited from a class which is then inherited from another class and the sequence goes on. It is similar to a family tree.

Note: In this we will be having some overriding in this session.

The least class will have the methods all from the parent class and its parent class.

If you are accessing a method from the least class then it will check for the method in the least class if not present then it will go to the upper class if not still then again it will go to the upper class at last if it finds there then it will be printed else error will be thrown.

In this way overwriting will also be done the method in the least class will be given most importance and will be printed even if the upper class also has the same method.

**Abstraction and Encapsulation:** Abstraction in Python is hiding internal details of a method in class and showing only functionalities Abstracting something means to give names to things, so that the name captures the basic idea of what a function or a whole program does.

Encapsulation: Encapsulation in python is the way to abstract a method in other words taking a method and encapsulating it (packing it such that the functionality of it is still intact but internal details are not shown).

**Public, Protected, Private variables:** In object-oriented programming we can show the variables publicly (to every one namely class) protected means only for that class and that inherited classes. Private means only for that particular class.

public Syntax:

class something:

var

protected Syntax:

class something:

\_var

Private Syntax:

class something:

\_\_var

Note: for accessing private variable you need to do like this

print(instance.\_\_class(\_var))

**Polymorphism:** Polymorphism in python oops is the ability to take multiple forms. This is done by in class, using dunder methods, overriding attributes in classes

**overriding:** overriding in python oops is nothing but if we need to have method in child class which is already available in parent class but with some changes then we can override the method in child class but using super() function.

Overriding problem comes when you have to use the method which is in the parent class and you have already overridden method in child class to deal with this problem we have super function.

Syntax:

Super().method\_in\_parent\_class() this imports variables from parent to child class even though they are overridden

Note: this works if the super function is at last of the method.

**Dunder Methods:** Dunder methods are the methods which start and end with \_\_ these methods will be executed while we are creating objects. These are constructors.

**Operator overloading:** Operator overloading in Python is the ability of a single operator to perform more than one operation based on the class (type) of operands.

For example, the + operator can be used to add to add numbers, concatenate two strings and, merge two lists.

Dunder methods are used for operating overloading for objects as by default there is no proper way to overload operators.

\_\_add\_\_(self,other):

Return self + other both are strings by the way

This dunder method is used to overload + operator

**Abstract Base class:** Abstract Base class is a class in which we cannot make any instances and it is definced in ABC module.

This along with abstractmethod which is also in ABC module is used to set up restrictions for an ordinary class to have some methods guranteed else that class will not be considered as a class.

**Property and setters**: Property decorators are used in oops to make methods, function llike attributes.

After using property decorator we don’t need to run that method as a function we can use it like an instance variable.

**Setter:** Setter is also a decorator in python which is used to set an instance variable ( re write an instance variable ) with the help of another variable.

**Deleter:** Deleter is a decorator in python which is used to delete an instance oops

Python matplotlib:

Matplotlib is a library for mapping 2d graphs and stuff.

Matplotlib has some specific functions which helps us to visualize data using bar graphs, pie charts, histograms and other such stuff

It has its own api interface for viewing graph which we made in the code.

Using it we can plot graphs of different colors, different sizes, different types and so on