Python Practicals

Q1. Write a program related to functions and modules.

import math

def calculate\_rectangle\_area(length, width): area = length \* width

return area

rectangle\_area = calculate\_rectangle\_area(5, 3) print("The area of the rectangle is:", rectangle\_area)

square\_root = math.sqrt(16)

print("The square root of 16 is:", square\_root)

Output:-

The area of the rectangle is: 15 The square root of 16 is: 4.0

Q2. Program to demonstrate the use of dictionary and related functions.

dict = {1:"Prem", 2 : "Vaishnavi", 3 : "Om"} print("The length of dict is :", len(dict)) dict\_copy = dict.copy()

print("This is a copy of dict :", dict\_copy) print("The keys in the dict are: ",dict.keys()) print("The values in the dict are:", dict.values()) print("The items of the dict are :", dict.items())

# Output:

The length of dict is : 3

This is a copy of dict : {1: 'Prem', 2: Vaishnavi, 3: 'Om'} The keys in the dict are: dict\_keys([1, 2, 3])

The values in the dict are: dict\_values(['Prem', Vaishnavi, 'Om'])

The items of the dict are : dict\_items([(1, 'Prem'), (2, Vaishnavi), (3, 'Om')])

Q3. Program to demonstrate the working of classes and objects.

class calculate:

def square\_area(x): return x\*x

area = print("The area of square", calculate.square\_area(4)) Output:

The area of square 16

Q4. Write a program to demonstrate working of inheritance.

# Single:

class Vehicle:

def Vehicle\_info(self): print("Inside the parent class")

class Car(Vehicle): def car\_info(self):

print("Inside the derived class")

car= Car() car.Vehicle\_info()

Output:

Inside the parent class

# Multiple Inheritance:

class Person:

def person\_info(self, name, age): print("Inside person class") print("Name : ",name," Age :", age)

class Company:

def company\_info(self,company\_name,location): print("Inside Company Class")

print("Name: ",company\_name, ' location : ',location)

class Employee(Person, Company): def Employe\_info(self, salary, skill):

print("Inside employee class")

print("Salary : ", salary , ' Skill : ',skill)

emp = Employee() emp.person\_info('Prem', 21) emp.Employe\_info('IBM', 'Pune')

Output:

Inside person class Name : Prem Age : 21 Inside employee class

Salary : IBM Skill : Pune

# Multi-Level

class Animal:

def speak(self): print("Animal Speaking")

class Dog(Animal): def bark(self):

print("dog barking") class DogChild(Dog):

def eat(self): print("Eating bread...")

d = DogChild() d.bark()

d.speak()

d.eat()

Output:

dog barking Animal Speaking Eating bread...

Hierarchical

class Vehicle:

def info(self):

print("This is Vehicle") class Car(Vehicle):

def car\_info(self, name): print("Car name is:", name)

class Truck(Vehicle):

def truck\_info(self, name): print("Truck name is:", name)

obj1 = Car() obj1.info() obj1.car\_info('BMW') obj2 = Truck() obj2.info() obj2.truck\_info('Ford')

Output:

This is Vehicle

Car name is: BMW This is Vehicle Truck name is: Ford

Hybrid

class Vehicle:

def vehicle\_info(self): print("Inside Vehicle class")

class Car(Vehicle): def car\_info(self):

print("Inside Car class") class Truck(Vehicle):

def truck\_info(self): print("Inside Truck class")

class SportsCar(Car, Vehicle): def sports\_car\_info(self):

print("Inside SportsCar class") s\_car = SportsCar() s\_car.vehicle\_info() s\_car.car\_info() s\_car.sports\_car\_info()

Output:

Inside Vehicle class Inside Car class Inside SportsCar class

Q5. Demonstrate the working of Overloading methods and operator.

class Add:

def init (self,a): self.a = a

def add (self,o): return self.a + o.a

ob1 = Add(1) ob2 = Add(2)

obj1 = Add("Prem")

obj2 = Add(" is a Student")

print(ob1 + ob2) print(obj1 + obj2)

Output:

3

Prem is a Student

Q6. Program to demonstrate Exception handling mechanism.

x = 10

try:

divide = x/0

print("The output of the above division is:",divide)

except ZeroDivisionError as e: print("An error occurred:",e)

Output:

An error occurred: division by zero

Q7. Demonstrate Regular Expression in python.

import re

txt = "The rain in Spain"

x = re.search("^The.\*Spain$", txt) if x:

print("YES! We have a match!") else:

print("No match")

Output:

YES! We have a match!

Q8. Demonstrate RadioButton, checkbox, DialogBoxes using python tkinter.

from tkinter import \* top = Tk()

top.geometry("400x400") def selection():

selection = "You have selected: "+ str(radio.get()) label.config(text=selection)

radio = IntVar()

option1 = Radiobutton(top, value = 1, text="Nodejs", variable = radio, command = selection, textvariable = "Nodejs")

option1.pack()

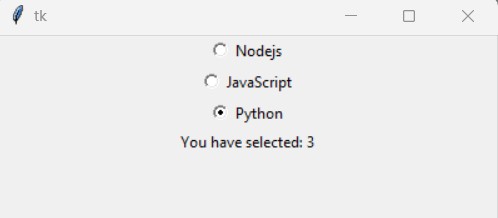
option2 = Radiobutton(top, value = 2, text="JavaScript", variable = radio, command = selection, textvariable = "JavaScript")

option2.pack()

option3 = Radiobutton(top, value = 3, text="Python", variable = radio, command = selection, textvariable = "Python")

option3.pack() label = Label(top) label.pack() top.mainloop

Output:



Q9. Learn GUI using Tkinter.

from tkinter import \* top = Tk()

top.geometry("400x400") def selection():

selection = "You have selected: "+ str(radio.get()) label.config(text=selection)

radio = IntVar()

option1 = Radiobutton(top, value = 1, text="Nodejs", variable = radio, command = selection, textvariable = "Nodejs")

option1.pack()

option2 = Radiobutton(top, value = 2, text="JavaScript", variable = radio, command = selection, textvariable = "JavaScript")

option2.pack()

option3 = Radiobutton(top, value = 3, text="Python", variable = radio, command = selection, textvariable = "Python")

option3.pack() label = Label(top) label.pack()

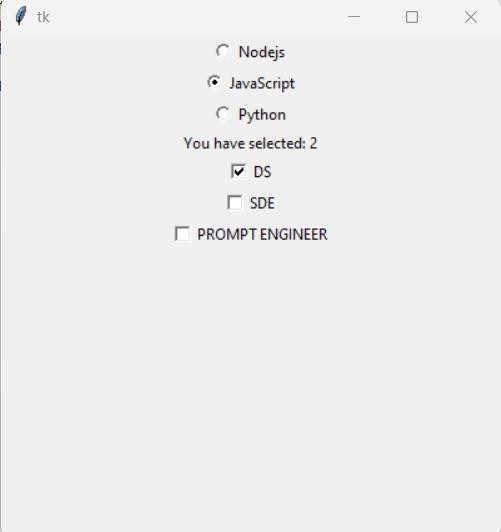
check1 = Checkbutton(top, text = "DS") check1.pack()

check2 = Checkbutton(top, text = "SDE") check2.pack()

check2 = Checkbutton(top, text = "PROMPT ENGINEER") check2.pack()

top.mainloop

Output:-



Q10. Program to create a database for insert, update, and delete in SQL.

import mysql.connector

try:

mydb = mysql.connector.connect( host="localhost",

user="root", password="Prem@2715",

auth\_plugin = "mysql\_native\_password"

)

mycursor = mydb.cursor() mycursor.execute("CREATE DATABASE PREM") mycursor.execute("USE PREM")

mycursor.execute("CREATE TABLE Employee (name VARCHAR(255), profession VARCHAR(255))")

sql = ("INSERT INTO Employee (name, profession) VALUES (%s,%s)") val = ("Prem Rathod","Data Scientist")

mycursor.execute(sql, val)

update = "UPDATE Employee SET name = 'Yug Rathod' WHERE name = 'Prem Rathod'"

mydb.commit()

delete = "DELETE FROM Employee WHERE name = 'Yug Rathod'" mydb.commit()

except Exception as e: print("An error occurred:", e)