Python Training

Day 1 : 26-Feb-2024

Software :

System software

Application software

BCPL

Cobol

Fortan 77

Pascal

C 1970 : structure programming language

C+ Oops = C++ or C with classes

1980

1990 : Java as well as Python

2000 : .net : C#

2010 : JavaScript UI cloud technologies

2020 : AI, Machine learning : Python as well as R language

Web Service : Giving the service for web application when two application running using different technologies.

Micro service

Pip

Mvn

Npm

int a=10;

a=10;

int a=100;

a=”Steven”;

a=100;

a=”Ravi”;

a=True;

REPL Terminal :

Variable

int a=10; C or C++ or Java

a=20;

a = “Akash”;

var a=10; in JavaScript

let a:number =100; in typescript

a=10; in python

name=”Akash”;

result = True

a=False

differentTypeOfVariable.py

>>> id=100;

>>> name="Ravi";

>>> result = True;

>>> list =[10,20,30];

>>> set ={10,20,30};

>>> tuple=(10,20,30);

>>> dic = {id:100,name:"Ravi",age:21};

>>>print(type(id));

Multi variable declaration

>>> id,name,salary,result=100,"Steven",34000,True;

>>> print(id,name,salary,result);

100 Steven 34000 True

Same value storing in more than one variable

>>> a=b=c=100;

>>> print(a,b,c);

100 100 100

id() pre defined function which is use to find the reference code of that variable.

Mutable and immutable example

>>> empid=500;

>>> print(id(empid));

2784931300144

>>> empid=600;

>>> print(id(empid));

2784931314736

>>> ll=[100,200,300];

>>> print(ll);

[100, 200, 300]

>>> print(id(ll));

2784928387648

>>> print(ll[0]);

100

>>> ll[0]=1000;

>>> print(ll[0]);

1000

>>> print(id(ll));

2784928387648

ArithmeticOperator.py

a=10;

b=3;

res1 = a+b;

res2 = a-b;

res3 = a\*b;

res4 = a/b;

res5 = a//b;

res6 = a\*\*b;

res7 = a%b;

print(res1);

print(res2);

print(res3);

print(res4);

print(res5);

print(res6);

print(res7);

a=1+a;

a+=1;

in and is operator example

name="Welcome to python training";

print("in operator example");

print(name);

print("o" in name);

print("W" in name);

print("w" in name);

print("python" in name);

print("is operator example");

a=500;

b=500;

c=500;

print("a ",a);

print("b",b);

print("c",c);

print(id(a));

print(id(b));

print(id(c));

print(id(500));

print(a is b);

print(a is c);

a=600;

print("a ",a);

print(id(a));

if statement example

#a=-5;

#if a > 0 :

# print("number is +ve");

# print("Condition true");

#print("Normal Statement execute");

'''

a=10;

b=50;

if a > b:

print("a is largest");

else:

print("b is largest");

print("Normal Statement execute");

'''

sub1=80;

sub2=90;

sub3=90;

total = sub1+sub2+sub3;

avg = total/3;

if avg > 90:

print("A+");

elif avg >80:

print("A");

elif avg>70:

print("B");

else:

print("C");

print("done!");

looping.py

'''

for n in range(1,10):

print(n);

'''

'''

for n in range(1,10,2):

print(n);

'''

'''

list =[100,200,300,400,500];

for n in list:

print(n);

'''

'''

i=1;

n=10;

while i <= n:

print(i);

i=i+1;

print("while loop finish");

'''

x=0;

while True:

print(x);

x=x+1;

if(x==5):

break;

print("loop finish");

Day 2: 28-Feb-2024

input(): it is a pre defined function which help to take the value through console.

name=”Ravi Kumar”;

Printf(“your name is”,name);

Taking the value through keyword in Python

name=input(“Enter your name”);

print(“your name is “,name);

a=10;

a=20;

a=30;

if we want to store more than one value of same type or different types then in python we can use sequence data types.

list

set

dict

tuple

Stack

Queue

LinkedList

Tree

Map

Day 3 : 01-Feb-2024

Small application with basic python programming

Banking application

Dictionary, List, while, input, int, float, function etc.

1. Create account
2. Check balance
3. Withdraw
4. Deposit
5. Check account details

Python OOPs

OOPs using python

Object Oriented Programming

Object : object is any real world entity.

Property or state --🡪 have

Person

Behaviour 🡪 do/does

Bank

Animal

Car

Customer

Employee

Class : blue print of object or template of object or collection of object which have same property and behaviour or user defined data type which help to create the object.

ES5 javascript didn’t support class concept to create the object.

Using class we can implements objects.

Wheel, colour, price -🡪 has -🡪 variable or fields

car

start(), applied\_gear(), moving(), stop()🡪 do the task

function or methods.

Constructor : constructor is a type of special function which help to create the memory.

Java : constructor have same name as class itself.

JavaScript : we need write function name with as constructor.

Python : python provided pre defined function with name as \_\_init\_\_(self): which behave like a constructor.

Encapsulation : binding or wrapping data (variables ) and function (code) in a single unit is known as encapsulation.

Example : Class itself by default follow Encapsulation rules.

If we want to make the instance variable as private we need to start with \_\_ prefix as variable then it consider as private.

Inheritance : Inheritance is use to inherits the properties and behaviour of old class to new class. mainly it is use to do re-usability.

Once we do inheritance then with help of sub class object we can access super class property and behaviour. But with help of super class object we can access only its own property and behaviour.

Employee -🡪super class eid,name,salary

Manager 🡪 sub class employee numberofemp

Developer 🡪 sub class of employee projectname

ProjectManager 🡪 sub class Manager 🡪 clientInfo

Employee --- Manager 🡪 Single

Employee – Manager – ProjectManager – Multilevel

Employee – Manager and Developer – Hierarchical

Polymorphism : One name many forms or many implementation.

2 types

Compile time polymorphism : function overloading or operator overloading

Run time polymorphism : function overriding

Function overloading : the function have same name but different parameter list is known as function overloading.

Function overriding : the function have same name and a same function signature : means number of parameter and type of parameter must be same but different implementation.

Decorator or annotation

All decorator or annotation start with prefix @ followed by name. which provide extra behaviour to function or property.

Day4

Pre defined modules and user defined modules

Connection database mysql or postgres sql using python

We develop small application

Custom decorator or annotation

Day 5 and 6

Generic typing

Last two days

Python with Django with database we develop small application