PYTHON PRACTICALS

#Basic data types, operators, expressions and input output statement

Practical No. 1

```
name=input("Enter your name:")
age=int(input("Enter your age:"))
cgpa=float(input("Enter your cgpa:"))
is_Passed=bool(input("is_Passed? True or False:"))
List=[45,"Python",78,99,"Sem 3"]
Dict={"name": "Vaishnavi","age":19,"sem":"III","roll_no":420}
Tuple=(12,43,54,58,94,54)
Set={12,23,45,66,33}
print(name,type(name))
print(age,type(age))
print(cgpa,type(cgpa))
print(is_Passed,type(is_Passed))
print(List,type(List))
print(Dict,type(Dict))
print(Tuple,type(Tuple))
print(Set,type(Set))
a=int(input("Enter a:"))
b=int(input("Enter b:"))
c=int(input("Enter c:"))
Sum=a+b
Subtraction=a-b
Multiplication=a*b
Division=a/b
Floor_Division=a//b
Modulus=a%b
Exponent=a**c
print(a,"+",b,"=",Sum)
print(a,"-",b,"=",Subtraction)
print(a,"*",b,"=",Multiplication)
print(a,"/",b,"=",Division)
print(a,"//",b,"=",Floor_Division)
print(a,"%",b,"=",Modulus)
print(a,"^",c,"=",Exponent)
```

Output:

```
Enter your name:Vaishnavi Kale
Enter your age:19
Enter your cgpa:8.5
is_Passed? True or False:True
Vaishnavi Kale <class 'str'>
19 <class 'int'>
8.5 <class 'float'>
True <class 'bool'>
[45, 'Python', 78, 99, 'Sem 3'] <class 'list'>
{'name': 'Vaishnavi', 'age': 19, 'sem': 'III', 'roll_no': 420} <class 'dict'>
```

```
(12, 43, 54, 58, 94, 54) <class 'tuple'>
{33, 66, 23, 12, 45} <class 'set'>
Enter a:20
Enter b:24
Enter c:2
20 + 24 = 44
20 - 24 = -4
20 * 24 = 480
20 / 24 = 0.8333333333333334
20 // 24 = 0
20 % 24 = 20
20 ^ 2 = 400
```

—--

# Control flow statement (if, if…else, nested if)

Practical No 2

#1. Write a program to print you are adult if age is greater than equal to 18 using if statement.

```
age=int(input("Enter age : "))
if age>=18:
print("You are Adult.")
print(" ")
```

#2. Write a program to print whether given number is positive or negative using if..else statement.

```
a=int(input("Enter a number (a) : "))
b=int(input("Enter a number (b): "))
if a>0:
print(a,"is Positive.")
elif a<0:
print(a,"is Negative.")
if b>0:
print(b,"is Positive.")
elif b<0:
print(b,"is Negative.")
print(" ")
```

#3. Write a program to print whether student is passed or not based on marks using nested it statement.

```
marks=int(input("Enter total marks :"))
if marks>40:
if marks>75:
 print("Student is passed with distinction.")
elif marks<75:
 print("Student is passed.")
else:
 print("Student is failed.")
```

Output:

```
Enter age : 19
You are Adult.
```

```
Enter a number (a) : 200
Enter a number (b): -300
200 is Positive.
-300 is Negative.
Enter total marks :85
Student is passed with distinction.
```

—---

#Looping in python while loop, for loop.

Practical No. 3

1. Write a program to print prime number from 1 to 50 using for loop.

```
for x in range(2,60):
 for i in range(2,x):
 if x%i==0:
 break
 else:
 print(x,end=",")
print(" ")
```

Output:

```
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59
```

2. Write a program to print multiplication table of given number using while loop.

```
a=int(input("Enter a:"))
i=1
while a>0 and i<11:
 print(a,"x",i,"=",a*i)
 i=i+1
```

Output:

```
Enter a: 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

—---

# 1)different List and tuple operation using built in function 2)built in set and string function

Practical No. 4

---->1.LIST OPERATIONS

```
L1=[10,20,"Janhavi",30,40,"Python",50] #list creation
print("L1 is :",L1,type(L1))
print("Length of a L1 is :",len(L1)) #to find length of the list
print("Element at index 3 is:",L1[3]) #to access element in the list
```

```
L1[3]='List' #to replace element using index
print(L1)
L1.append(60) #to insert item at the end of the list
print(L1)
L1.insert(1,"Lab") #to insert item in list
print(L1)
L2=[100,70,90,80]
print(L2)
print("List 2 is:",L2)
print("Sum of elements in the L2:",sum(L2)) #to find sum of all elements in list
print("Maximum element in L2 is:",max(L2)) #to find maximum element in list
print("Minimum element in L2 is:",min(L2)) #to find minimum element in list
print("Sorted list is:",sorted(L2)) #to sort list elements
L1.extend(L2) #to extend list using another list
print("Extended L1 is:",L1)
L1.remove("Lab") #to remove element from the list
print(L1)
L1.clear() #to clear list
print("After clear operation L1 is:",L1)
print(" ")
```

---> 2.Tuple Operations

```
T1=(45,78,96,74,35,54,45,78)
print("Tuple T1 is:",T1,type(T1))
print("Length of a T1 is :",len(T1)) #to find length of the tuple
print("Maximum element in T1 is: ",max(T1))
print("Minimum element in T1 is: ",min(T1))
print("Sorted tuple T1 is:",sorted(T1))
print("Sum of all elements in tuple is:",sum(T1))
print("45 is repeated for :",T1.count(45),"times") #to count occurrence of element
print("Index of 78 is:",T1.index(78)) #returns the index of the 1st occurrence
print(" ")
```

---->3.Set Operations

```
print(" ")
S1={12,34,56,76,89,98} print("S1 is:",S1,type(S1))
print("Length of set S1 is:",len(S1))
S1.add(45) #adds element in the set
print("After adding an element in S1:",S1)
S1.remove(34) #removes element from the set
print("After removing an element in S1:",S1)
S1.pop() #removes 1st element from the set
print("After pop() in S1:",S1)
S2={44,89,15,74,18}
print("S2 is:",S2)
S1.union(S2)
print("S1 union S2 is:",S1)
S1.intersection(S2)
```

```
print("S1 intersection S2 is:",S1)
S1.difference(S2)
print("S1 difference S2 is:",S1)
print(" ")
```

---->4.String Operations

```
Str1=" Janhavi"
Str2="Koli"
print("Str1 is:",Str1,type(Str1))
print("Str2 is:",Str2,type(Str2))
print("Length of the string is:",len(Str1))
print(Str1.upper()) #converts lower to uppercase
print(Str2.lower()) #converts upper to lowercase
Str3=" Hello World!! "
print("Str3 is:",Str3,type(Str3))
print(Str3.strip()) #removes leading and trailing white spaces
print(Str1 + " "+ Str2)
print(Str3.replace("World","Guys"))
print(Str3.find("Hello"))
print(Str3.split())
print(",".join(Str1))
```

Output:

--->1.

```
L1 is : [10, 20, ' Janhavi', 30, 40, 'Python', 50] <class 'list'>
Length of a L1 is : 7
Element at index 3 is: 30
[10, 20, ' Janhavi', 'List', 40, 'Python', 50]
[10, 20, ' Janhavi’, 'List', 40, 'Python', 50, 60]
[10, 'Lab', 20, ' Janhavi ', 'List', 40, 'Python', 50, 60]
[100, 70, 90, 80]
List 2 is: [100, 70, 90, 80]
Sum of elements in the L2: 340
Maximum element in L2 is: 100
Minimum element in L2 is: 70
Sorted list is: [70, 80, 90, 100]
Extended L1 is: [10, 'Lab', 20, ' Janhavi', 'List', 40, 'Python', 50, 60, 100, 70, 90, 80]
[10, 20, ' Janhavi', 'List', 40, 'Python', 50, 60, 100, 70, 90, 80]
After clear operation L1 is: []
```

--->2.

```
Tuple T1 is: (45, 78, 96, 74, 35, 54, 45, 78) <class 'tuple'>
Length of a T1 is : 8
Maximum element in T1 is: 96
Minimum element in T1 is: 35
Sorted tuple T1 is: [35, 45, 45, 54, 74, 78, 78, 96]
Sum of all elements in tuple is: 505
45 is repeated for : 2 times
```

```
Index of 78 is: 1

--->3.
S1 is: {34, 98, 56, 89, 12, 76} <class 'set'>
Length of set S1 is: 6
After adding an element in S1: {34, 98, 56, 89, 12, 45, 76}
After removing an element in S1: {98, 56, 89, 12, 45, 76}
After pop() in S1: {56, 89, 12, 45, 76}
S2 is: {18, 89, 74, 44, 15}
S1 union S2 is: {56, 89, 12, 45, 76}
S1 intersection S2 is: {56, 89, 12, 45, 76}
S1 difference S2 is: {56, 89, 12, 45, 76}

--->4.
Str1 is: Janhavi<class 'str'>
Str2 is: Koli<class 'str'>
Length of the string is: 8
JANHAVI
koli
Str3 is: Hello World!! <class 'str'>
Hello World!!
Janhavi Koli
Hello Guys!!
1
['Hello', 'World!!']
J,a,n,h,a,v,i
—-------------------------------------------------
```

```
# Basic array operation on 1-D and multidimensional arrays using numpy
Practical No. 5
#1-D Array
import array as arr
a=arr.array('i',[4,5,6])
print(type(a))
for i in a:
 print(i,end=" ")
#2-D Array
import numpy as np
list1=[[11,12,13],[14,15,16],[17,18,19]]
a1=np.array(list1)
print(a1.shape)
print(a1.size)
print(a1.dtype)
print(a1.ndim)
print(a1.itemsize)
print(a1[1,2])
print(a1[0,:])
print(a1[:,1])
print(a1[1,])
print(a1[1:3,1:3])
print(a1[1:3,])
print(a1[:,1:3])
print(a1[1:3,1])
print(a1[1:3,:1])
```

Output:

```
<class 'array.array'>
4 5 6 (3, 3)
9
int32
2
4
16
[11 12 13]
[12 15 18]
[14 15 16]
[[15 16]
[18 19]]
[[14 15 16]
[17 18 19]]
[[12 13]
[15 16]
[18 19]]
[15 18]
[[14]
[17]]
```

—---

# Anonymous function (lambda, map , reduce , filter)

Practical No 6

Program :

```
#To study anonymous functions in python
L1=[10,11,12,13,14,15,16,17,18,19,20]
get=lambda L1:list(filter(lambda x:x%2==0,L1)) #filter() filters out the element
e=get(L1)
print("Even numbers using filter():",e)
L1=[10,11,12,13,14,15,16,17,18,19,20]
get=lambda L1:list(map(lambda x:x%2==0,L1)) #map() filters out the element
e=get(L1)
print("Even numbers using map():",e)
L1=[1,2,3,4,5,6,7,8]
sq=list(map(lambda x:x**2,L1))
print("square numbers are:",sq)
from functools import reduce
L1=[1,2,3,4,5]
r=reduce(lambda x,y:x*y,L1) #reduce() function reduces an iterable to a single value
print("Product of all numbers is:",r)
```

```
Output:
Even numbers using filter(): [10, 12, 14, 16, 18, 20]
Even numbers using map(): [True, False, True, False, True, False, True, False, True,
False, True]
square numbers are: [1, 4, 9, 16, 25, 36, 49, 64]
Product of all numbers is: 120
```
—---
```
# Display employee information
Practical No 7
class EmployeeInfo:
 def __init__(self,eid,name,address,contact):
 self.eid=eid
 self.name=name
 self.address=address
 self.contact=contact
 def show(self):
 print("Employee details are: ",self.eid,self.name,self.address,self.contact)
obj1=EmployeeInfo("E101 :","Vrushali ,","Sion ,","7458963211")
obj2=EmployeeInfo("E102 :","Kumud ,","Powai ,","9874563214")
obj3=EmployeeInfo("E103 :","Ankita ,","Jogeshwari ,","8527419630")
obj4=EmployeeInfo("E104 :","Shravani ,","Goregaon ,","8745696321")
obj5=EmployeeInfo("E105 :","Vaishnavi ,","Dadar ,","8965741236")
obj1.show()
obj2.show()
obj3.show()
obj4.show()
obj5.show()

Output:
Employee details are: E101 : Vrushali , Sion , 7458963211
Employee details are: E102 : Kumud , Powai , 9874563214
Employee details are: E103 : Ankita , Jogeshwari , 8527419630
Employee details are: E104 : Shravani , Goregaon , 8745696321
Employee details are: E105 : Vaishnavi , Dadar , 8965741236
```
—---
```
#Study inheritance
Practical No. 8
#single inheritance
print("Single Inheritance :")
class parent:
 def show(self):
 print("I am in parent class.")
class child(parent):
 def display(self):
 print("I am in child class.")
obj=child()
obj.show()
obj.display()
```

```
#Multiple inheritance
print("Multiple Inheritance :")
class father:
 def show_father(self):
 print("This is class father.")
class mother:
 def show_mother(self):
 print("This is class mother.")
class child(father,mother):
 def show_child(self):
 print("This is child class.")
ob=child()
ob.show_father()
ob.show_mother()
ob.show_child()
#multilevel inheritance
print("Multilevel Inheritance :")
class vehicle:
 def show_vehicle(self):
 print("This is class vehicle.")
class car(vehicle):
 def show_car(self):
 print("This is class car.")

class bus(car):
 def show_bus(self):
 print("This is class bus.")
ob=car()
ob2=bus()
ob.show_vehicle()
ob2.show_car()
ob2.show_bus()
#hierarchical inheritance
print("Hierarchical Inheritance :")
class mathematics:
 def show_maths(self):
 print("This is base class mathematics.")
class geometry(mathematics):
 def show_geo(self):
 print("This is derived class geometry.")

class algebra(mathematics):
 def show_algebra(self):
 print("This is derived class algebra.")

ob=geometry()
ob2=algebra()
ob.show_geo()
ob.show_maths()
ob2.show_algebra()
ob2.show_maths()
```

Output:

```
Single Inheritance :
I am in parent class.
I am in child class.
Multiple Inheritance :
This is class father.
This is class mother.
This is child class.
Multilevel Inheritance :
This is class vehicle.
This is class car.
This is class bus.
Hierarchical Inheritance :
This is derived class geometry.
This is base class mathematics.
This is derived class algebra.
This is base class mathematics.
```

—---

```
# Method overloading, Method overriding
```

Practical No. 9

```
#1. Method Overloading
class MathOperations:
 def add(self, a, b=0, c=0):
 return a + b + c
obj = MathOperations()
print(obj.add(5))
print(obj.add(5, 10))
print(obj.add(5, 10, 15))
print("="*25)
#2.Method overriding occurs
class Parent:
 def show(self):
 print("This is the Parent class")
class Child(Parent):
 def show(self):
 print("This is the Child class")
obj1 = Parent()
obj1.show()
obj2 = Child()
obj2.show()
print("="*25)
#3. Abstract Class & Abstract Method
from abc import ABC, abstractmethod
class Vehicle(ABC):
 @abstractmethod
 def start(self):
 pass
class Car(Vehicle):
 def start(self):
 print("Car is starting...")
class Bike(Vehicle):
 def start(self):
 print("Bike is starting...")
car = Car()
car.start()
bike = Bike()
bike.start()
print("="*25)
#4. Interfaces in Python
from abc import ABC, abstractmethod
class Animal(ABC):
 @abstractmethod
 def make_sound(self):
 pass
class Dog(Animal):
 def make_sound(self):
 return "Bark"
class Cat(Animal):
 def make_sound(self):
 return "Meow"
dog = Dog()
cat = Cat()
print(dog.make_sound())
print(cat.make_sound())
```

Output:

```
5
15
30
=========================
This is the Parent class
This is the Child class
=========================
Car is starting...
Bike is starting...
=========================
Bark
Meow
```

—---

# User defined modules/packages and import them in program

Practical No. 10

#1st Program : module1.py

```
#module is python program containing python code
def multiply(a,b):
 return a*b
def display(name):
 return f"Hello,{name}"
#2nd Program: Modules.py
def concat(a,b):
 return a + b
def multiplicationTable(n):
 result=[]
 for x in range(1,6):
 result.append(n*x)
 print(n*x)
 return result
#Using created modules
import module1
from module1 import multiply
print(multiply(5,3))
print(module1.display("Vrushali"))
import Modules
from Modules import concat
print(concat("Hello"," World"))
print(Modules.multiplicationTable(2))
```

Output :

```
15
Hello,Vrushali
Hello World
2
4
6
8
10
[2, 4, 6, 8, 10]
```

—---

# User defined multithreaded application with thread synchronisation and thread asynchronization.

Practical No. 11

```
#thread synchronisation
import threading
import time
lock=threading.Lock() #creating the object for synchronization
def printTable(n):
 with lock:
 for i in range(1,11):
 print(n,"x",i,"=",n*i)
 time.sleep(0.5)
 print("- "*20)
```

```
#creating thread. target specifies the function we wished to print, args pass the value to
the function
t1=threading.Thread(target=printTable,args=(10,))
t2=threading.Thread(target=printTable,args=(13,))
t1.start()
t2.start()
t1.join()
t2.join()
print("The table printing has been completed.")
```

Output :

```
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
- - - - - - - - - - - - - - - - - - - -
13 x 1 = 13
13 x 2 = 26
13 x 3 = 39
13 x 4 = 52
13 x 5 = 65
13 x 6 = 78
13 x 7 = 91
13 x 8 = 104
13 x 9 = 117
13 x 10 = 130
- - - - - - - - - - - - - - - - - - - -
The table printing has been completed..
```

```
#thread asynchronization
#without..
import
threading
import time
def print_table(n, n2):
for i in range(n, n2):
print(i)
time.sleep(0.2) #
Simulating delay print("-" *
20)
# Creating threads without synchronization
t1 = threading.Thread(target=print_table,
args=(1,11)) t2 =
threading.Thread(target=print_table,
args=(11,21)) t3 =
threading.Thread(target=print_table,
args=(21,31)) # Start both threads
t1.star
t()
t2.star
t()
t3.star
t()
# Wait for both threads to
complete
t1.join()
t2.join()
t3.join()
```

Output:

```
1
11
21
2
12
22
3
13
23
4
24
14
5
25
```

—---

```
#exception handling
Practical No. 12
try:
 x=10/0
except ZeroDivisionError:
 print("Cannot divide by 0")

def check_age(age):
 if age<18:
 raise ValueError("Age must be 18 or above")
 return "Access granted"
try:
 print(check_age(16))
except ValueError as e:
 print(e)
```

```
#multiple except block under one try block
try:
 value = int(input("Enter a number: "))
 result = 10 / value
except ValueError:
 print("Error: You must enter a valid integer.")
except ZeroDivisionError:
 print("Error: You cannot divide by zero.")
except Exception as e:
 print(f"An unexpected error occurred: {e}")
```

 Output :
1st Run

```
 Cannot divide by 0
Age must be 18 or above
Enter a number: fg
Error: You must enter a valid integer.
```

2nd Run

```
Cannot divide by 0
Age must be 18 or above
Enter a number: 0
Error: You cannot divide by zero.
```

—--

```
# Different file handling operations
```

Practical No 13

 Program 1:

```
#1. Writing and Reading a new file
with open("First.txt","w") as file:
 file.write("This is a sample text file.\n")
 file.write("This is file handling program.\n")
print("File written successfully.")
#reading from a file
with open("First.txt","r") as file:
 content=file.read()
 print("File contents:\n",content)
```

 Output :

```
File written successfully.
File contents:
 This is a sample text file.
This is file handling program.
```

 Program 2:

```
#2.Changing content of that file
with open("First.txt","w") as file:
 file.write("This is Python Lab\n")
 file.write("This is Vrushali Sakpal.\n")
print("File written successfully.")
#reading from a file
```

```
with open("First.txt","r") as file:
 content=file.read()
 print("File contents:\n",content)
```

 Output :

```
File written successfully.
File contents:
 This is Python Lab
This is Vrushali Sakpal.
```

Program 3:

```
#3. Appending Content
with open("First.txt","a") as file:
 file.write("This is File Handling Program\n")
 file.write("Experiment No 13.\n")
print("File written successfully.")
#reading from a file
with open("First.txt","r") as file:
 content=file.read()
 print("File contents:\n",content)
```

Output:

```
File written successfully.
File contents:
 This is Python Lab
This is Vrushali Sakpal.
This is File Handling Program
Experiment No 13.
```

Program No. 4

```
#4.Deleting file from directory
import os
file_name="First.txt"
if os.path.exists(file_name):
 os.remove(file_name)
 print("The given file has been deleted.")
else:
 print("File does not exist.")
```

Output:

1st Run :

```
The given file has been deleted
```

2nd Run :

```
File does not exist.
```

—---

```
#Design student information registration form
```

Practical No. 14

```
import
```

```
tkinter as tk
from tkinter import messagebox
def submit_form():
 name = entry_name.get()
 roll_no = entry_roll.get()
 gender = entry_gender.get()
 dept = entry_dept.get()
 email = entry_email.get()
 contact = entry_contact.get()
 if name and roll_no and gender and dept and email and contact:
 messagebox.showinfo("RegistraƟ on", f"Student {name} registered successfully!")
 clear_form()
 else:
 messagebox.showwarning("Input Error", "Please fill all the fields.")
def clear_form():
 entry_name.delete(0, tk.END)
 entry_roll.delete(0, tk.END)
 entry_gender.delete(0, tk.END)
 entry_dept.delete(0, tk.END)
 entry_email.delete(0, tk.END)
 entry_contact.delete(0, tk.END)
root = tk.Tk()
root.Ɵ tle("Student RegistraƟ on Form")
root.geometry("400x450")
root.resizable(False, False)
tk.Label(root, text="Student RegistraƟ on Form", font=("Arial", 16, "bold")).pack(pady=10)
form_frame = tk.Frame(root)
form_frame.pack(pady=10)
tk.Label(form_frame, text="Full Name:").grid(row=0, column=0, sƟ cky='w', padx=10,
pady=5)
entry_name = tk.Entry(form_frame, width=30)
entry_name.grid(row=0, column=1, pady=5)
tk.Label(form_frame, text="Roll Number:").grid(row=1, column=0, sƟ cky='w', padx=10,
pady=5)
entry_roll = tk.Entry(form_frame, width=30)
entry_roll.grid(row=1, column=1, pady=5)
tk.Label(form_frame, text="Gender:").grid(row=2, column=0, sƟ cky='w', padx=10, pady=5)
entry_gender = tk.Entry(form_frame, width=30)
entry_gender.grid(row=2, column=1, pady=5)
tk.Label(form_frame, text="Department:").grid(row=3, column=0, sƟ cky='w', padx=10,
pady=5)
entry_dept = tk.Entry(form_frame, width=30)
entry_dept.grid(row=3, column=1, pady=5)
tk.Label(form_frame, text="Email ID:").grid(row=4, column=0, sƟ cky='w', padx=10, pady=5)
entry_email = tk.Entry(form_frame, width=30)
entry_email.grid(row=4, column=1, pady=5)
tk.Label(form_frame, text="Contact No.:").grid(row=5, column=0, sƟ cky='w', padx=10,
pady=5)
```

entry_contact = tk.Entry(form_frame, width=30)
entry_contact.grid(row=5, column=1, pady=5)
tk.BuΣ on(root, text="Submit", command=submit_form, bg="green", fg="white",
width=15).pack(pady=10)
tk.BuΣ on(root, text="Clear", command=clear_form, bg="red", fg="white", width=15).pack()
root.mainloop()

Output: student registration form

—---
# Implement plot using matplotlib library
Practical No. 15
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [10, 20, 25, 30, 40]
plt.plot(x, y, marker='o')
plt.title("Simple Line Graph")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.grid(True)
plt.show()

Output : graph