Today's Topics

==================

->Overview of Python , and It's Applicaitons

->Python Environment

->Hello/First Program in python

->Data Type and variable

->Input/Output

->COndition

OVerview of Python , and It's Applicaitons

============================================

->python is open source (free) programing language

->python is platform independent (support to all operating -mac,win,linux etc.)

->python is light weight and easy to understand/use

->python is interpreter based language(line by line execution)

->python is indentation/allignment based language

::Core Python/Basic python

It's Applicaitons

=====================

->Data Sci.

-ML

-AI

-Automation

->Web Development : DJango

->Software testing : Selenium

->EH and Security

Python Environment

======================

Version:

3.7

3.6 \*\*

3.5

3.4

...

...

2.7 \*\*

Python Editor:

-Python Shell/Command Line / Console

>>>

-Python Editor: IDLE /GUI

Python code can be saved with .py extension

-Other:

-PyCharm

-Annoconda

-Spyder

-VSCode

etc.

Hello/First Program in python

===================================

a =11

b =3434

c =a+b

print(c)

print('sum of two numbers :',c)

Essentials:

-----

-Python is indentation bases language

# : single line comment

''' : multiple line comment

'''

Data Type and variable

=============================

int : a =111

float : a =3333.2

str : a ='sfff' , a ="sjhsjgfsh33"

bool : a =True

list : a =[111,2,2,34,'ffdd','fdfff']

tuple : a =(1,2,2,3,'444') # read only

dict : d ={'a':'alpha','b':'beta','c':'ceta',1:'one'}

set : a ={'dove','lux','dove'}

store only unique value , one dove will remove automatically

Day 1

===================

Functions

===================

input() : read data from console , default data type is str

print() : show data to user/ouput

int() : convert to int

str() : convert to str

type() : show data type /class type

Operator

===================

Arithmetic:

+

-

\* : 2\*3 = 6

\*\* : 2\*\*6 = 16

/ : 85/10 = 8.5

// : 85//10 = 8

% : 85%10 = 5

Conditional:

>

>=

<

<=

==

!=

Logical:

and

or

Keywords:

in

not in

Conditinal: is decision making statement

===========================================

-Python doesn't support switch case

-If condition types:

1. if condition

2. if else condition

3. if elif elif ..... else condition / ladder if else

4. nested if else / if inside if condition

Loop : is iterator or repeation of statement/command

===================

Example:

1 2 3 ...100

Fundamental of loop:

-init / start : 1

-condition/limit : 100

-increment/decrement : +1

There are following types of loop:

-while loop

-for loop : #for i in range(I,n)

Day 1

===================

Functions

===================

input() : read data from console , default data type is str

print() : show data to user/ouput

int() : convert to int

str() : convert to str

type() : show data type /class type

Operator

===================

Arithmetic:

+

-

\* : 2\*3 = 6

\*\* : 2\*\*6 = 16

/ : 85/10 = 8.5

// : 85//10 = 8

% : 85%10 = 5

Conditional:

>

>=

<

<=

==

!=

Logical:

and

or

Keywords:

in

not in

Conditinal: is decision making statement

===========================================

-Python doesn't support switch case

-If condition types:

1. if condition

2. if else condition

3. if elif elif ..... else condition / ladder if else

4. nested if else / if inside if condition

Loop : is iterator or repeation of statement/command

===================

Example:

1 2 3 ...100

Fundamental of loop:

-init / start : 1

-condition/limit : 100

-increment/decrement : +1

There are following types of loop:

-while loop

-for loop

Day 2

=====================

-break, continue

-nested loop

-list and it's operaitons

-GUI

break, continue

=======================

break: to terminate the iterations/loop when condition will match

continue : to skip the current iteration when condition will match

nested loop : loop inside loop /loop within loop

==============

list and it's operaitons

=================================

list : collection is collection of data or values

: multiple values can be stored on single variable

: a =[1,2,3,4,5,'fhsjs','444']

\*List index will start at 0

There are following inbuilt methods:

append() : add new elemement after last index

pop() : show and remove last element

insert() : add new element at given position

remove() : remove existing element

max() : return highest value

min() : return lowest value

sum() : return total of all elements

len() : return size of list (count of elements)

sort() : arrange data in acending order

slicer : return element by given index

etc.

GUI : graphical user interface

===================================

There is inbuilt module :

tkinter

There are folling inbuilt classes:

-Tk

-Label

-Button

-pack()

Day 3

=================

-String Functions

-File Handling

String Functions

=======================

There are following inbuilt functions/Mehtods:

upper() : convert to upper case

lower() : convert to lower case

title() : convert to proper case

this is = This Is

strip() : remove leading/extra space

replace() : find and replace

len() : return count of chars

list() : convert to list char by char

split() : break string by given sperator

slicer : read string/char by given index

etc.

File Handling : read and write operation with files (.txt,.dat)

====================

open(path,mode of file) : is inbuilt function

path: c://abcd/a.txt

mode of file:

r - read

w - write

a - append

w+ - read and write

a+ - read and append

read() : read all contents from file

readline() : read first line / line by line

readlines() : reas all lines/content from file and convert to list/array

write() : write content/string/data to file

close() : save and close the instance of file

##

q. wap to get count of gender(male, and female) from file

q. wap to show gender wise salary

male - 2 - 344344

female - 3 - 666

q. wap to beak/distribute one file by country name

india.txt

uk.txt

id,name,gender,salary,country

1,raman,male,4444,india

2,monika,female,422,uk

Function : is set of command and instructions which is reusable

-------

There are following types of functions:

-no argument , no return

-no argument , with return

-argument , with no return

-argument , with return

-function with default argument

-function with dynamic arguments

Module: is collection of functions, and classes which is reusable

---------------

import module

from module import funname,funname

import module as m

OOPS: object oriented programing structure/system

---------------------

Advantage :

-Reusability of source code

-Support to modular programing , large/complex task can be written in small set of instructions

-Easy to manage the source code

There are following fundamenal of oops:

-class : is wrapper of data member and function

-object : is an instance of class

Day 6

-----------------

Install driver in python

=================================

cd C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36\_64\Scripts

pip install mysql-connector

Database Details/Server Credentials

==============================

-> My SQL server :

-> Oracle

-> MS SQL Server

etc.

My SQL server :

ServerIp Address :

localhost or 127.0.0.1

User id :

root

Password :

root

Database :

show databases; -- show list of all existing databases

create database learning; --create learning db

use learning; -- switch database / go insiide db

show tables; --show list of existing tables

--create table

create table emp(eid int, name varchar(100));

--insert data

insert into emp(eid,name) values(1,'raman');

-view data

select \* from emp;

--where / search data

select \* from emp where eid=1;

select \* from emp where eid>1;

select \* from emp where name like 'a%';

select \* from emp where name like '%a';

select \* from emp where name like '%a%';

select \* from emp where email like '%@gmail.com';

select \* from emp where name like 'a%b\_\_\_@gmail.com';

% : any chars, any no. of times

\_\_: any chars given no. of times

Code:

======================

import mysql.connector#load /import library /module

#establish the connection between python to mysql server

c = mysql.connector.connect(host="localhost",user="root",password="root",database="learning")

#create object of cursor , which can execute/run/fire sql statement

cur = c.cursor()

while True:

op = input('press 1 for add new row, 2 for show 3 for delete 4 for exit ')

if op =='1':

eid = input('enter eid :')

name = input('enter name :')

# "+eid+"

#cur.execute("insert into emp(eid,name) values(1,'xyz')")

cur.execute("insert into emp(eid,name) values("+eid+",'"+name+"')")

c.commit()

elif op =='2':

#execute sql command/statement

cur.execute("select \* from emp")

#read data from cur object

out = cur.fetchall()

#iterate the list

for r in out:

print(r)

elif op =='3':

eid = input('enter eid to remove :')

cur.execute("delete from emp where eid ="+eid)

c.commit()

elif op =='4':

break

else:

print('invalid choice')

Day 7

===================

-> Update

-> Class - with loop

-> System Date

import time

time.localtime()

**TOpics**

===================

OOPS

---------

-class : is wrapper of data member and function

-object : is an instance of class

-encapsution : is wrapping of data memebr and function in one unit i.e encapsulation

: this is default concept

: every class is by default encapsulated

-abstraction : expose the essential features of class/method and hide the implementation i.e. abstracton

:example

print()

-constructor : is function which invoke/call/execute automatically when object will create

: def \_\_init\_\_() : is system defined/reserved function which is constructor

-> constructor cannot be invoke/call explicitly

-> constructor cannot have return type

-deconstructor :is function which invoke/call/execute automatically when object will be deleted

: def \_\_del\_\_() : is system defined/rserved function

-> deconstructor cannot be invoke/call explicitly

-> deconstructor cannot have return type

-inheritence : to extend one class features/functions to another class

(parent - child relationship)

: There are following types of inheritnece:

i. single level

A -> B

ii. multi level

A -> B -> C ...

iii. tree/hirarchical

A ->B , C, D -> M,N

iv. hybrid

A ->B -> C

M-> N,C

v. multiple

A,B,C -> D

-overriding

\*\*python doesn't support overloading

add(a,b):

add(a,b,c) :