Programming with Python

**5/14/2020**

Noman ul Haq

Learn python with practical example

Table of Contents

[Programming with Python 1](#_Toc40367008)

[Difference between hardware and software 1](#_Toc40367009)

[Software 1](#_Toc40367010)

[Compiler and interpreter 1](#_Toc40367011)

[Print Function 2](#_Toc40367012)

[Variables 3](#_Toc40367013)

[Variable for strings 3](#_Toc40367014)

[If Else Condition 3](#_Toc40367015)

# Programming with Python

"Guido van Rossum" is the founder of Python Language and it was in 1989. The name "Python" was selected after the television show Monty Python's Flying Circus.

We will learn python 3.7 (latest version of python language)

## Difference between hardware and software

All the tangible component of computer system is called hardware

* Keyboard, mouse, monitor, hard disk, printer, RAM

All the intangible component of computer system is called software

* Operating system, MS Office, web browser

### Software

* A software is a program or set of programs written using different programing languages
* Software is responsible for running hardware
* All operating systems are also software
* Operating system controls and runs hardware

# Compiler and interpreter

|  |  |
| --- | --- |
| **Interpreter** | **Compiler** |
| Translate program one statement at a time | Scans the all program and translate it as a whole in machine code |
| It takes less amount of time in analyze the source code but the overall execution time is slower | It takes large amount of time in analyze the source code but the overall execution time is comparatively faster |
| No intermediate object code is generated, hence are memory efficient | Generate intermediate object code which further require linking, hence require more memory |
| Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy | It generates an error message only after scanning the whole program. Hence debugging is comparatively hard |
| Programming language like python, ruby use interpreter | Programming language like C, C++, Java uses compiler |

## Print Function

|  |  |
| --- | --- |
| **Integer/ Floor division**  5/2 = 2.5  (in normal division it gives us answer with decimals ) | By default python gives floating point division. For getting integer division we should user “//”  5//2 = 2  In case we want to eliminate decimals we can do it through “//” |
| 2 \* 2 \* 2 = 8 | 2^3 = 2\*\*2 or 2^6 = 2\*\*6 or (2\*2\*2\*2\*2\*2) |
| To get **division** result we use | 10/3= 3.3333 or 10//3 = 3.0 |
| To get **reminder** we use **modular** **%** **operator** | 10%3 = 1 |
| **Functions** |  |
| Print = print is built in function which return our result value we are passing through the function | * Print should be print (all small letters) * print(‘Noman’) * Print(‘Noman’s Laptop) it will give an error * Print(“Noman’s Laptop”) for overcome previous error we should use double code instead * Print(‘Noman “Laptop”’) it will print double quote with the result * Print(‘Noman’s “Laptop”’) it will give us error * Print(‘Noman\’s “Laptop”’) when we want to bypass any command we have to put “\” slash (skipped special meaning of single quote) * Raw string   + Print(‘c:\user\document’) it will give result in with breaking line because here “\” means next line   + Print(r‘c:\user\document’) in this case we use “r” which is **Raw string** |
| Integer data | The values which are printing number is called integer values  print(12, 1000, 256000) |
| Floating | The number with decimals is called float  print(12.5, 12.3, 153.589) |
| String values | * The value which is enter as text * print("Noman") print(“12”) * in string + operator is not an addition but it is called as concatination |
| Alpha numeric data | To print alpha numeric data |
| Function signature (shift + tab) | To check signature of the command (description of command ) |
| Sep=’ ‘ (for changing default value of separator) | print("Noman","Pakistan",'Karachi')  Noman Pakistan Karachi  print("Noman","Pakistan",'Karachi',sep="/")  Noman/Pakistan/Karachi |
| end='\n' (by default print will use next line for printing second command  It is called python escape character | print("Noman","Pakistan",'Karachi',sep="/")  print("Kamran","Pakistan",'Karachi',sep="/")  print("Arsalan","Pakistan",'Karachi',sep="/")  Noman/Pakistan/Karachi  Kamran/Pakistan/Karachi  Arsalan/Pakistan/Karachi  print("Noman","Pakistan",'Karachi',end="/ ")  print("Kamran","Pakistan",'Karachi',end="/ ")  print("Arsalan","Pakistan",'Karachi',end="/ ")  Noman Pakistan Karachi/ Kamran Pakistan Karachi/ Arsalan Pakistan Karachi/ |

## Variables

Variable is a container where you put the values and the name of container is called variable. Why we called variable because we can change the value in the container. We don’t need to define type of variable as soon as we assign value it automatically get type accordingly.

|  |  |
| --- | --- |
| X =10  Y = 20  X+Y | Assign value in x variable |
| \_ (underscore represents output of previous operations) | \_+ Y |
| Name = youtube | Name + ‘rocks’ |
| Name[1] | Index start from 0123456 (from left to right ) |
| Name[-1]  Name[-2] | Index start fro -1,-2,-3,-4, (from right to left) |
| Name [0:2]  Name [1:4]  Name[1:]  Name[:4] | ‘yo’  ‘out’  ‘outube’  ‘yout’ |
| Name[0]= ‘R’ | String in python is **immutable** We cannot change  Value of it. |
| ‘My’ + name[3:] | You can concatenate value with variable |

## Variable for strings

## If Else Condition

a= 10\*200  
b= 25\*5  
if a<b:  
 print("a is less then b")  
 print("b is greater then a")  
else:  
 print("a is greater")  
#print("end")

# Functions

|  |  |
| --- | --- |
| Python | JavaScript |
| def add(): # parameter less Function  val1 = float(input("Please enter Value 1 "))  val2 = float(input("Please enter Value 2 "))  print(val1+val2)  add() | function message () {      val1 = Number(prompt("Please enter Value 1"));      val2 = Number(prompt("Please enter Value 2"));      console.log(val1+val2);  };  message(); |
| def fullName(first,middle,last):  print(first,middle,last)  **# Positional arguments**  fullName("Noman","ul","Haq")  **# Key-Word Argument**  fullName(first="Noman",last="Haq",middle="ul") | function fullName(first,middle,last){      console.log(first,middle,last)  }  fullName(first="Noman","Ul","Haq")  JavaScript does not have native support for named parameters like Python and many other languages. |
| Default values parameter in Function  All default parameters should place at last  def fullName(first,last,middle=" "):  print(first,middle,last)  fullName("Noman",middle="ul",last="Haq")  Noman ul Haq  fullName("Noman",last="Haq")  Noman Haq  fullName("Noman","Haq")  Noman Haq |  |
| **Dealing with unknown number of arguments**  # Deals with Arbitrary arguments  # Arbitrary arguments should be added at the last with \* sign  def pizzaOrder(size, flavor,\*toppings):  print(f"Your pizza is ready with the size of {size} the flavor of pizza is {flavor} and the toppings are {toppings}")  pizzaOrder(21,'Chiken','Cheese','Mustard','Extra Chiken') |  |
| **Passing information back from function**  **Function will ignore all the lines after the return as soon as return execute the function will gets finished**  def add(val1,val2):  result = val1\*val2  return result  add(5,8)  var = add(5,8)  print(var+25)  def add(val1,val2):  result = val1\*val2  return result, "This is passing values function"  var = add(5,8)  var  var = add(5,8)  print(var[0]\*5) |  |
| **Using functions as variables**  def mul(val1,val2):  result = val1\*val2  return result  def add(val1,val2):  result = val1+val2  return result  finalResult = mul(5,5) + add(8,8)  finalResult |  |
| **Functions Local and Global Variables in Python**  def Happy():  happyMan = "Mr. Abdul Rahman" # **Local Varialble**  print(f"{happyMan} is very happy today")  Happy()  SadMan = "Mr. Noman" # **Global variable**  def sad():  print(f"{SadMan} is sad today")  sad()  SadMan | **Functions Local and Global Variables in JavaScript**  Local variable is define with the key words of **var,** if we define without **var** it will be automatically converted to global variable |
| **Function within a function**  def commission(sales):  if sales > 1000:  return sales\*.05;  elif sales > 2000:  return sales\*.10;  else:  return sales\*0;  def grossSalary(salary,sales):  grossSalary = salary + commission(sales)  print(f"Gross salary is {grossSalary}")  grossSalary(50000,2500) |  |
| While Loop  x = 0  while x<=10:  print (f"the value of x is {x}")  x += 1  x = 3  while x>=1:  userinput = input("Please enter your favorite food ")  print (f"the value of x is {x} {userinput}")  x -= 1  flag = True  favFood =[]  a = 0  while flag:  userFavfood = input("Enter your favorite food")  a += 1  if userFavfood == "N":  flag = False  else:  print(f"user Number {a} choice food is {userFavfood}")  favFood.append(userFavfood)  favFood |  |
|  | function scores(sub1,sub2, sub3){      return (sub1 + sub2 + sub3 )/ 3  }  function grading (score){      if (score > 80 && score <= 100 ){          console.log("A1 Grade");      }      else if (score >70 && score <=80){          console.log("A Grade");      }      else if(score >60 && score <=70){          console.log("B Grade")      }      else if(score >50 && score <=60){          console.log("C Grade")      }      else {      console.log("Failed")}  }  a = scores(45,65,47);  console.log(a);  grading(a);  a = scores(77,90,80);  console.log(a);  grading(a);  a = scores(68,85,12);  console.log(a);  grading(a); |
|  | **Function Hoisting** |
| **Classes**  **Define a class**  class Car():  # Attributes # variables in programming  def \_\_init\_\_(self, make,model,year,speed):  self.make = make  self.model = model  self.year = year  self.speed = speed  # Bahviours # Functions in programming  def cardescription (self):  print(f"The car make is {self.make}")  print(f"The car model is {self.model}")  print(f"The car make is {self.year}")  def carmove(self):  print(f"the {self.make} {self.model} car move with the speed of {self.speed}")  def carbreak(self):  print(f"the {self.make} car break pad model is {self.year}") how to create objects landrover = Car("Land Rover","LR 2",2008,220)  toyota = Car("Toyota Corrola","Camri",2021,260)  landrover.carbreak()  toyota.carbreak()  landrover.cardescription()  toyota.cardescription() Changing an attribute value landrover.make = "Range Rover"  **direct hit changing attribute (not recommended)**  class Car():  # Attributes # variables in programming  def \_\_init\_\_(self, make,model,year,speed):  self.make = make  self.model = model  self.year = year  self.speed = 220 # setting default values  # Bahviours # Functions in programming  def cardescription (self):  print(f"The car make is {self.make}")  print(f"The car model is {self.model}")  print(f"The car make is {self.year}")  def carmove(self):  print(f"the {self.make} {self.model} car move with the speed of {self.speed}")  def carbreak(self):  print(f"the {self.make} car break pad model is {self.year}")  landrover.speed  220  landrover.speed = 260  260  toyota.speed  260  **changing attribute via function set and get (recommended)**  class Car():  # Attributes # variables in programming  def \_\_init\_\_(self, make,model,year,speed):  self.make = make  self.model = model  self.year = year  self.speed = speed  # Bahviours # Functions in programming  def cardescription (self):  print(f"The car make is {self.make}")  print(f"The car model is {self.model}")  print(f"The car make is {self.year}")  def carmove(self):  print(f"the {self.make} {self.model} car move with the speed of {self.speed}")  def carbreak(self):  print(f"the {self.make} car break pad model is {self.year}")  def setCarspeed(self,newSpeed):  self.speed = newSpeed  def getCarspeed(self):  print(f"The car speed {self.speed}")  landrover = Car("Land Rover","LR 2",2008,220)  toyota = Car("Toyota Corrola","Camri",2021,260)  landrover.speed  toyota.speed = 240  toyota.speed |  |
| **Data Files**  # Writing to a text file  # With open ("Filename","w") as file: # if file does not exist w mode will create anf write file  # file.write("This is my file")  # With open ("Filename","r") as file: # r mode will read content from file  # content = file.read()  # print(content)  # With open ("Filename","a") as file: # a mode will append in file # with w will overwrite all content # if we need to edit file the we user append mode # append mode will not create file  # Reading Mode  with open("test.txt","r") as file:  content = file.read()  print(content)  # Writing Mode  with open("test.txt","w") as file:  content = file.write("We are writng this text from python command")  with open("test.txt","r") as file:  content = file.read()  print(content)  with open("test1.txt","w") as f: # if file does not exist w will create and write new file  f.write("New file created from python")  with open("test1.txt","r") as file:  content = file.read()  print(content)  # Append Mode  with open("test1.txt","a") as f:  f.write(" New data edit with append mode")  with open("test1.txt","r") as file:  content = file.read()  print(content)  **# r+ Mode and w+ mode**  # with w+ mode we can write and read the files  with open("test2.txt","w+") as file:  file.write("We are writng this text from python command")  file.seek(0)  content = file.read() # if we directly start reading the file it will not show anything because it start reading from end so we need to set pointer with seek function at zero index  print(content)  # r+ mode also read and write the file  # the only different in r+, it will not create any new file it will read and write in existing file only  with open("test2.txt","r+") as file:  file.write("We are writng this by r+ command from python command")  file.write("We are writng this by r+ command from python command")  file.write("We are writng this by r+ command from python command")  file.write("We are writng this by r+ command from python command")  file.seek(0)  content = file.read() # if we directly start reading the file it will not show anything because it start reading from end so we need to set pointer with seek function at zero index  print(content) |  |
| **# Modules**  File create with name of Maths.py  def add(a,b):  return a + b  def mul(a,b):  return a\*b  def sub(a,b):  return a-b  def div(a,b):  return a/b  extracting any file as below:  %load maths.py  import Maths as m  m.add(2,8)  m.div(2,6)  m.mul(5,9)  m.sub(9,3) |  |
| **CSV Files**  **import csv**  **# Reading Data from a CSV file**  with open("csvtest.csv") as file:  content = csv.reader(file)  for cont in content:  contArr += cont  print(cont)  **# Writing Data into csv file**  with open("csvtest1.csv","w",newline="") as file:  fileWriter = csv.writer(file)  fileWriter.writerow(['Year', ' Event', ' Winner'])  fileWriter.writerow(['2020', ' Cricket World Cup', ' Australia'])  with open("csvtest1.csv") as file:  content = csv.reader(file)  for cont in content:  contArr += cont  print(cont)  **# Writing Data into csv file with append mode**  with open("csvtest1.csv","a",newline="") as file:  fileWriter = csv.writer(file)  fileWriter.writerow(['Year', ' Event', ' Winner'])  fileWriter.writerow(['1994', ' Hockey World Cup', ' Pakistan '])  fileWriter.writerow(['1994', ' Hockey World Cup', ' Pakistan '])  with open("csvtest1.csv") as file:  content = csv.reader(file)  for cont in content:  contArr += cont  print(cont) |  |
| JSON Format  Java Script Object Notation  # writing data into json file with List  import json  with open ("jsonfile.json","w") as file:  json.dump([12,13,14,15,16,17,18,19,20], file)  with open("jsonfile.json","r") as file:  content = json.load(file)  # Reading data from json file    with open("jsonfile.json","r") as file:  content = json.load(file)  # Append data into json file  import json  with open ("jsonfile.json","a") as file:  json.dump([21,22,23,24], file)  # writing data dictionary in json file  myDic = {"first name":"Noman", "middle Name":"Haq","last name":"Haq"}  with open ("jsondicfile.json","w") as file:  json.dump(myDic, file)  with open("jsondicfile.json","r") as file:  content = json.load(file) |  |
| **Exceptions**  Exceptions are handling with **try**, **except** and **finally** key words.  try:  fileName = input("Please enter file name")  with open(fileName,"r") as f:  content = f.read()  print(content)  except FileNotFoundError:  print("Sorry, " + fileName + " not found.")  while True:  try:  fileName = input("Please enter file name")  with open(fileName,"r") as f:  content = f.read()  print(content)  break    except FileNotFoundError:  print("Sorry, " + fileName + " not found.")  try:  fileName = input("Please enter file name")  with open(fileName,"r") as f:  content = f.read()  print(content)  except FileNotFoundError:  print("Sorry, " + fileName + " not found.")  finally:  print("Finally will always run")  while True:  try:  fileName = input("Please enter file name")  with open(fileName,"r") as f:  content = f.read()  print(content)  break    except FileExistsError:  print("Sorry, " + fileName + " not found.")  finally:  print("Finally will always run") |  |