

Python

Solved Slips



STUDENT NAME: TELORE GANESH BHASKAR
GUIDE: PROF.LANDE R.D

SLIP-1

Q1. Write a Python program to accept n numbers in list and remove duplicates from a list

Ans:

```
n=int(input("Enter Limit: "))
myarr=[]
for i in range(0,n):
    val=int(input(f"Enter element {i}:"))
    if val in myarr:
        print("")
    else:
        myarr.append(val)

print(myarr)
```

SLIP-1

Q2. Write Python GUI program to take accept your birthdate and output your age when a button is pressed.

Ans:

```
from tkinter import *
from tkinter import messagebox
gui=Tk()
gui.geometry("800x800")
gui.title("Slip-1")
gui.config(bg="#000")
def calcage():
    bdate=tb1.get()
    byear=bdate[-4:]
    age=2023-int(byear)
    messagebox.showinfo("Output",f"Your age is : {age}")
lbl=Label(gui,text="Enter Birthdate: ",font=("Arial",20),fg="#fff",bg="#000")
tb1=Entry(gui,font=("Arial",20),bg="#fff")
btn=Button(gui,text="Submit",font=("Arial",17),bg="#fff",fg="black",bd=0,width
=17,command=calcage)
lbl.place(x=100,y=100)
tb1.place(x=310,y=100)
btn.place(x=340,y=200)
gui.mainloop()
```

SLIP-2

Q.1 Write a Python function that accepts a string and calculate the number of upper case letters and lower case letters. Sample String: 'The quick Brown Fox' Expected Output: No. of Upper case characters: 3 No. of Lower case characters: 13

Ans:

```
string="The quick Brown Fox"
```

```
upp=0
```

```
lw=0
```

```
length=len(string)
```

```
for i in string:
```

```
    if i.isupper():
```

```
        upp=upp+1
```

```
    elif i.islower():
```

```
        lw=lw+1
```

```
print(f"Uppercase letters: {upp} \n Lowercase letters: {lw}")
```

SLIP-2

Q2. Write Python GUI program to create a digital clock with Tkinter to display the time.

Ans:

```
from tkinter import *
from time import strftime
gui=Tk()
gui.geometry("800x800")
gui.title("Q1")
def ctime():
    string=strftime("%I:%M:%S %p")
    lbl.config(text=string)
    lbl.after(1000,ctime)
lbl=Label(gui,font=("Arial",30,"bold"),bg="red")
lbl.place(x=300,y=300)
ctime()
gui.mainloop()
```

SLIP-3

Q1. Write a Python program to check if a given key already exists in a dictionary. If key exists replace with another key/value pair.

Ans:

```
dictionary={1:"Ganesh",2:"Prashant",3:"Sarthak"}
```

```
key=int(input("Enter key to search: "))
```

```
if key in dictionary.keys():
```

```
    newkey=int(input("Enter new key: "))
```

```
    if newkey in dictionary.keys():
```

```
        print("key already exists")
```

```
    else:
```

```
        del dictionary[key]
```

```
        dictionary[newkey]=input("Enter new value: ")
```

```
        print(dictionary)
```

```
else:
```

```
    print(dictionary[key])
```

SLIP-3

Q2. Write a python script to define a class student having members roll no, name, age, gender. Create a subclass called Test with member marks of 3 subjects. Create three objects of the Test class and display all the details of the student with total marks.

Ans:

```
class Student:
```

```
    def __init__(self, rollno, name, age, gender):
```

```
        print(f"Rollno: {rollno}")
```

```
        print(f"Name: {name}")
```

```
        print(f"Age: {age}")
```

```
        print(f"Gender: {gender}")
```

```
class Test(Student):
```

```
    def __init__(self, rollno, name, age, gender, m1, m2, m3):
```

```
        print(f"Mark1: {m1}")
```

```
        print(f"Mark2: {m2}")
```

```
        print(f"Mark3: {m3}")
```

```
        print(f"Total Marks: {m1+m2+m3}")
```

```
        super().__init__(rollno, name, age, gender)
```

```
ob1=Test(1,"Ganesh",19,"Male",95,92,97)
```

```
ob2=Test(1,"Prashant",19,"Male",95,92,97)
```

```
ob3=Test(1,"Sarthak",19,"Male",95,92,97)
```

SLIP-4

Q1. Write Python GUI program to create background with changing colors

Ans:

```
from tkinter import *
import random
gui=Tk()
gui.geometry("800x800")
gui.title("Q1")
def cngcolor():
    color=["green","blue","cyan","red","yellow","black"]
    selcolor=random.choice(color)
    gui.config(bg=selcolor)
    gui.after(1000,cngcolor)
cngcolor()
gui.mainloop()
```


SLIP-4

Q2. Define a class Employee having members id, name, department, salary. Create a subclass called manager with member bonus. Define methods accept and display in both the classes. Create n objects of the manager class and display the details of the manager having the maximum total salary (salary+bonus).

Ans:

```
class Employee:
```

```
    def __init__(self, memberid, name, dept, salary):
```

```
        self.memberid = memberid
```

```
        self.name = name
```

```
        self.dept = dept
```

```
        self.salary = salary
```

```
    def display(self):
```

```
        print("Member ID:", self.memberid)
```

```
        print("Member Name:", self.name)
```

```
        print("Member Department:", self.dept)
```

```
        print("Member Salary:", self.salary)
```

```
class Manager(Employee):
```

```
    bonus = None
```

```
    def __init__(self, memberid, name, dept, salary, bonus):
```

```
        super().__init__(memberid, name, dept, salary)
```

```
        self.bonus = bonus
```

```
def display(self):  
    super().display()  
    total = self.salary + self.bonus  
    print("Total Salary:", total)
```

```
def display_max_salary(*employees):  
    max_employee = max(employees, key=lambda emp: emp.salary)  
    max_employee.display()
```

```
ob1 = Manager(1, "GT", "Management", 100000, 15000)  
ob1.display()  
ob2 = Manager(2, "PT", "Marketing", 80000, 10000)  
ob2.display()  
ob3 = Manager(3, "SS", "Sales", 110000, 20000)  
ob3.display()  
print("Details of Maximum Salary Employee:")  
display_max_salary(ob1, ob2, ob3)
```

SLIP-5

Q1. Write a Python script using class, which has two methods get_String and print_String. get_String accept a string from the user and print_String print the string in upper case.

Ans:

```
class Q1:
```

```
    String=None
```

```
    def get_string(self,string):
```

```
        self.String=string
```

```
    def print_string(self):
```

```
        print("String in uppercase: ",self.String.upper())
```

```
obj=Q1()
```

```
string=input("Enter a String: ")
```

```
obj.get_string(string)
```

```
obj.print_string()
```

SLIP-5

Q2. Write a python script to generate Fibonacci terms using generator function.

Ans:

```
def generate(n):  
    n1=1;  
    n2=1;  
    print(0)  
    print(1)  
    print(1)  
    for i in range(3,n):  
        res=n1+n2  
        print(res)  
        n1=n2  
        n2=res  
  
generate(8)
```

GT

SLIP-6

Q1. Write python script using package to calculate area and volume of cube and sphere

Ans:

```
import math
```

```
class Q1:
```

```
    def __init__(self, edge, radius):
```

```
        print(f"Area of cube :{6*edge*edge}")
```

```
        print(f"Volume of cube: {edge*edge*edge}")
```

```
        print(f"Area of sphere :{4*math.pi*radius*radius}")
```

```
        print(f"Volume of sphere: {4//3*math.pi*radius*radius*radius}")
```

```
radius=int(input("Enter radius: "))
```

```
edge=int(input("Enter edge: "))
```

```
obj=Q1(edge,radius)
```

SLIP-6

Q2. Write a Python GUI program to create a label and change the label font style (font name, bold, size). Specify separate check button for each style.

Ans:

```
from tkinter import *
def changestyle():
    lbl.config(font=(cv1.get(),30,"bold"))
gui=Tk()
gui.title("Q2")
gui.geometry("800x800")
gui.config(bg="#000")
lbl=Label(gui,text="I Am GT",bg="#000",fg="#fff")
cv1=StringVar()
cb1=Checkbutton(gui,text="New Times Roman",variable=cv1,onvalue="New Times Roman",offvalue="",bg="#000",fg="#fff",command=changestyle)
cb2=Checkbutton(gui,text="Algerian",variable=cv1,onvalue="Algerian",offvalue="",bg="#000",fg="#fff",command=changestyle)
lbl.place(x=100,y=50)
cb1.place(x=100,y=150)
cb2.place(x=250,y=150)
gui.mainloop()
```

SLIP-7

Q1. Write Python class to perform addition of two complex numbers using binary + operator overloading.

Ans:

```
class Q1:
```

```
    def __init__(self,o):
```

```
        self.o=o
```

```
    def __add__(self,other):
```

```
        return complex(self.o)+complex(other.o)
```

```
ob1=Q1(5)
```

```
ob2=Q1(6)
```

```
print("Addition of two complex number is: ",ob1+ob2)
```

SLIP-7

Q2. Write python GUI program to generate a random password with upper and lower case letters.

Ans:

```
from tkinter import *
import random
import string
def generate():

    pwd=random.choices(string.ascii_uppercase+string.digits+string.ascii_lowercase,k=8)
    lbl.config(text=pwd)
gui=Tk()
gui.geometry("800x800")
btn=Button(gui,text="Generate",font=("Times New Roman",20,"bold"),command=generate)
lbl=Label(gui,font=("Times New Roman",20,"bold"))
btn.place(x=400,y=100)
lbl.place(x=400,y=200)
gui.mainloop()
```


SLIP-8

Q1. Write a python script to find the repeated items of a tuple

Ans:

```
Tuple=(1,1,2,4,4,5)
```

```
repeated=[]
```

```
for i in Tuple:
```

```
    if Tuple.count(i)>1 and i not in repeated:
```

```
        repeated.append(i)
```

```
print(repeated)
```

GT

SLIP-8

Q2. Write a Python class which has two methods get_String and print_String. get_String accept a string from the user and print_String print the string in upper case. Further modify the program to reverse a string word by word and print it in lower case

Ans:

```
class Q2:
```

```
    String=None
```

```
    def get_string(self,string):
```

```
        self.String=string
```

```
    def print_string(self):
```

```
        print("String in uppercase: ",self.String.upper())
```

```
        rev=self.String[::-1]
```

```
        print("String in reverse and lowercase: ",rev.lower())
```

```
obj=Q2()
```

```
string=input("Enter a String: ")
```

```
obj.get_string(string)
```

```
obj.print_string()
```

SLIP-9

Q1. Write a Python script using class to reverse a string word by word

Ans:

```
class Class:
```

```
    def __init__(self,para):
```

```
        print("String in reverse order: ",para[::-1])
```

```
obj=Class("Ganesh Telore")
```

GT

SLIP-9

Q2. Write Python GUI program to accept a number n and check whether it is Prime, Perfect or Armstrong number or not. Specify three radio buttons.

Ans:

```
from tkinter import *
import math

def checknum():
    if(int(rb.get())==1):
        flag=False
        num=int(tb1.get())
        if(num==1):
            messagebox.showinfo("Alert",f"{num} is not Prime Number.")
        elif num>1:
            for i in range(2,num):
                if(num%i)==0:
                    flag=True
                    break

            if flag:
                messagebox.showinfo("Alert",f"{num} is not Prime Number.")
            else:
                messagebox.showinfo("Alert",f"{num} is Prime Number.")

    elif int(rb.get())==2:
```

```
num=int(tb1.get())
summ=0
for i in range(1,num):
    if num%i==0:
        summ=summ+i
if(summ==num):
    messagebox.showinfo("Output","Number is Perfect")
else:
    messagebox.showinfo("Output","Number is not Perfect")
elif int(rb.get())==3:
    num=tb1.get()
    length=len(num)
    summ=0
    for i in num:
        summ=summ+int(i)**length

    if(summ==int(num)):
        messagebox.showinfo("Output","Number is Armstrong")
    else:
        messagebox.showinfo("Output","Number is not Armstrong")
```

```
from tkinter import messagebox
gui=Tk()
gui.geometry("800x800")
```

```
gui.config(bg="#000")
lbl1=Label(gui,text="Enter a Number: ", font=("New Times Roman",20),
bg="#000", fg="#fff")
tb1=Entry(gui, font=("New Times Roman",20))
lbl2=Label(gui,text="Select Option: ",font=("Arial",20),bg="#000",fg="#fff")
rb=IntVar()
r1=Radiobutton(gui,text="Prime
Number",variable=rb,value=1,font=("Arial",20),bg="#000",fg="#fff",command=
checknum)
r2=Radiobutton(gui,text="Perfect
Number",variable=rb,value=2,font=("Arial",20),bg="#000",fg="#fff",command=
checknum)
r3=Radiobutton(gui,text="Armstrong
Number",variable=rb,value=3,font=("Arial",20),bg="#000",fg="#fff",command=
checknum)
lbl1.place(x=100,y=50)
tb1.place(x=100,y=100)
lbl2.place(x=100,y=150)
r1.place(x=100,y=200)
r2.place(x=100,y=250)
r3.place(x=100,y=300)
gui.mainloop()
```

SLIP-10

Q1. Write Python GUI program to display an alert message when a button is pressed.

Ans:

```
from tkinter import *  
def click():  
    messagebox.showinfo("Alert","Button is Pressed..!")  
from tkinter import messagebox  
gui=Tk()  
gui.geometry("800x800")  
gui.config(bg="#000")  
btn=Button(gui,text="Click  
Me",font=("Arial",20),bg="#000",fg="#fff",command=click)  
btn.place(x=340,y=250)  
gui.mainloop()
```

SLIP-10

Q2. Write a Python class to find validity of a string of parentheses, '(', ')', '{', '}', '[' ']'. These brackets must be close in the correct order. for example "()" and "([]{})" are valid but "[]", "{[]}" and "{{{" are invalid.

Ans:

class Q2:

```
    def valid(self, string):
```

```
        data=[]
```

```
        brackets={')': '(', '}': '{', ']': '['}
```

```
        for i in string:
```

```
            if i in brackets:
```

```
                top=data.pop() if data else "
```

```
                if brackets[i]!=top:
```

```
                    return False
```

```
            else:
```

```
                data.append(i)
```

```
        return not data
```

```
ob=Q2()
```

```
para=input("Enter parantheses sequence: ")
```

```
output=ob.valid(para)
```

```
if(output==True):
```

```
    print("Parentheses are valid.")
```

```
else:
```

```
    print("Parantheses are invalid.")
```


SLIP-11

Q1. Write a Python program to compute element-wise sum of given tuples.
Original lists: (1, 2, 3, 4) (3, 5, 2, 1) (2, 2, 3, 1) Element-wise sum of the said
tuples: (6, 9, 8, 6)

Ans:

```
tuple1=(1,2,3,4)
tuple2=(3,5,2,1)
tuple3=(2,2,3,1)
res=tuple(map(sum,zip(tuple1,tuple2,tuple3)))
print(res)
```

GT

SLIP-11

Q2. Write Python GUI program to add menu bar with name of colors as options to change the background color as per selection from menu option.

Ans:

```
from tkinter import *

def red():
    gui.config(bg="red")

def cyan():
    gui.config(bg="cyan")

def pink():
    gui.config(bg="pink")

def green():
    gui.config(bg="green")

def magenta():
    gui.config(bg="magenta")

gui=Tk()
gui.geometry("800x800")
menubar=Menu(gui)
colors=Menu(menubar,tearoff=0)
colors.add_command(label="Red",command=red)
colors.add_command(label="Cyan",command=cyan)
colors.add_command(label="Pink",command=pink)
colors.add_command(label="Green",command=green)
```

```
colors.add_command(label="Magenta",command=magenta)
menubar.add_cascade(label="Colors",menu=colors)
gui.config(menu=menubar)
gui.mainloop()
```


GT

SLIP-12

Q1. Write a Python GUI program to create a label and change the label font style (font name, bold, size) using tkinter module.

Ans:

```
from tkinter import *  
gui=Tk()  
gui.geometry("1000x1000")  
gui.config(bg="#000")  
lbl=Label(gui,text="I Am GT", font=("Times New  
Roman",50,"bold"),fg="#fff",bg="#000")  
lbl.place(x=400,y=300)  
gui.mainloop()
```



SLIP-12

Q2. Write a python program to count repeated characters in a string. Sample string: 'thequickbrownfoxjumpsoverthelazydog' Expected output: o-4, e-3, u-2, h-2, r-2, t-2

Ans:

```
from collections import Counter
string="thequickbrownfoxjumpsoverthelazydog"
count=Counter(string)
repeated={char: count for char,count in count.items() if count>1}
for i,j in repeated.items():
    print(i, " :",j)
```

GT

SLIP-13

Q1. Write a Python program to input a positive integer. Display correct message for correct and incorrect input. (Use Exception Handling)

Ans:

```
class MyException(Exception):
```

```
    def __init__(self,msg):
```

```
        self.msg=msg
```

```
try:
```

```
    num=int(input("Enter a number: "))
```

```
    if(num<0):
```

```
        raise MyException("Input should be Postive, given Negative..!")
```

```
    else:
```

```
        print("Number is Positive.")
```

```
except ValueError:
```

```
    print("Invalid input")
```

```
except MyException as e:
```

```
    print(e)
```

SLIP-13

Q2. Write a program to implement the concept of queue using list.

Ans:

```
List=[]
```

```
n=int(input("Enter how many elements: "))
```

```
for i in range(0,n):
```

```
    data=int(input(f"Enter {i+1} element: "))
```

```
    List.append(data)
```

```
print("Elements of list in FIFO Manner: ",List)
```

```
index=int(input("Enter upto which index you want to remove elements: "))
```

```
for i in range(0,index):
```

```
    List.pop(0)
```

```
print("After Popping out elements in FIFO Manner: ",List)
```

SLIP-14

Q1. Write a Python GUI program to accept dimensions of a cylinder and display the surface area and volume of cylinder

Ans:

```
from tkinter import *
from tkinter import messagebox
def calculate():
    r=int(tb1.get())
    h=int(tb2.get())
    area=2*(3.14*r)*r+2*3.14*r*h
    vol=3.14*r*r*h
    messagebox.showinfo("Output",f"Area of cylinder: {area}\n Volume of cylinder: {vol}")
gui=Tk()
gui.geometry("800x800")
lbl=Label(gui,text="Enter Radius: ",font=("Arial",20,"bold"))
tb1=Entry(gui,font=("Arial",20,"bold"))
lbl2=Label(gui,text="Enter Height: ",font=("Arial",20,"bold"))
tb2=Entry(gui,font=("Arial",20,"bold"))
btn=Button(gui,text="Calculate",font=("Arial",20,"bold"),command=calculate)
lbl.place(x=100,y=50)
tb1.place(x=300,y=50)
lbl2.place(x=100,y=150)
tb2.place(x=300,y=150)
```



```
btn.place(x=350,y=250)  
gui.mainloop()
```

GT

SLIP-14

Q2. Write a Python program to display plain text and cipher text using a Caesar encryption.

Ans:

```
text=input("Enter Text: ")
key=int(input("Enter Key: "))
res=""
for i in text:
    if i.isalpha():
        if i.isupper():
            res=res+chr((ord(i)+key-65)%26+65)
        else:
            res=res+chr((ord(i)+key-97)%26+97)
    else:
        res=res+i

print("Plain Text: ",text)
print("Cipher Text: ",res)
```

SLIP-15

Q1. Write a Python class named Student with two attributes student_name, marks. Modify the attribute values of the said class and print the original and modified values of the said attributes

Ans:

```
class Student:
```

```
    student_name=None
```

```
    marks=None
```

```
    def __init__(self,para1="GT",para2=98):
```

```
        self.student_name=para1
```

```
        self.marks=para2
```

```
    def show(self):
```

```
        print(f"Student Name: {self.student_name}\nStudent Marks: {self.marks}")
```

```
ob1=Student("PT",89)
```

```
ob1.show()
```

SLIP-15

Q2. Write a python program to accept string and remove the characters which have odd index values of given string using user defined function.

Ans:

```
def oddremove(string):  
    res=""  
    for i in range(len(string)):  
        if i%2==0:  
            res=res+string[i]  
    return res
```

```
string="Hello, World!"  
op=oddremove(string)  
print(op)
```

GT

SLIP-16

Q1. Write a python script to create a class Rectangle with data member's length, width and methods area, perimeter which can compute the area and perimeter of rectangle.

Ans:

```
class Rectangle:
```

```
    def area(self,length,width):
```

```
        print(f"Area of Rectangle is: {length*width}")
```

```
    def perimeter(self,length,width):
```

```
        print(f"Perimeter of Rectangle is: {2*(length*width)}")
```

```
ob=Rectangle()
```

```
ob.area(5,5)
```

```
ob.perimeter(5,5)
```

SLIP-16

Q2. Write Python GUI program to add items in listbox widget and to print and delete the selected items from listbox on button click. Provide three separate buttons to add, print and delete.

Ans:

```
from tkinter import *
from tkinter import messagebox

def addlist():
    box.insert(0,"Ganesh")
    box.insert(1,"Prashant")
    box.insert(2,"Sarthak")

def printlist():
    for i in box.curselection():
        messagebox.showinfo("Output",box.get(i))

def deletelist():
    for i in box.curselection()[::-1]:
        box.delete(i)

gui=Tk()
gui.geometry("800x800")
box=Listbox(gui,font=("Arial",15),selectmode=MULTIPLE)
addbtn=Button(gui,text="Add",font=("Arial",15),command=addlist)
```

```
printbtn=Button(gui,text="Print",font=("Arial",15),command=printlist)
deletebtn=Button(gui,text="Delete",font=("Arial",15),command=deletelist)
box.place(x=50,y=50)
addbtn.place(x=120,y=300)
printbtn.place(x=50,y=300)
deletebtn.place(x=200,y=300)
gui.mainloop()
```

GT

SLIP-17

Q1. Write Python GUI program that takes input string and change letter to upper case when a button is pressed.

Ans:

```
from tkinter import *
from tkinter import messagebox
def changecase():
    string=tb1.get()
    messagebox.showinfo("Output",f"String in uppercase: {string.upper()}")
gui=Tk()
gui.geometry("800x800")
gui.config(bg="#000")
lbl=Label(gui,text="Enter String: ",font=("Arial",20,"bold"),bg="#000",fg="#fff")
tb1=Entry(gui,font=("Arial",20))
btn=Button(gui,text="Submit",font=("Arial",20),command=changecase)
lbl.place(x=100,y=100)
tb1.place(x=300,y=100)
btn.place(x=300,y=150)
gui.mainloop()
```


SLIP-17

Q2. Define a class Date (Day, Month, Year) with functions to accept and display it. Accept date from user. Throw user defined exception “invalid Date Exception” if the date is invalid.

Ans:

```
class InvalidDate(Exception):
```

```
    def __init__(self,msg):  
        self.msg=msg
```

```
class Date:
```

```
    day=None
```

```
    month=None
```

```
    year=None
```

```
    def getdata(self,day,month,year):
```

```
        self.day=day
```

```
        self.month=month
```

```
        self.year=year
```

```
    def validate(self):
```

```
        try:
```

```
            if self.day>31:
```

```
                raise InvalidDate("Error: Invalid Day Entered..!")
```

```
elif self.month<=0 and self.month>12:
    raise InvalidDate("Error: Invalid Month Entered..!")
elif self.year<0000:
    raise InvalidDate("Error: Invalid Year Entered..!")
else:
    print(f"Date is: {self.day}/{self.month}/{self.year}")
except ValueError:
    print("Invalid data entered, please enter numbers...")
except InvalidDate as e:
    print(e)
```

```
ob=Date()
```

```
try:
```

```
    day=int(input("Enter day: "))
```

```
    month=int(input("Enter month: "))
```

```
    year=int(input("Enter year: "))
```

```
    ob.getdata(day,month,year)
```

```
    ob.validate()
```

```
except ValueError:
```

```
    print("Invalid data entered, only numbers allowed")
```

SLIP-18

Q1. Create a list `a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]` and write a python program that prints out all the elements of the list that are less than 5

Ans:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

```
for i in a:
```

```
    if i<5:
```

```
        print(i)
```

GT

SLIP-18

Q2. Write a python script to define the class person having members name, address. Create a subclass called Employee with members staffed salary. Create 'n' objects of the Employee class and display all the details of the employee.

Ans:

```
class person:
```

```
    member_name=None
```

```
    address=None
```

```
    def __init__(self,para1,para2):
```

```
        self.member_name=para1
```

```
        self.address=para2
```

```
    def show(self):
```

```
        print("person name: ",self.member_name,"\nperson address: ",self.address)
```

```
class Employee(person):
```

```
    salary=None
```

```
    def __init__(self,para1,para2,para3):
```

```
        super().__init__(para1,para2)
```

```
        self.salary=para3
```

```
    def show2(self):
```

```
super().show()  
print("Employee salary: ",self.salary)
```

```
n=int(input("Enter how many objects: "))  
objects=[]  
for i in range(n):  
    name=input("Enter Name: ")  
    add=input("Enter Address: ")  
    sal=int(input("Enter Salary: "))  
    ob=Employee(name,add,sal)  
    objects.append(ob)  
  
for j in objects:  
    j.show2()
```

GT

SLIP-19

Q1. Write a Python GUI program to accept a number form user and display its multiplication table on button click.

Ans:

```
from tkinter import *
```

```
def table():
```

```
    num = int(tb1.get())
```

```
    for i in range(1,11):
```

```
        res=num*i
```

```
        box.insert(i,res)
```

```
gui=Tk()
```

```
gui.config(bg="#000")
```

```
gui.geometry("800x800")
```

```
lbl=Label(gui,text="Enter Number: ", font=("Arial",15),bg="#000",fg="#fff")
```

```
tb1=Entry(gui,font=("Arial",15))
```

```
btn=Button(gui,text="Calculate",font=("Arial",15),width=20,command=table)
```

```
box=Listbox(gui,font=("Arial",15),bg="#000",fg="#fff",width=50)
```

```
lbl.place(x=100,y=50)
```

```
tb1.place(x=300,y=50)
```

```
btn.place(x=300,y=150)
```

```
box.place(x=100,y=200)
```

```
gui.mainloop()
```

SLIP-19

Q2. Define a class named Shape and its subclass(Square/ Circle). The subclass has an init function which takes an argument (Lenght/redious). Both classes should have methods to calculate area and volume of a given shape.

Ans:

```
class Shape:
```

```
    pi=3.14
```

```
    print("Area and Volume of Square and Circle: ")
```

```
class Square(Shape):
```

```
    length=None
```

```
    def __init__(self,length):
```

```
        self.length=length
```

```
    def area(self):
```

```
        print("Area of square: ",self.length*self.length)
```

```
    def volume(self):
```

```
        print("Perimeter of square: ",self.length*4)
```

```
class Circle(Shape):
```

```
    radius=None
```

```
    def __init__(self,radius):
```

```
        self.radius=radius
```

```
    def area(self):
```

```
        print("Area of circle: ",self.pi*self.radius*self.radius)
```

```
def volume(self):  
    print("Circumference of circle: ",2*self.pi*self.radius)
```

```
length=float(input("Enter Length: "))  
radius=float(input("Enter Radius: "))  
ob=Square(length)  
ob1=Circle(radius)  
ob.area()  
ob.volume()  
ob1.area()  
ob1.volume()
```

GT

SLIP-20

Q1. Write a python program to create a class Circle and Compute the Area and the circumferences of the circle.(use parameterized constructor).

Ans:

```
class Circle:
```

```
    radius=None
```

```
    def __init__(self,r):
```

```
        self.radius=r
```

```
    def area(self):
```

```
        print(f"Area of circle: {3.14*self.radius*self.radius}")
```

```
    def circumference(self):
```

```
        print(f"Circumference of a circle: {2*3.14*self.radius}")
```

```
r=int(input("Enter Radius: "))
```

```
ob=Circle(r)
```

```
ob.area()
```

```
ob.circumference()
```

SLIP-20

Q2. Write a Python script to generate and print a dictionary which contains a number (between 1 and n) in the form(x,x*x). Sample Dictionary (n=5)
Expected Output: {1:1, 2:4, 3:9, 4:16, 5:25}

Ans:

```
dictionary={}
```

```
n=5
```

```
for i in range(1,n+1):
```

```
    dictionary[i]=i*i
```

```
print(dictionary)
```

GT

SLIP-21

Q1. Define a class named Rectangle which can be constructed by a length and width. The Rectangle class has a method which can compute the area and Perimeter.

Ans:

```
class Rectangle:
```

```
    w=None
```

```
    l=None
```

```
    def __init__(self,para1,para2):
```

```
        self.w=para1
```

```
        self.l=para2
```

```
    def area(self):
```

```
        print("Area of a Rectangle :",self.w*self.l)
```

```
    def perimeter(self):
```

```
        print("Perimeter of a rectangle :",2*(self.l*self.w))
```

```
r=int(input("Enter radius: "))
```

```
w=int(input("Enter width: "))
```

```
ob=Rectangle(r,w)
```

```
ob.area()
```

```
ob.perimeter()
```

SLIP-21

Q2. Write a Python program to convert a tuple of string values to a tuple of integer values. Original tuple values: (('333', '33'), ('1416', '55')) New tuple values: ((333, 33), (1416, 55))

Ans:

```
Tuple1 = ('333', '33')
```

```
Tuple2 = ('1416', '55')
```

```
res1 = ()
```

```
res2 = ()
```

```
for i, j in Tuple1, Tuple2:
```

```
    res1 = res1+(int(i),)
```

```
    res2 = res2+(int(j),)
```

```
mytuple = tuple(zip(res1, res2))
```

```
print(mytuple)
```

SLIP-22

Q1. Write a python class to accept a string and number n from user and display n repetition of strings by overloading * operator

Ans:

class op:

```
def __init__(self,string):
```

```
    self.string=string
```

```
def __mul__(self,other):
```

```
    return self.string*other
```

```
string=input("Enter a string: ")
```

```
n=int(input("Enter repition limit: "))
```

```
ob=op(string)
```

```
print(ob*n)
```

SLIP-22

Q2. Write a python script to implement bubble sort using list

Ans:

```
def bubblesort(List):  
    n=len(List)  
    for i in range(n-1):  
        for j in range(n-i-1):  
            if List[j]>List[j+1]:  
                List[j],List[j+1]=List[j+1],List[j]
```

```
List=[6,2,1,7,5]  
bubblesort(List)  
print("Sorted List is: ",List)
```

GT

SLIP-23

Q1. Write a Python GUI program to create a label and change the label font style (font name, bold, size) using tkinter module.

Ans:

```
from tkinter import *  
gui=Tk()  
gui.geometry("1000x1000")  
gui.config(bg="#000")  
lbl=Label(gui,text="I Am GT",bg="#000",fg="#fff")  
lbl.place(x=100,y=100)  
lbl.config(font=("Arial",30,"bold"))  
gui.mainloop()
```

SLIP-23

Q2. Create a class circles having members radius. Use operator overloading to add the radius of two circle objects. Also display the area of circle.

Ans:

```
class circles:
```

```
    def __init__(self,para1):
```

```
        self.para1=para1
```

```
    def __add__(self,other):
```

```
        print("Area of circle: ",3.14*self.para1*self.para1)
```

```
        return self.para1+other.para1
```

```
ob=circles(3)
```

```
ob1=circles(3)
```

```
print(ob+ob1)
```


SLIP-24

Q1. Write a Python Program to Check if given number is prime or not. Also find factorial of the given no using user defined function.

Ans:

```
def fun(num):  
    flag=0  
    n=num  
    if(n==0 and n==1):  
        flag=1  
    for i in range(2,n):  
        if n%i==0:  
            flag=1  
            break  
  
    if(flag==0):  
        print("Number is prime.")  
    else:  
        print("Number is not prime")  
  
    res=1  
    for i in range(1,n+1):  
        res=res*i  
  
    print(res)
```

```
num=int(input("Enter a Number: "))  
fun(num)
```

GT

SLIP-24

Q2. Write Python GUI program which accepts a number n to displays each digit of number in words.

Ans:

```
from tkinter import *
def identifynum(digit):
    if digit==0:
        print("Zero",end=" ")
    elif digit=='1':
        print("One",end=" ")
    elif digit=='2':
        print("Two",end=" ")
    elif digit=='3':
        print("Three",end=" ")
    elif digit=='4':
        print("Four",end=" ")
    elif digit=='5':
        print("Five",end=" ")
    elif digit=='6':
        print("Six",end=" ")
    elif digit=='7':
        print("Seven",end=" ")
    elif digit=='8':
        print("Eight",end=" ")
```

```
elif digit=='9':  
    print("Nine",end=" ")
```

```
def shownuminwords():
```

```
    n=tb1.get()
```

```
    i=0
```

```
    l=len(n)
```

```
    while i<l:
```

```
        identifynum(n[i])
```

```
        i=i+1
```

```
gui=Tk()
```

```
gui.config(bg="#000")
```

```
gui.geometry("1000x1000")
```

```
lbl=Label(gui,text="Enter Number:",font=("Arial",20),bg="#000",fg="#fff")
```

```
tb1=Entry(gui,font=("Arial",20))
```

```
btn=Button(gui,text="Submit",font=("Arial",15),width=27,command=shownuminwords)
```

```
lbl.place(x=100,y=100)
```

```
tb1.place(x=300,y=100)
```

```
btn.place(x=300,y=200)
```

```
gui.mainloop()
```

SLIP-25

Q1. Write a Python function that accepts a string and calculate the number of upper case letters and lower case letters. Sample String : 'The quick Brown Fox' Expected Output : No. of Upper case characters : 3 No. of Lower case Characters : 12

Ans:

```
String="The quick Brown Fox"
```

```
upper=0
```

```
lower=0
```

```
for i in String:
```

```
    if i.isupper():
```

```
        upper=upper+1
```

```
    elif i.islower():
```

```
        lower=lower+1
```

```
print("Number of lowercase letters: ",lower)
```

```
print("Number of uppercase letters: ",upper)
```

GT

SLIP-25

Q2. Write a Python script to Create a Class which Performs Basic Calculator Operations.

Ans:

```
class Q2:
    val1=None
    val2=None
    def __init__(self,val1,val2):
        self.val1=val1
        self.val2=val2
    def add(self):
        print("Addition of two numbers: ",self.val1+self.val2)

    def sub(self):
        print("Subtraction of two numbers: ",self.val1-self.val2)

    def mult(self):
        print("Multiplication of two numbers: ",self.val1*self.val2)

    def div(self):
        print("Division of two numbers: ",self.val1//self.val2)

print("*****Operations*****")
```

```
print("1. Addition\n2. Subtraction\n3. Multiplication\n4. Division")
choice=input("Enter Choice: ")
val1=int(input("Enter First Number: "))
val2=int(input("Enter Second Number: "))
ob=Q2(val1,val2)
if choice=='1':
    ob.add()
elif choice=='2':
    ob.sub()
elif choice=='3':
    ob.mult()
elif choice=='4':
    ob.div()
```

GT

SLIP-26

Q1. Write an anonymous function to find area of square and rectangle.

Ans:

```
areasq=lambda side:side*side
arearect=lambda length,width:length*width
side=int(input("Enter Side: "))
length=int(input("Enter length: "))
width=int(input("Enter width: "))
areasquare=areasq(side)
arearectangle=arearect(length,width)
print("Area of Square: ",areasquare)
print("Area of Rectangle: ",arearectangle)
```


SLIP-26

Q2. Write Python GUI program which accepts a sentence from the user and alters it when a button is pressed. Every space should be replaced by *, case of all alphabets should be reversed, digits are replaced by?.

Ans:

```
from tkinter import *
def fun():
    data=tb1.get()
    data2=""
    for i in data:
        if i.isupper():
            data2=data2+i.lower()
        elif i.islower():
            data2=data2+i.upper()
        elif i.isspace():
            data2=data2+"*"
        elif i.isdigit():
            data2=data2+"?"
        else:
            data2=data2+i
    lbl2.config(text="Output: "+data2)

gui=Tk()
gui.geometry("1000x1000")
```

```
gui.config(bg="#000")
lbl=Label(gui,text="Enter String: ",font=("Arial",20),bg="#000",fg="#fff")
tb1=Entry(gui,font=("Arial",20))
btn=Button(gui,text="Submit",font=("Arial", 15),command=fun)
lbl2=Label(gui,text="Output: ",font=("Arial",20),bg="#000",fg="#fff")
lbl.place(x=100,y=100)
tb1.place(x=300,y=100)
btn.place(x=300,y=200)
lbl2.place(x=100,y=300)
gui.mainloop()
```

GT

SLIP-27

Q1. Write a Python program to unzip a list of tuples into individual lists.

Ans:

```
List=[(1,'GT'),(2,'PT'),(3,'SS')]
```

```
newList=list(zip(*List))
```

```
print(newList)
```

GT

SLIP-27

Q2. Write Python GUI program to accept a decimal number and convert and display it to binary, octal and hexadecimal number.

Ans:

```
from tkinter import *
from tkinter import messagebox
def convert():
    n=int(tb1.get())
    messagebox.showinfo("Output",f"Number in Binary: {bin(n)}\nNumber in Octal: {oct(n)}\nNumber in Hexadecimal: {hex(n)}")
gui=Tk()
gui.geometry("1000x1000")
lbl=Label(gui,text="Enter Number: ",font=("Arial",20))
tb1=Entry(gui,font=("Arial",20))
btn=Button(gui,text="Convert",font=("Arial",15),width=10,command=convert)
lbl.place(x=100,y=100)
tb1.place(x=300,y=100)
btn.place(x=200,y=200)
gui.mainloop()
```

SLIP-28

Q1. Write a Python GUI program to create a list of Computer Science Courses using Tkinter module (use Listbox).

Ans:

```
from tkinter import *
from tkinter import messagebox
gui=Tk()
gui.geometry("1000x1000")
lbl=Label(gui,text="Computer Science Courses: ",font=("Arial",20))
box=Listbox(gui,font=("Arial",20),width=23)
box.insert(0,"PHP")
box.insert(1,"Java")
box.insert(2,"Python")
box.insert(3,"HTML")
lbl.place(x=100,y=100)
box.place(x=100,y=150)
gui.mainloop()
```

SLIP-28

Q2. Write a Python program to accept two lists and merge the two lists into list of tuple.

Ans:

```
list1=[]
n=int(input("Enter limit: "))
for i in range(n):
    item=list(input(f"Enter {i+1} Number: "))
    list1=list1+item

list2=[]
n2=int(input("Enter limit: "))
for i in range(n2):
    item=list(input(f"Enter {i+1} Number: "))
    list2=list2+item

listtuple=list(zip(list1,list2))
print("List of Tuples: ",listtuple)
```

SLIP-29

Q1. Write a Python GUI program to calculate volume of Sphere by accepting radius as input.

Ans:

```
from tkinter import *
from tkinter import messagebox
def calculate():
    r=int(tb1.get())
    volume=1.33*3.14*r*r*r
    messagebox.showinfo("Output",f"Volume of Sphere :{volume}")

gui=Tk()
gui.geometry("800x800")
gui.config(bg="#000")
lbl=Label(gui,text="Enter Radius: ",font=("Arial",20),bg="#000",fg="#fff")
tb1=Entry(gui,font=("Arial",20))
btn=Button(gui,text="Submit",font=("Arial",20),command=calculate)
lbl.place(x=100,y=100)
tb1.place(x=300,y=100)
btn.place(x=300,y=200)
gui.mainloop()
```

SLIP-29

Q2. Write a Python script to sort (ascending and descending) a dictionary by key and value.

Ans:

```
dictionary={2:"Ganesh",3:"Prashant",1:"Mahadev"}  
print("Dictionary values in ascending order: ",sorted(dictionary.values()))  
print("Dictionary keys in ascending order: ",sorted(dictionary.keys()))  
print("Dictionary values in ascending order:  
",sorted(dictionary.values(),reverse=True))  
print("Dictionary keys in ascending order:  
",sorted(dictionary.keys(),reverse=True))
```


SLIP-30

Q1. Write a Python GUI program to accept a string and a character from user and count the occurrences of a character in a string

Ans:

```
from tkinter import *
from tkinter import messagebox
def count():
    string=tb1.get()
    count={}
    for i in string:
        if i in count:
            count[i]=count[i]+1
        else:
            count[i]=1
    messagebox.showinfo("Output",count)
gui=Tk()
gui.geometry("800x800")
gui.config(bg="#000")
lbl=Label(gui,text="Enter String: ",font=("Arial",20),bg="#000",fg="#fff")
tb1=Entry(gui,font=("Arial",20))
btn=Button(gui,text="Submit",font=("Arial",20),command=count)
lbl.place(x=100,y=100)
tb1.place(x=300,y=100)
btn.place(x=300,y=200)
gui.mainloop()
```

SLIP-30

Q2. Python Program to Create a Class in which One Method Accepts a String from the User and Another method Prints it. Define a class named Country which has a method called print Nationality. Define subclass named state from Country which has a mehtod called printState. Write a method to print state, country and nationality.

Ans:

```
class demo:
```

```
    def accept(self):
```

```
        self.x=input("Enter a String :")
```

```
    def display(self):
```

```
        print(f"Given String is :{self.x}")
```

```
class country:
```

```
    def accept(self):
```

```
        self.country=str(input("Enter country :"))
```

```
        self.nationality=str(input("Enter nationality :"))
```

```
    def displaycountry(self):
```

```
        print(f"Country is :{self.country}\n Nationality is :{self.nationality}")
```

```
class state(country):
```

```
def accept(self):  
    self.state=str(input("Enter the state :"))
```

```
def displaystate(self):  
    print(f"State is :{self.state}")
```

```
demo=demo()  
demo.accept()  
demo.display()  
country=country()  
country.accept()  
country.displaycountry()  
state=state()  
state.accept()  
state.displaystate()
```

GT

GT