Python-Slip1

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

First- Solution

```
marks =[]
n=int(input('Enter number of elements : '))
for i in range(n):
    value=int(input())
    marks.append(value)
print(marks)
new_marks=[]
for x in marks:
    if x not in new_marks:
        new_marks.append(x)
print(new_marks)
```

```
Enter number of elements : 5
1
2
3
5
2
[1, 2, 3, 5, 2]
[1, 2, 3, 5]
```

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

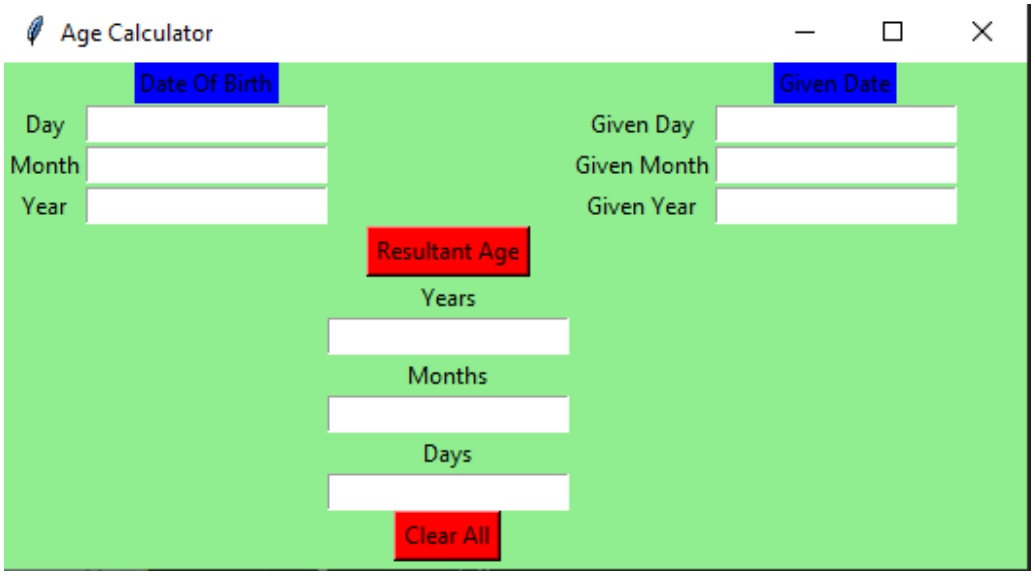
```
# import all functions from the tkinter
from tkinter import *

# import messagebox class from tkinter
from tkinter import messagebox

# Function for clearing the
# contents of all text entry boxes
def clearAll() :

    # deleting the content from the entry box
    dayField.delete(0, END)
    monthField.delete(0, END)
    yearField.delete(0, END)
    givenDayField.delete(0, END)
    givenMonthField.delete(0, END)
    givenYearField.delete(0, END)
    rsltDayField.delete(0, END)
    rsltMonthField.delete(0, END)
    rsltYearField.delete(0, END)

# function for checking error
def checkError() :

    # if any of the entry field is empty
    # then show an error message and clear
    # all the entries
    if (dayField.get() == "" or monthField.get() == ""
        or yearField.get() == "" or givenDayField.get() == ""
        or givenMonthField.get() == "" or givenYearField.get() == "") :

        # show the error message
        messagebox.showerror("Input Error")

        # clearAll function calling
        clearAll()

        return -1

# function to calculate Age
def calculateAge() :

    # check for error
```

```
value = checkError()

# if error is occur then return
if value ==  -1 :
    return

else :

    # take a value from the respective entry boxes
    # get method returns current text as string
    birth_day = int(dayField.get())
    birth_month = int(monthField.get())
    birth_year = int(yearField.get())

    given_day = int(givenDayField.get())
    given_month = int(givenMonthField.get())
    given_year = int(givenYearField.get())


    # if birth date is greater then given birth_month
    # then donot count this month and add 30 to the date so
    # as to subtract the date and get the remaining days
    month =[31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

    if (birth_day > given_day):
        given_month = given_month - 1
        given_day = given_day + month[birth_month-1]


    # if birth month exceeds given month, then
    # donot count this year and add 12 to the
    # month so that we can subtract and find out
    # the difference
    if (birth_month > given_month):
        given_year = given_year - 1
        given_month = given_month + 12

    # calculate day, month, year
    calculated_day = given_day - birth_day;
    calculated_month = given_month - birth_month;
    calculated_year = given_year - birth_year;

    # calculated day, month, year write back
```

```
        # to the respective entry boxes

        # insert method inserting the
        # value in the text entry box.

        rsltDayField.insert(10, str(calculated_day))
        rsltMonthField.insert(10, str(calculated_month))
        rsltYearField.insert(10, str(calculated_year))


# Driver Code
if __name__ == "__main__" :

    # Create a GUI window
    gui = Tk()

    # Set the background colour of GUI window
    gui.configure(background = "light green")

    # set the name of tkinter GUI window
    gui.title("Age Calculator")

     # Set the configuration of GUI window
    gui.geometry("525x260")

    # Create a Date Of Birth : label
    dob = Label(gui, text = "Date Of Birth", bg = "blue")

    # Create a Given Date : label
    givenDate = Label(gui, text = "Given Date", bg = "blue")

    # Create a Day : label
    day = Label(gui, text = "Day", bg = "light green")

    # Create a Month : label
    month = Label(gui, text = "Month", bg = "light green")

    # Create a Year : label
    year = Label(gui, text = "Year", bg = "light green")

    # Create a Given Day : label
    givenDay = Label(gui, text = "Given Day", bg = "light green")
```

```
    # Create a Given Month : label
    givenMonth = Label(gui, text = "Given Month", bg = "light green")

    # Create a Given Year : label
    givenYear = Label(gui, text = "Given Year", bg = "light green")

    # Create a Years : label
    rsltYear = Label(gui, text = "Years", bg = "light green")

    # Create a Months : label
    rsltMonth = Label(gui, text = "Months", bg = "light green")

    # Create a Days : label
    rsltDay = Label(gui, text = "Days", bg = "light green")

    # Create a Resultant Age Button and attached to calculateAge
function
    resultantAge = Button(gui, text = "Resultant Age", fg = "Black", bg =
"Red", command = calculateAge)

    # Create a Clear All Button and attached to clearAll function
    clearAllEntry = Button(gui, text = "Clear All", fg = "Black", bg = "Red",
command = clearAll)

    # Create a text entry box for filling or typing the information.
    dayField = Entry(gui)
    monthField = Entry(gui)
    yearField = Entry(gui)

    givenDayField = Entry(gui)
    givenMonthField = Entry(gui)
    givenYearField = Entry(gui)

    rsltYearField = Entry(gui)
    rsltMonthField = Entry(gui)
    rsltDayField = Entry(gui)


    # grid method is used for placing
    # the widgets at respective positions
    # in table like structure .
    dob.grid(row = 0, column = 1)
```

```
day.grid(row = 1, column = 0)
dayField.grid(row = 1, column = 1)

month.grid(row = 2, column = 0)
monthField.grid(row = 2, column = 1)

year.grid(row = 3, column = 0)
yearField.grid(row = 3, column = 1)

givenDate.grid(row = 0, column = 4)

givenDay.grid(row = 1, column = 3)
givenDayField.grid(row = 1, column = 4)

givenMonth.grid(row = 2, column = 3)
givenMonthField.grid(row = 2, column = 4)

givenYear.grid(row = 3, column = 3)
givenYearField.grid(row = 3, column = 4)

resultantAge.grid(row = 4, column = 2)

rsltYear.grid(row = 5, column = 2)
rsltYearField.grid(row = 6, column = 2)

rsltMonth.grid(row = 7, column = 2)
rsltMonthField.grid(row = 8, column = 2)

rsltDay.grid(row = 9, column = 2)
rsltDayField.grid(row = 10, column = 2)

clearAllEntry.grid(row = 12, column = 2)

# Start the GUI
gui.mainloop()
```



Pythone-Slip2 –

A.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

```
string=input('enter a string : ')
up=low=ele=0
for x in string:
    if x.isupper():
        up+=1
    elif x.islower():
        low+=1
    else:
        ele+=1
print('No. of Upper case characters : ',up)
```

```
print('No. of Lower case characters : ',low)
print('other special symbols',ele)print(count1)
print("The number of uppercase characters is:")
print(count2)
```

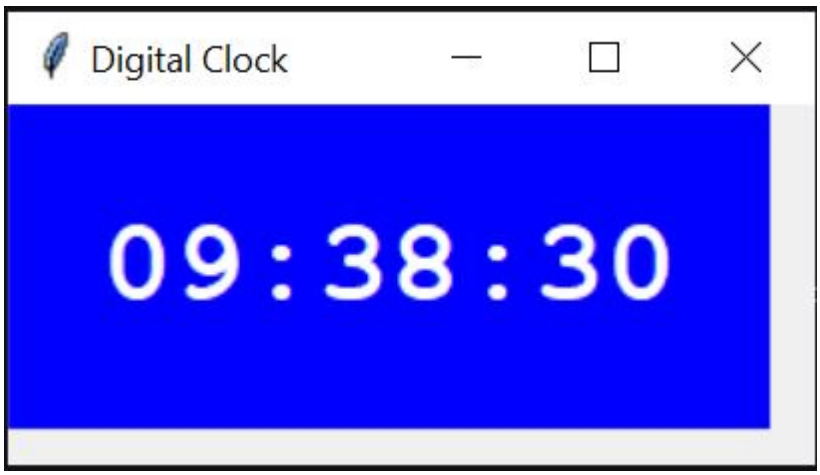
```
enter a string : Abcdefg
No. of Upper case characters :  1
No. of Lower case characters :  6
other special symbols 0
```

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

```
import time
from tkinter import *
canvas = Tk()
canvas.title("Digital Clock")
canvas.geometry("350x200")
canvas.resizable(1,1)
label = Label(canvas, font=("Courier", 30, 'bold'), bg="blue", fg="white", bd =30)
label.grid(row =0, column=1)
def digitalclock():
  text_input = time.strftime("%H:%M:%S")
  label.config(text=text_input)
  label.after(200, digitalclock)
digitalclock()
canvas.mainloop()
```

Output

Running the above code gives us the following result –



Pythone-Slip 3

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

```
  Dict={}
n=int(input('enter number of keys :'))
for x in range(0,n):
    key=input('enter the key :')
    if key in Dict:
        print('the given key exists!')

        for key in Dict.keys():
            print(key,end=' ')
        print('\n ^ use differnt keys from above list')
        key=input('enter the key :')
    value=input('enter the value :')
    Dict[key]=value
print(Dict)
```

```
enter number of keys :2
enter the key :abc
enter the value :200
enter the key :pqr
enter the value :300
{'abc': '200', 'pqr': '300'}
```

B) 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.

```
class Student():
    def __init__(self,roll_no,name,age,gender):
        self.roll_no=roll_no
        self.name=name
        self.age=age
        self.gender=gender


class Test(Student):
    def
__init__(self,roll_no,name,age,gender,sub1mark,sub2mark,sub3mark,):
        super().__init__(roll_no,name,age,gender)
        self.mark1=sub1mark
        self.mark2=sub2mark
        self.mark3=sub3mark


    def get_marks(self):
        self.total=self.mark1+self.mark2+self.mark3
        print(self.name , "\b's marks:", self.total)
```

```
        print("sub1 marks :",self.mark1)
        print("sub2 marks :",self.mark2)
        print("sub3 marks :",self.mark3)

p1=Test(1,"amar",19,'male',82,89,76)
p2=Test(2,'priya',20,'female',94,91,84)
p1.get_marks()
p2.get_marks()
```

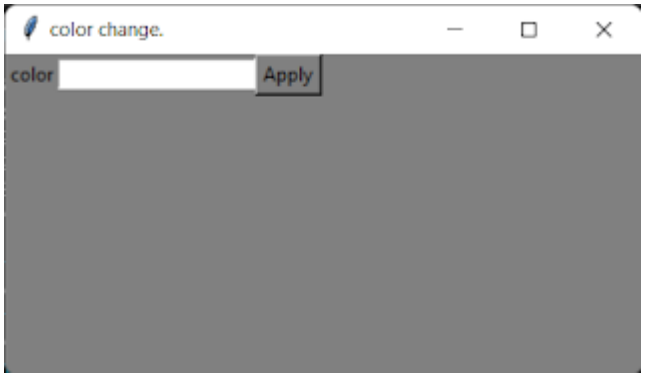
```
abc's marks: 247
sub1 marks : 82
sub2 marks : 89
sub3 marks : 76
pqr's marks: 269
sub1 marks : 94
sub2 marks : 91
sub3 marks : 84
```

Pythone-Slip 4

A) Write Python GUI program to create background with changing colors

```
from tkinter import Button, Entry, Label, Tk

def changecolor():

        newvalue = value.get()

    gui.configure(background = newvalue)
```

```
gui=Tk()

gui.title("color change.")

gui.configure(background = "gray")

gui.geometry("400x300")

color = Label(gui, text = "color", bg = "gray")

value = Entry(gui)

apply = Button(gui, text = "Apply", fg = "Black", bg = "gray", command
= changecolor)

    color.grid(row=0,column=0)

value.grid(row=0,column=1)

    apply.grid(row=0,column=2)

    gui.mainloop()
```



B)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).

```
class Employee:

    def __init__(self, id, name, department, salary):

        self.id=id

        self.name=name

        self.department=department
```

```
        self.salary=salary



class manager(Employee):

    def __init__(self, id, name, department, salary ,bonus):

        super(manager, self).__init__(id, name, department, salary)

        self.bonus=bonus



    def totalsalary(self):

        print(self.name,'got total salary :',self.salary+self.bonus)



n=manager('A023','ABC','GENERAL MANAGEMENT',20000,8000)

m=manager('A025','pQR','MARKETIG',25000,6400)

n.totalsalary()

m.totalsalary()
```

ABC got total salary : 28000
pQR got total salary : 31400

Pythone-Slip 5

A) 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.

```
class Str1():

    def __init__(self,demo=0):

        self.demo=demo



    def set_String(self,demo):

        self.demo=demo



    def print_streing(self):

        str=self.demo
```

```
        print(str.upper())



    A=Str1()

    rowinput=input('enter a string :')

    A.set_String(rowinput)

    A.print_streing()
```

```
enter a string :hi i am abc
HI I AM ABC
```

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

```
def generator(r):

    a=0;b=1

    for i in range (1,r):

        print(b)

        a,b=b,a+b
```

```
n=int(input('Enter a number :'))

if n==0:

    print('0')

else:

    print(0)

    generator(n)
```

```
Enter a number :5
0
1
1
2
3
```

Pythone-Slip 6

A)Write python script using package to calculate area and volume of cube and sphere

```
import math
```

```
class cube():
    def __init__(self,edge):
        self.edge=edge

    def cube_area(self):
        cubearea=6*self.edge*self.edge
        print("Area of cube :",cubearea)

    def cube_volume(self):
        cubevolume=self.edge*self.edge*self.edge
        print("Volume of cube :",cubevolume)

class sphere():
    def __init__(self,radius):
        self.radius=radius

    def sphere_area(self):
        spherearea=4*math.pi*self.radius*self.radius
        print("Area of sphere :",spherearea)

    def sphere_volume(self):
        spherevolume=float(4/3*math.pi*self.radius**3)
        print("volume of sphere :",spherevolume)
```

```
e1=cube(5)
e1.cube_area()
e1.cube_volume()

r1=sphere(5)
r1.sphere_area()
r1.sphere_volume()
```

```
Area of cube : 150
Volume of cube : 125
Area of sphere : 314.1592653589793
volume of sphere : 523.5987755982989
```

B) 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.

```
import tkinter as tk

import tkinter.font as tkFont




class App:

    def __init__(self):
```

```
root=tk.Tk()

# create a custom font

self.customFont = tkFont.Font(family="Helvetica", size=12)



# create a couple widgets that use that font

buttonframe = tk.Frame()

label = tk.Label(root, text="Hello, world", font=self.customFont)

text = tk.Text(root, width=20, height=2, font=self.customFont)

buttonframe.pack(side="top", fill="x")

label.pack()

text.pack()

text.insert("end","press +/- buttons to change\nfont size")



# create buttons to adjust the font
```

```
    bigger = tk.Button(root, text="+", command=self.OnBigger)
    smaller = tk.Button(root, text="-", command=self.OnSmaller)
    bigger.pack(in_=buttonframe, side="left")
    smaller.pack(in_=buttonframe, side="left")


    root.mainloop()


def OnBigger(self):
    '''Make the font 2 points bigger'''
    size = self.customFont['size']
    self.customFont.configure(size=size+2)


def OnSmaller(self):
    '''Make the font 2 points smaller'''
```

```
        size = self.customFont['size']

        self.customFont.configure(size=size-2)



app=App()
```

Pythone-Slip 7

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

```
    class Complex ():

        def initComplex(self):

            self.realPart = int(input("Enter the Real
    Part: "))

            self.imgPart = int(input("Enter the
    Imaginary Part: "))



        def display(self):

            print(self.realPart,"+",self.imgPart,"i",
    sep="")
```

```
    def sum(self, c1, c2):
        self.realPart = c1.realPart + c2.realPart
        self.imgPart = c1.imgPart + c2.imgPart

c1 = Complex()
c2 = Complex()
c3 = Complex()

print("Enter first complex number")
c1.initComplex()
print("First Complex Number: ", end="")
c1.display()

print("Enter second complex number")
c2.initComplex()
print("Second Complex Number: ", end="")
c2.display()

print("Sum of two complex numbers is ",
end="")
```

```
c3.sum(c1,c2)

c3.display()
```

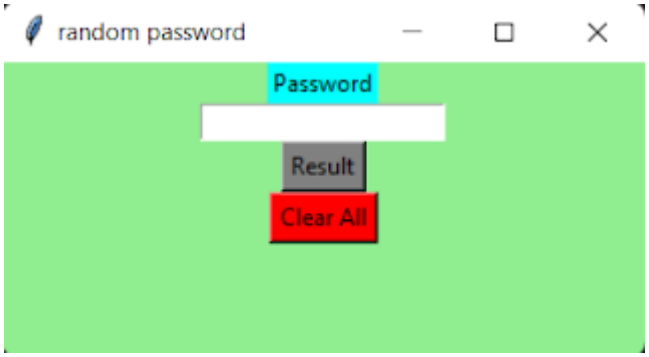
```
Enter first complex number
Enter the Real Part: 5
Enter the Imaginary Part: 2
First Complex Number: 5+2i
Enter second complex number
Enter the Real Part: 5
Enter the Imaginary Part: 3
Second Complex Number: 5+3i
Sum of two complex numbers is 10+5i
```

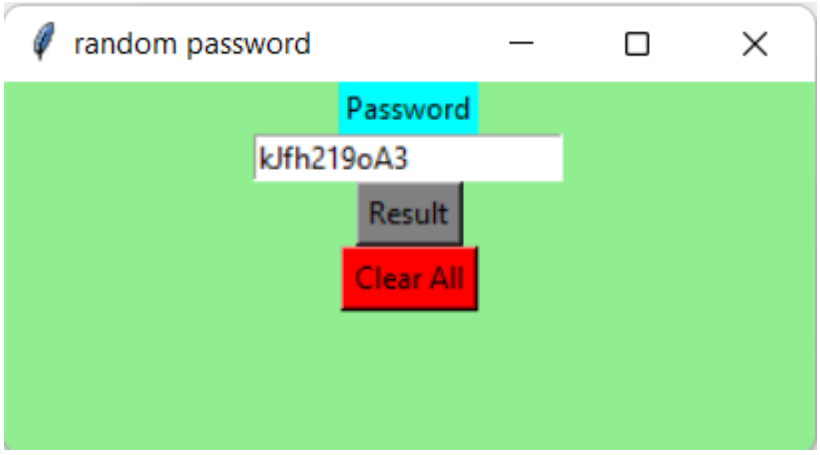
B) **Write python GUI program to generate a random password with upper and lower case letters.**

```
import string,random

from tkinter import *



def password():

    clearAll()

    String = random.sample(string.ascii_letters, 6)
+ random.sample(string.digits, 4)

    random.SystemRandom().shuffle(String)

    password=''.join(String)

    passField.insert(10, str(password))
```

```
def clearAll() :

    passField.delete(0, END)


if __name__ == "__main__" :


    gui = Tk()

    gui.configure(background = "light green")

    gui.title("random password")

    gui.geometry("325x150")


    passin = Label(gui, text = "Password", bg =
"#00ffff").pack()

    passField = Entry(gui);passField.pack()


    result = Button(gui,text = "Result",fg = "Black",

                bg = "gray", command =
password).pack()

    clearAllEntry = Button(gui,text = "Clear All",fg =
"Black",

                bg = "Red", command =
clearAll).pack()
```

gui.mainloop()





Python-Slip 8

A)-Write a python script to find the repeated items of a tuple

import collections

```
tuplex = 2,4,5,6,2,3,4,4,7,5,6,7,1
dictx=collections.defaultdict(int)
for x in tuplex:
    dictx[x]+=1
for x in sorted(dictx,key=dictx.get):
    if dictx[x]>1:
        print('%d repeted %d times'%(x,dictx[x]))
```

```
2 repeted 2 times
5 repeted 2 times
6 repeted 2 times
7 repeted 2 times
4 repeted 3 times
```

B)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.

```
class Str1():
    def __init__(self,demo=0):
        self.demo=demo

    def set_String(self,demo):
        demo=demo.upper()
        self.demo=demo

    def print_streing(self):

        return self.demo

A=Str1()
```

```
    str1=input('enter a string to display :')
    A.set_String(str1)
        print('Upper string :',A.print_streing())
```

```
enter a string to display :abcdefg
Upper string : ABCDEFG
```

Python-Slip9-

A)Write a Python script using class to reverse a string word by word

```
def reverse_string(str):
    str1 = ""   # Declaring empty string to store the reversed string
    for i in str:
        str1 = i + str1
    return str1    # It will return the reverse string to the caller function

str = "python"    # Given String
print("The original string is: ",str)
print("The reverse string is",reverse_string(str)) # Function call
```

```
The original string is:  python
The reverse string is nohtyp
```

B)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.

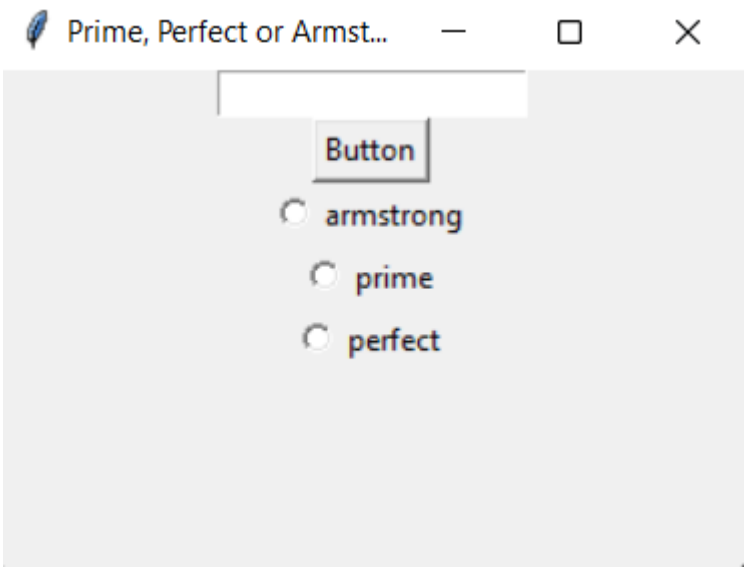
```
from tkinter import*

def perfect():

    number=int(numberFeald.get())

    count = 0

    for i in range(1, number):

        if number % i == 0:

            count = count + i

    if count == number:

        perfect1.select()

        print(number, 'The number is a Perfect number!')

    else:

        perfect1.deselect()

        print(number, 'The number is not a Perfect number!')
```

```
def armstrong():
    number=int(numberFeald.get())
    count = 0
    temp = number
    while temp > 0:
        digit = temp % 10
        count += digit ** 3
        temp //= 10
    if number == count:
        armstrong1.select()
        print(number, 'is an Armstrong number')
    else:
        armstrong1.deselect()
        print(number, 'is not an Armstrong number')
```

```
def prime():
    number=int(numberFeald.get())
    if number > 1:
        for i in range(2,number):
            if (number % i) == 0:
                prime1.deselect()
                print(number,"is not a prime number")
                print(i,"times",number//i,"is",number)
                break
            else:
                prime1.select()
                print(number,"is a prime number")
    else:
```

```
        prime1.deselect()

        print(number,"is not a prime number")



root=Tk()

root.title('Prime, Perfect or Armstrong number')

root.geometry('300x200')

numberFeald=Entry(root)

numberFeald.pack()



Button1=Button(root,text='Button',command=lambda:[armstrong()
,prime(),perfect()])

Button1.pack()



prime2=IntVar()
```

```
perfect2=IntVar()

armstrong2=IntVar()



armstrong1=Radiobutton(root,text='armstrong',variable=armstron
g2,value=1)

prime1=Radiobutton(root,text='prime',variable=prime2,value=1)

perfect1=Radiobutton(root,text='perfect',variable=perfect2,value=
1)

armstrong1.pack()

prime1.pack()

perfect1.pack()



root.mainloop()
```



Python-Slip 10

A) Write Python GUI program to display an alert message when a button is pressed. import tkinter.messagebox

```
import tkinter.messagebox

def onClick():

    tkinter.messagebox.showinfo("Title goes here","Message goes here")

root = tkinter.Tk()

button = tkinter.Button(root,text = "Click Me", command = onClick)

button.pack()

root.mainloop()
```

B)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.

```
class validity:


    def f(str):


        a= ['()', '{}', '[]']


        while any(i in str for i in a):


            for j in a:


                str = str.replace(j, '')
```

```
        return not str



s = input("enter : ")



print(s, "-", "is balanced"



     if validity.f(s) else "is Unbalanced")
```

```
enter : ()
() - is balanced
```

Python-Slip 11

A-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)

```
x = (1,2,3,4)

y = (3,5,2,1)
```

```
z = (2,2,3,1)

print("Original lists:")

print(x)

print(y)

print(z)

print("\nElement-wise sum of the said tuples:")

result = tuple(map(sum, zip(x, y, z)))

print(result)
```

```
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)
```

B) 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

```
# Import the library tkinter

from tkinter import *



# Create a GUI app
```

```
app = Tk()

# Set the title and geometry to your app
app.title("Geeks For Geeks")
app.geometry("800x500")

# Create menubar by setting the color
menubar = Menu(app, background='blue', fg='white')

# Declare file and edit for showing in menubar
file = Menu(menubar, tearoff=False, background='yellow')
edit = Menu(menubar, tearoff=False, background='pink')

# Add commands in in file menu
file.add_command(label="New")
file.add_command(label="Exit", command=app.quit)

# Add commands in edit menu
edit.add_command(label="Cut")
```

```
edit.add_command(label="Copy")

edit.add_command(label="Paste")


# Display the file and edit declared in previous step

menubar.add_cascade(label="File", menu=file)

menubar.add_cascade(label="Edit", menu=edit)


# Displaying of menubar in the app

app.config(menu=menubar)


# Make infinite loop for displaying app on screen

app.mainloop()
```

Python-Slip 12

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

```
import tkinter as tk

parent = tk.Tk()

parent.title("-Welcome to Python tkinter Basic exercises-")
```

```
my_label = tk.Label(parent, text="Hello", font=("Arial Bold", 70))
my_label.grid(column=0, row=0)
parent.mainloop()
```

Copy

Sample Output:

-B-Write a python program to count repeated characters in a string.
Sample string: 'thequickbrownfoxjumpsoverthelazydog'

```
import collections



str1 = input('Enter a string : ')



d = collections.defaultdict(int)
```

```
for c in str1:
    if c ==' ':
        continue
    d[c] += 1

for c in sorted(d, key=d.get):
    if d[c] > 1:
        print('%s - %d' % (c, d[c]))
```

```
Enter a string : gdhdggdgs
d - 3
g - 4
```

Python -Slip 13

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

```
try:
```

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

except ValueError:

    print("\nThis is not a number!")

else:

    print('\nnumber is : ',num)
```

Enter a number :dc

This is not a number!

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

```
 q=[]
q.append(10)
q.append(100)
q.append(1000)
q.append(10000)
print("Initial Queue is:",q)
print(q.pop(0))
print(q.pop(0))
print(q.pop(0))
print("After Removing elements:",q)
```

```
Initial Queue is: [10, 100, 1000, 10000]
10
100
1000
After Removing elements: [10000]
```

Python-Slip 14

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

```
from tkinter import *

from math import pi

from tkinter import messagebox



def clearAll() :

   RadiusField.delete(0, END)

   HeightField.delete(0, END)

   volumeField.delete(0, END)
```

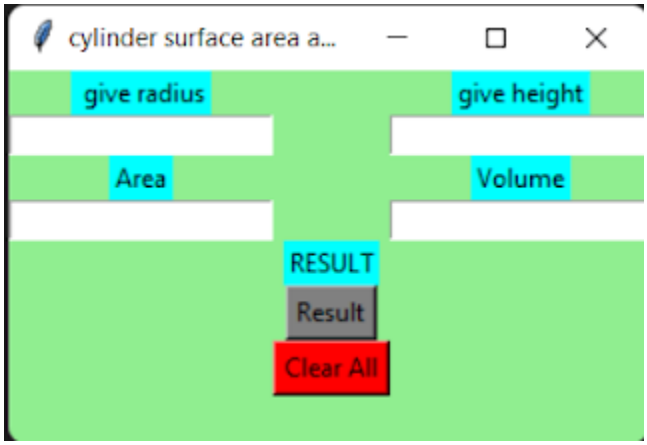
```
    areaField.delete(0,END)



def checkError() :

    if (RadiusField.get() == "" or HeightField.get() == "") :

        messagebox.showerror("Input Error")

        clearAll()

        return -1



def result() :

    value = checkError()

    if value == -1 :

        return

    else :
```

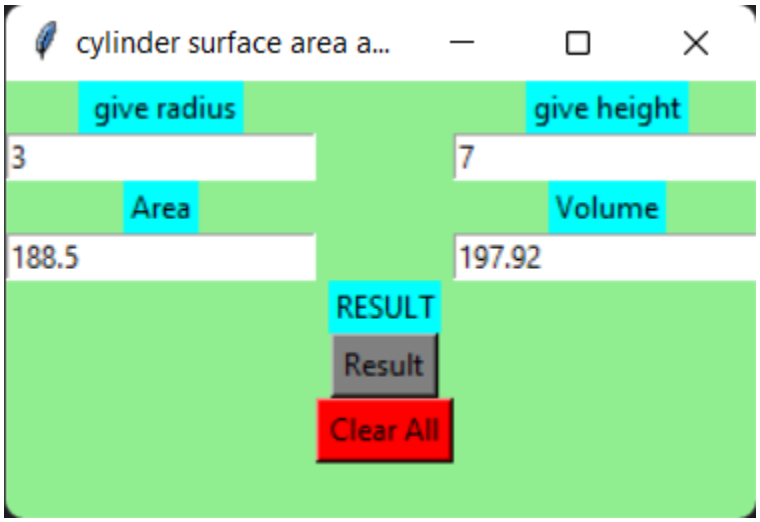
```
        Radius = int(RadiusField.get())

        Height = int(HeightField.get())



        volume=round(pi*Height*Radius**2,2)

        area=round((2*pi*Radius*Height)+(2*pi*Radius**2),2)



        volumeField.insert(10, str(volume))

        areaField.insert(10, str(area))



if __name__ == "__main__" :



    gui = Tk()

    gui.configure(background = "light green")
```

```
gui.title("cylinder surface area and volume of cylinder")

gui.geometry("300x175")



radius = Label(gui, text = " give radius", bg = "#00ffff")

height = Label(gui, text = "give height", bg = "#00ffff")

area = Label(gui, text = "Area", bg = "#00ffff")

volume = Label(gui, text = "Volume", bg = "#00ffff")



resultlabel = Label(gui, text = "RESULT", bg = "#00ffff")



resultbutton = Button(gui, text = "Result", fg = "Black",
```

```
                    bg = "gray", command = result)

    clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

                    bg = "Red", command = clearAll)



    RadiusField = Entry(gui)

    HeightField = Entry(gui)

    volumeField = Entry(gui)

areaField =Entry(gui)



radius.grid(row = 0, column = 0)

height.grid(row = 0, column = 2)

area.grid(row = 2, column = 0)

volume.grid(row = 2, column = 2)
```

```
resultlabel.grid(row = 4, column = 1)

resultbutton.grid(row = 5, column = 1)



RadiusField.grid(row = 1, column = 0)

HeightField.grid(row = 1, column = 2)



areaField.grid(row=3,column=0)

volumeField.grid(row = 3, column = 2)

clearAllEntry.grid(row = 6, column = 1)



    gui.mainloop()
```





B) **Write a Python program to display plain text and cipher text using a Caesar encryption.**

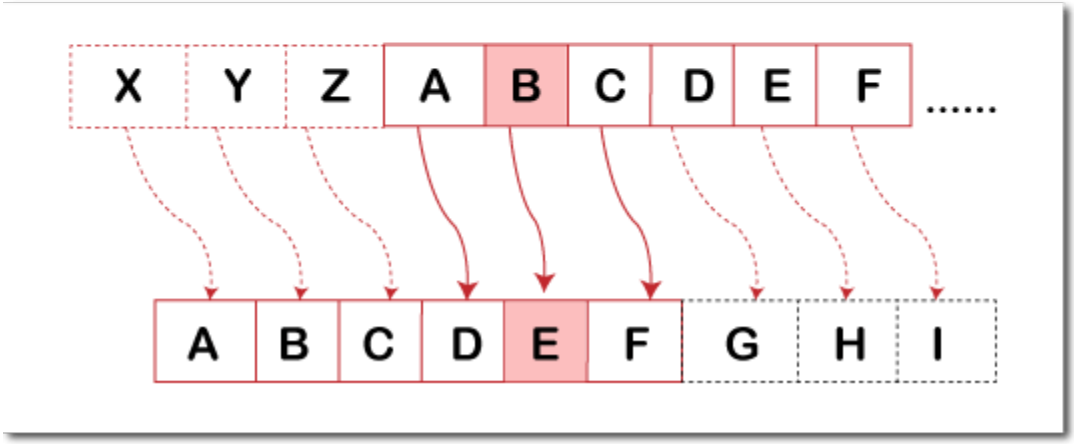
The same function can be used for decryption. Instead, we will modify the shift value such that shifts = **26 - shift.**

$$E_n(x) = (x + n) \bmod 26$$

(Encryption Phase with shift n)

$$D_n(x) = (x + n) \bmod 26$$

(Decryption Phase with shift n)



Let's understand the following example -

Example -

```
def encypt_func(txt, s):

    result = ""

    for i in range(len(txt)):

        char = txt[i]

        if (char.isupper()):
```

```
            result += chr((ord(char) + s - 64) % 26 + 65)

        else:

            result += chr((ord(char) + s - 96) % 26 + 97)

    return result



txt = input('Enter a string : ')

s = int(input('ENtER number to shift pattern encript : '))



print("Plain txt : " + txt)

print("Shift pattern : " + str(s))

print("Cipher: " + encypt_func(txt, s))
```

```
Enter a string : hi good
ENtER number to shift pattern encript : 8
Plain txt : hi good
Shift pattern : 8
Cipher: qrwpxxm
```

Python-Slip 15

A) **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.**

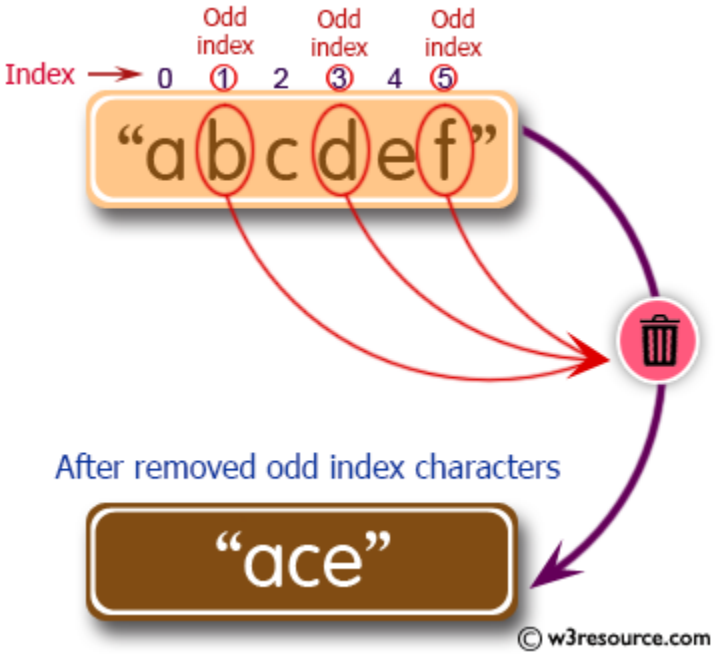
```
class Student:

    def __init__(self, Student_name, marks):

        self.Student_name=Student_name

        self.marks=marks



    def get_marks(self):

        print("\nOriginal name and values")

        print(self.Student_name,'marks : ',self.marks)



    def modify_marks(self):

        self.Student_name1=input('Enter modifyed name : ')
```

```
        self.marks1=int(input('Enter modifyed marks : '))

        print(self.Student_name1,'modifyed
marks',self.marks)



    def modifyed_marks(self):

        print("\nmodified name and values")

            print(self.Student_name1,'marks : ',self.marks1)



    x=Student('Abc',81)

x.get_marks()

x.modify_marks()



x.get_marks()
```

x.modifyed_marks()

```
Original name and values
Abc marks :  81
Enter modifyed name : pqr
Enter modifyed marks : 87
pqr modifyed marks 81

Original name and values
Abc marks :  81

modified name and values
pqr marks :  87
```

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



Sample Solution:-

Python Code:

```
def modify(string):
```

```
    final = ""
    for i in range(len(string)):
      if i % 2 == 0:
        final = final + string[i]
    return final
string=input("Enter string:")
print("Modified string is:")
    print(modify(string))
```

Sample Output:

```
Enter string:abcdefg
Modified string is:
aceg
```

Python-Slip 16

A) **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.**

```
class Ractangle():
  def __init__(self,l,w):
    self.l=l
    self.w=w
  def rectangle_area(self):
```

```
        return self.l*self.w
    def rectangle_Perimeter(self):
        return (self.l*2)+(self.w*2)


L=int(input('enter Length  of Rectangle :'))
W=int(input('enter Width  of Rectangle :'))
a1=Ractangle(L,W)
print(a1.rectangle_area())
print(a1.rectangle_Perimeter())
```

```
enter Length  of Rectangle :2
enter Width  of Rectangle :3
6
10
```

B) **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.**

```
# Import Module

from tkinter import *
```

```
# Function will remove selected Listbox items
def remove_item():
    selected_checkboxs = listbox.curselection()

    for selected_checkbox in selected_checkboxs[::-1]:
        listbox.delete(selected_checkbox)

# Create Object
root = Tk()

# Set Geometry
root.geometry("200x200")

# Add Listbox
listbox = Listbox(root, selectmode=MULTIPLE)
listbox.pack()

# Listbox Items List
items = ["Apple", "Orange", "Grapes", "Banana", "Mango"]
```

```
# Iterate Through Items list
for item in items:
    listbox.insert(END, item)

Button(root, text="delete", command=remove_item).pack()

# Execute Tkinter
root.mainloop()
```

Python-Slip 17

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

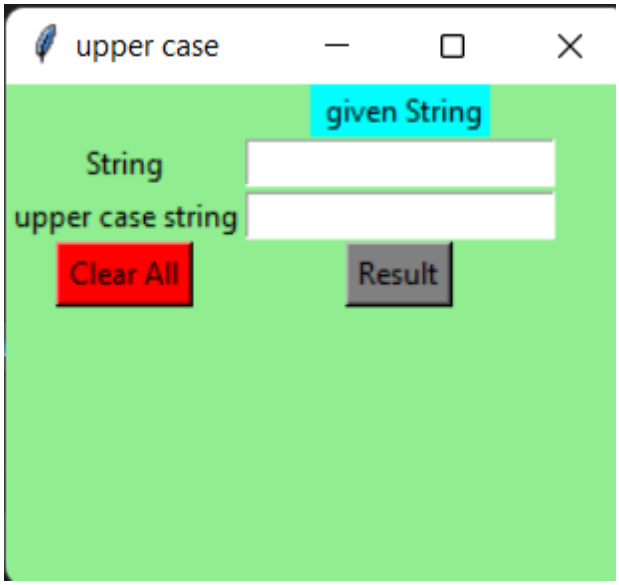
```
from tkinter import *
from tkinter import messagebox
```

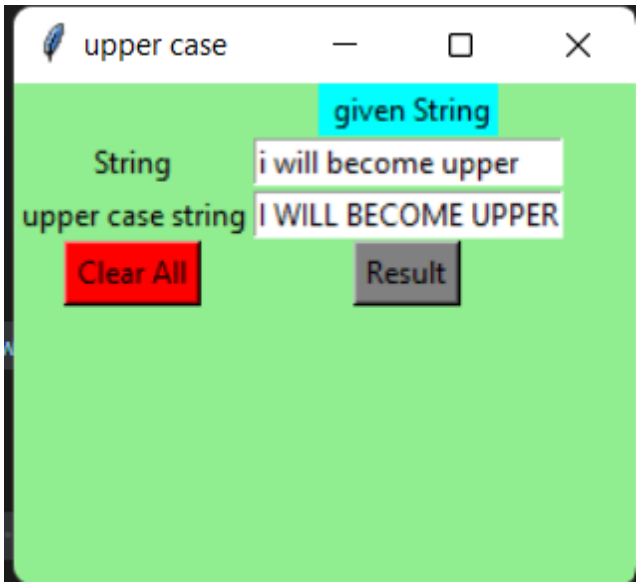
```
def clearAll() :
    str1Field.delete(0, END)
    altersField.delete(0, END)


def checkError() :
    if (str1Field.get() == "" ) :
        messagebox.showerror("Input Error")
        clearAll()
        return -1


def upper() :
    value = checkError()
    if value == -1 :
        return
```

```
    else :

        String0 = (str1Field.get())



        newstr=String0.upper()



        altersField.insert(20, str(newstr))



if __name__ == "__main__" :

    gui = Tk()

    gui.configure(background = "light green")

    gui.title("upper case")

    gui.geometry("250x200")
```

```
Stringin = Label(gui, text = " given String", bg = "#00ffff")

str1 = Label(gui, text = "String", bg = "light green")

str1Field = Entry(gui)



result = Button(gui, text = "Result", fg = "Black",

        bg = "gray", command = upper)



alters = Label(gui, text = "upper case string", bg = "light green")

altersField = Entry(gui)



clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

        bg = "Red", command = clearAll)
```

```
Stringin.grid(row = 0, column = 1)

str1.grid(row = 1, column = 0)

str1Field.grid(row = 1, column = 1)




alters.grid(row = 2, column = 0)

altersField.grid(row = 2, column = 1)

clearAllEntry.grid(row = 3, column = 0)

result.grid(row = 3, column = 1)

    gui.mainloop()
```





B) 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.

Python-Slip 18

A) 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

```
list1 =[]

list3=[]

list2=[]

n=int(input('Enter number of elements : '))

for i in range(n):

    value=int(input())

    list1.append(value)

print(list1)

n=int(input('Enter a number to sort list : '))




for i in list1:

    if n > i:
```

```
            list2.append(i)

        else:

            list3.append(i)

    print('less then {} value list : '.format(n),list2)

        print('greater then {} value list :'.format(n),list3)
```

```
Enter number of elements : 11
1
2
3
4
5
5
6
4
8
9
7
[1, 2, 3, 4, 5, 5, 6, 4, 8, 9, 7]
Enter a number to sort list : 5
less then 5 value list :  [1, 2, 3, 4, 4]
greater then 5 value list : [5, 5, 6, 8, 9, 7]
```

B) **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.**

```
class person:

    def __init__(self,name,address):

        self.empname=name

        self.address=address



    def display(self):

        print('name : {}\taddress : {}\tsalary : {}'.format(self.empname,

        self.address,a.getsalary()))
```

```
class employee(person):

    def __init__(self, name, address,salary):

        super().__init__(name, address)

        self.salary=salary



    def getsalary(self):

        return self.salary

name1=input('enter name : ')

address=input('enter address : ')

salary=int(input('enter salary : '))

a=employee(name1,address,salary)

a.display()
```

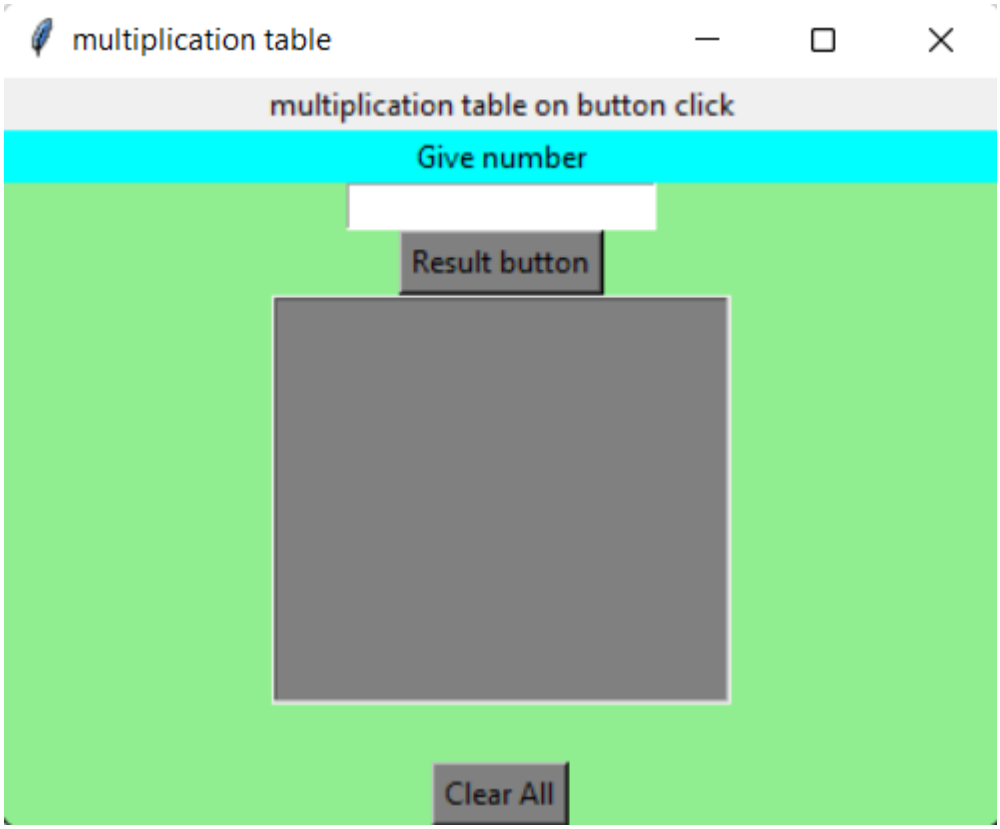
```
enter name : abc
enter address : saswad
enter salary : 100000
name : abc              address : saswad    salary : 100000
```

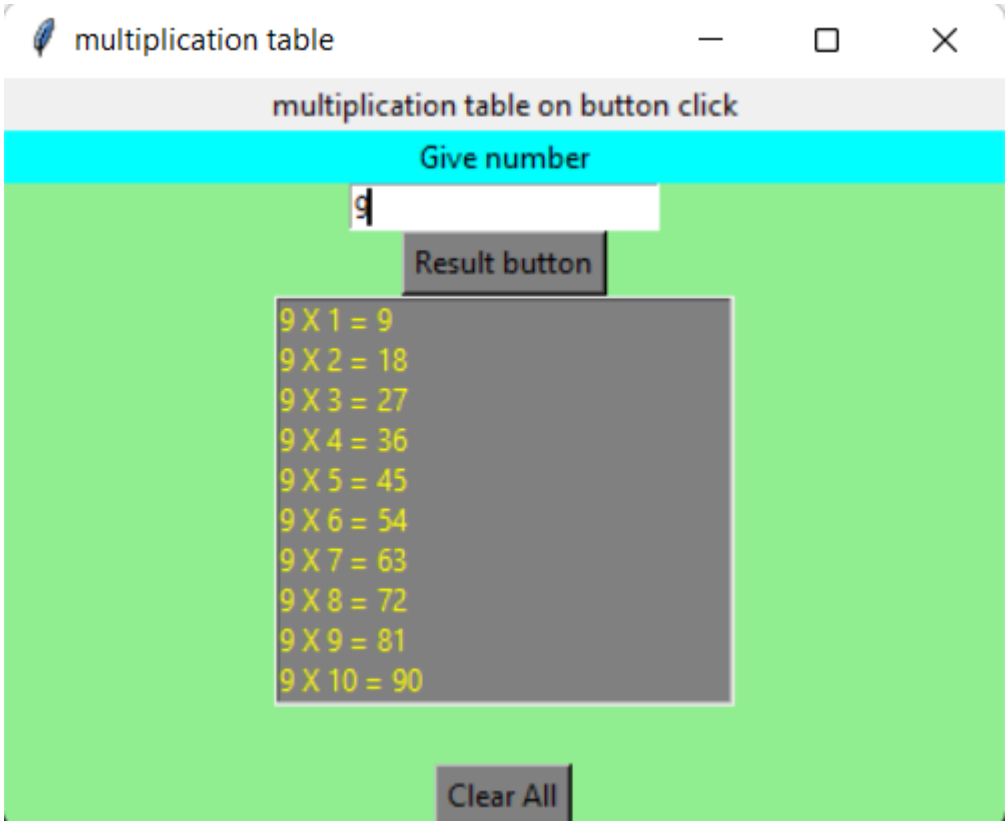
Python-Slip 19

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

```
from tkinter import *


def clearAll() :
    numberField.delete(0, END);Lb1.delete(0,END)
def multiplication():
    num = int(numberField.get())
    Lb1.insert(0, '{} X 1 = {}'.format(num,1*num))
    Lb1.insert(1, '{} X 2 = {}'.format(num,2*num))
    Lb1.insert(2, '{} X 3 = {}'.format(num,3*num))
    Lb1.insert(3, '{} X 4 = {}'.format(num,4*num))
    Lb1.insert(4, '{} X 5 = {}'.format(num,5*num))
    Lb1.insert(5, '{} X 6 = {}'.format(num,6*num))
    Lb1.insert(6, '{} X 7 = {}'.format(num,7*num))
    Lb1.insert(7, '{} X 8 = {}'.format(num,8*num))
    Lb1.insert(8, '{} X 9 = {}'.format(num,9*num))
    Lb1.insert(9,'{} X 10 = {}'.format(num,10*num))
if __name__=="__main__" :
    gui = Tk()
    gui.configure(background = "light green")
    gui.title("multiplication table")
```

```
    gui.geometry("400x300")
    label=Label(gui,text='multiplication table \
            on button click').pack(side=TOP,fill=BOTH)
    number = Label(gui, text = "Give number", bg =
"#00ffff").pack(fill=BOTH)
    numberField = Entry(gui)
    numberField.pack()
    resultbutton = Button(gui, text = "Result button",
            fg = "Black", bg = "gray",command=multiplication).pack()


Lb1 =
Listbox(gui,fg='yellow',width=30,bg='gray',bd=1,activestyle='dotbox')


clearAllEntry = Button(gui, text = "Clear All",
        fg = "Black", bg = "gray", command =
clearAll).pack(side=BOTTOM)


Lb1.pack()
gui.mainloop()
```





B)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.

```
class Shape(object):
    def __init__(self):
        pass

    def area(self):
        return 0

class Square(Shape):
    def __init__(self, l):
        Shape.__init__(self)
        self.length = l

    def area(self):
        return self.length*self.length

aSquare= Square(3)
    print (aSquare.area())
```

Python-Slip 20-

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

```
from math import pi

class Circle():

    def __init__(self,Radius):

        self.Radius=Radius

    def area(self):

        a=pi*self.Radius*self.Radius

        return round(a,2)

    def circumference(self):

        c=2*self.Radius*pi

        return round(c,2)
```

```
r= int(input('enter radius of circle : '))

a=Circle(r)

print('Area of circle is : ',a.area())

print('Circumference of circle is : ',a.circumference())
```

```
enter radius of circle : 5
Area of circle is :  78.54
Circumference of circle is :  31.42
```

B-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}

Expected Output:
{1:1, 2:4, 3:9, 4:16, 5:25}

```
n=int(input("Input a number : "))

d ={}

for x in range(1,n+1):

    d[x]=x*x

print(d)
```

```
Input a number : 5
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

Python-Slip 21

A) **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.**

```
class Ractangle():
    def __init__(self,l,w):
        self.l=l
        self.w=w
    def rectangle_area(self):
        return self.l*self.w
    def rectangle_Perimeter(self):
        return (self.l*2)+(self.w*2)

L=int(input('enter Length  of Rectangle :'))
W=int(input('enter Width  of Rectangle :'))
a1=Ractangle(L,W)

print(a1.rectangle_area())
    print(a1.rectangle_Perimeter())
```

```
enter Length  of Rectangle :5
enter Width  of Rectangle :6
30
22
```

B) **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))**

```
tup1=(('333','33'),('1416','55'))
```

```
a,b=tup1

newlista=[]

newlistb=[]

arr=[]

for i in a:

    newlista.append(int(i))

arr.append(newlista)

for i in b:

    newlistb.append(int(i))

arr.append(newlistb)

finaltup=tuple(map(tuple, arr))

print(finaltup)
```

((333, 33), (1416, 55))

Python-Slip 22

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

```
class One:
    def __init__(self, a):
        self.a = a
    def __mult__(self, object2):
        return self.a + object2.a
class Two:
    def __init__(self, a):
        self.a = a
    def __mult__(self, object2):
        return self.a + object2.a
a_instance = One(10)
b_instance = Two(20)
print(a_instance + b_instance)
```

B) **Write a python script to implement bubble sort using list**

```
def bubble_sort(list1):
    for i in range(0,len(list1)-1):
        for j in range(len(list1)-1):
            if(list1[j]>list1[j+1]):
                temp = list1[j]
```

```
            list1[j] = list1[j+1]

            list1[j+1] = temp

    return list1



list1 =[]

n=int(input('Enter number of elements : '))

for i in range(n):

    value=int(input())

    list1.append(value)

print("The unsorted list is: ", list1)



print("The sorted list is: ", bubble_sort(list1))
```

```
Enter number of elements : 5
4
6
39
9
5
The unsorted list is:  [4, 6, 39, 9, 5]
The sorted list is:  [4, 5, 6, 9, 39]
```

Python-Slip 23

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

```
import tkinter as tk

parent = tk.Tk()

parent.title("-Welcome to Python tkinter Basic exercises-")

my_label = tk.Label(parent, text="Hello", font=("Arial Bold", 70))

my_label.grid(column=0, row=0)

parent.mainloop()
```

Copy

Sample Output:

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

```
import math

class Circle:

    def __init__(self, radius):
        self.radius = radius

    def get_result(self):
        return self.radius

    def area(self):
        return math.pi * self.radius ** 2

    def __add__(self, another_circle):
        return Circle(self.radius + another_circle.radius)

    def __sub__(self, another_circle):
        return Circle(self.radius - another_circle.radius)

    def __mul__(self, another_circle):
        return Circle(self.radius * another_circle.radius)

    def __gt__(self, another_circle):
        return Circle(self.radius > another_circle.radius)

    def __lt__(self, another_circle):
        return Circle(self.radius < another_circle.radius)

    def __ge__(self, another_circle):
        return Circle(self.radius >= another_circle.radius)

    def __le__(self, another_circle):
        return Circle(self.radius <= another_circle.radius)

    def __eq__(self, another_circle):
        return Circle(self.radius == another_circle.radius)

    def __ne__(self, another_circle):
        return Circle(self.radius != another_circle.radius)
```

```
c1 = Circle(10)
print(c1.get_result())
print(c1.area())

c2 = Circle(15)
print(c2.get_result())
print(c1.area())

c3 = c1 + c2
print(c3.get_result())

c3 = c2 - c1
print(c3.get_result())

c4 = c1 * c2
print(c4.get_result())

c5 = c1 < c2
print(c5.get_result())

c5 = c2 < c1
print(c5.get_result())
```

Python-Slip 24

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

```
class factorial():


    def __init__(self,num):
```

```
    self.num=num

    factnum=1

    if self.num<0:

        print("Sorry, factorial does not exist for negative
numbers")

    elif self.num==0:

        print("The factorial of 0 is 1")

    else:

        for i in range(1,num+1):

            factnum=factnum*i

        print('The factorial of {} is {}
factorial'.format(num,factnum))



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

```
if num > 1:
    for i in range(2,num):
        if (num % i) == 0:
            print(num,"is not a prime number")
            print(i,"times",num//i,"is",num)
            break
    else:
        print(num,"is a prime number")
else:
    print(num,"is not a prime number")


    factorial(num)
```

```
Enter a number: 5
5 is a prime number
The factorial of 5 is 120 factorial
```

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

```
from tkinter import END, Button, Entry, Label, Tk




def printWord(N):

    i = 0

    length = len(N)

    while i < length:

        printValue(N[i])

        i += 1



def printValue(digit):
```

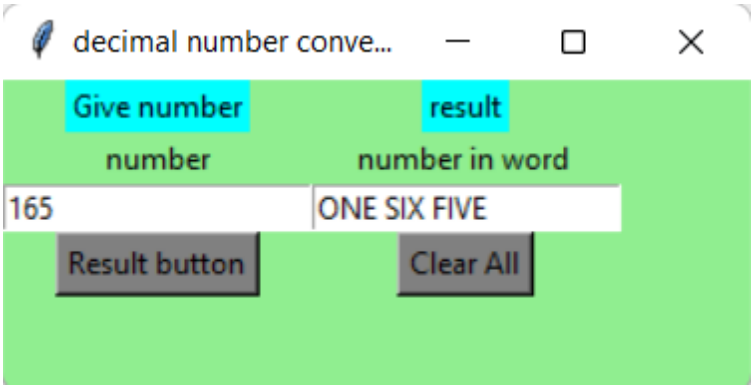
```
if digit == '0':
    wordField.insert(30,'ZERO ')
elif digit == '1':
    wordField.insert(30,'ONE ')
elif digit == '2':
    wordField.insert(30,'TWO ')
elif digit=='3':
    wordField.insert(30,'THREE ')
elif digit == '4':
    wordField.insert(30,'FOUR ')
elif digit == '5':
    wordField.insert(30,'FIVE ')
elif digit == '6':
```

```
        wordField.insert(30,'SIX ')

    elif digit == '7':

        wordField.insert(30,'SEVEN ')

    elif digit == '8':

        wordField.insert(30,'EIGHT ')

    elif digit == '9':

        wordField.insert(30,'NINE ')



def clearAll() :

    numberField.delete(0, END)

    wordField.delete(0, END)
```

```
def wordconvert():


    number0 = numberField.get()

    printWord(number0)



if __name__=="__main__" :

    gui = Tk()

    gui.configure(background = "light green")

    gui.title("decimal number converter")

    gui.geometry("300x125")

    number = Label(gui, text = "Give number", bg = "#00ffff")

    number1 = Label(gui, text = "number", bg = "light green")

    numberField = Entry(gui)
```

```
result = Label(gui, text = "result", bg = "#00ffff")

resultbutton = Button(gui, text = "Result button",fg = "Black",

bg = "gray", command = wordconvert)

numberinword = Label(gui, text ="number in word",bg ="light
 green")

wordField = Entry(gui)

clearAllEntry = Button(gui, text = "Clear All", fg ="Black",

bg = "gray", command = clearAll)



number.grid(row = 0, column = 1)

number1.grid(row = 1, column = 1)

numberField.grid(row = 2, column = 1)

resultbutton.grid(row = 3, column = 1)
```

```
result.grid(row = 0, column = 5)

numberinword.grid(row = 1, column = 5)

wordField.grid(row = 2, column = 5)

clearAllEntry.grid(row = 3, column = 5)

gui.mainloop()
```



Pythone-Slip25

-A.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

```
string=input('enter a string : ')
```

```
up=low=ele=0
for x in string:
    if x.isupper():
        up+=1
    elif x.islower():
        low+=1
    else:
        ele+=1
print('No. of Upper case characters : ',up)
print('No. of Lower case characters : ',low)
print('other special symbols',ele)
```

```
enter a string : hi python
No. of Upper case characters :  0
No. of Lower case characters :  8
other special symbols 1
```

-B- **Write a Python script to Create a Class which Performs Basic Calculator Operation**

```
class Calculator:
    def __init__(self,num1,num2,operation):
        self.num1=num1
        self.num2=num2
        self.operation=operation
        if self.operation=='*':
            print('Multiplication of {} and {} is : '.format(num1,num2),
                self.num1*self.num2)
```

```
    elif self.operation=='/':
        print('division of {} and {} is : '.format(num1,num2),
            self.num1/self.num2)

    elif self.operation=='-':
        print('Subtraction of {} and {} is : '.format(num1,num2),
            self.num1/self.num2)

    elif self.operation=='+':
        print('Addition of {} and {} is : '.format(num1,num2),
            self.num1+self.num2)

num1=int(input('Enter 1st number : '))
operator1=input('ENTER A CALCULATOR OPERATOR FROM FOLLOWING
 : / , * , - ,  +\n')
num2=int(input('Enter 2st number : '))
Calculator(num1,num2,operator1)
```

```
Enter 1st number : 2
ENTER A CALCULATOR OPERATOR FROM FOLLOWING : / , * , - ,  +
*
Enter 2st number : 3
Multiplication of 2 and 3 is :  6
```

Python-Slip 26-

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

```
area_square=lambda x: x*x       #area of square is a^2
side=int(input('Enter a side value of square : '))
print(area_square(side))


area_rectangle=lambda x,y:x*y   #area of rectangle is l*w
Length=int(input('Enter a Length value of rectangle : '))
Width=int(input('Enter a Width value of rectangle : '))
print(area_rectangle(Length,Width))
```

```
Enter a side value of square : 5
25
Enter a Length value of rectangle : 6
Enter a Width value of rectangle : 5
30
```

B)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?.

```
from tkinter import *


from tkinter import messagebox
```

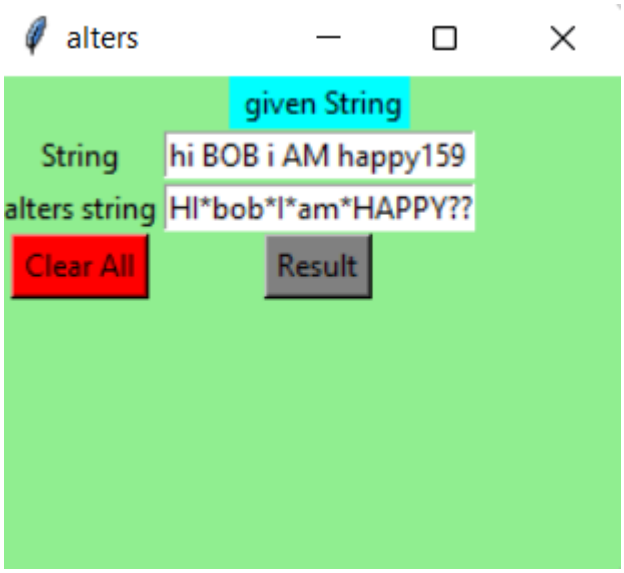
```
def clearAll() :
    str1Field.delete(0, END)
    altersField.delete(0, END)


def checkError() :
    if (str1Field.get() == "" ) :
        messagebox.showerror("Input Error")
        clearAll()
        return -1


def occurrences() :
    value = checkError()
    if value == -1 :
```

```
    return

else :

    String0 = (str1Field.get())



    newstr=''

    for char in String0:

        if char.isupper():

            char=char.lower()

            newstr+=char

        elif char.islower():

            char=char.upper()

            newstr+=char

        elif char==' ':

            char=char.replace(' ','*')
```

```
            newstr+=char
        elif char.isdigit():
            char=char.replace(char,'?')
            newstr+=char
        else:
            newstr+=char

    altersField.insert(10, str(newstr))

if __name__ == "__main__" :
    gui = Tk()
    gui.configure(background = "light green")
    gui.title("alters")
    gui.geometry("250x200")
```

```
Stringin = Label(gui, text = " given String", bg = "#00ffff")

str1 = Label(gui, text = "String", bg = "light green")

str1Field = Entry(gui)



result = Button(gui, text = "Result", fg = "Black",

bg = "gray", command = occurrences)



alters = Label(gui, text = "alters string", bg = "light green")

altersField = Entry(gui)



clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

bg = "Red", command = clearAll)
```

```
Stringin.grid(row = 0, column = 1)

str1.grid(row = 1, column = 0)

str1Field.grid(row = 1, column = 1)




alters.grid(row = 2, column = 0)

altersField.grid(row = 2, column = 1)

clearAllEntry.grid(row = 3, column = 0)

result.grid(row = 3, column = 1)

gui.mainloop()
```



Python-Slip27-

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

```
l = [(1,2), (3,4), (8,9)]
print(list(zip(*l)))
```

```
[(1, 3, 8), (2, 4, 9)]
```

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

```
from tkinter import *

from tkinter import messagebox
```

```
def clearAll() :

    numberField.delete(0, END)

    binaryField.delete(0, END)

    octalField.delete(0, END)

    hexadecimalField.delete(0, END)



def checkError() :

    if (numberField.get() == "") :

        messagebox.showerror("Input Error")

        clearAll()
```

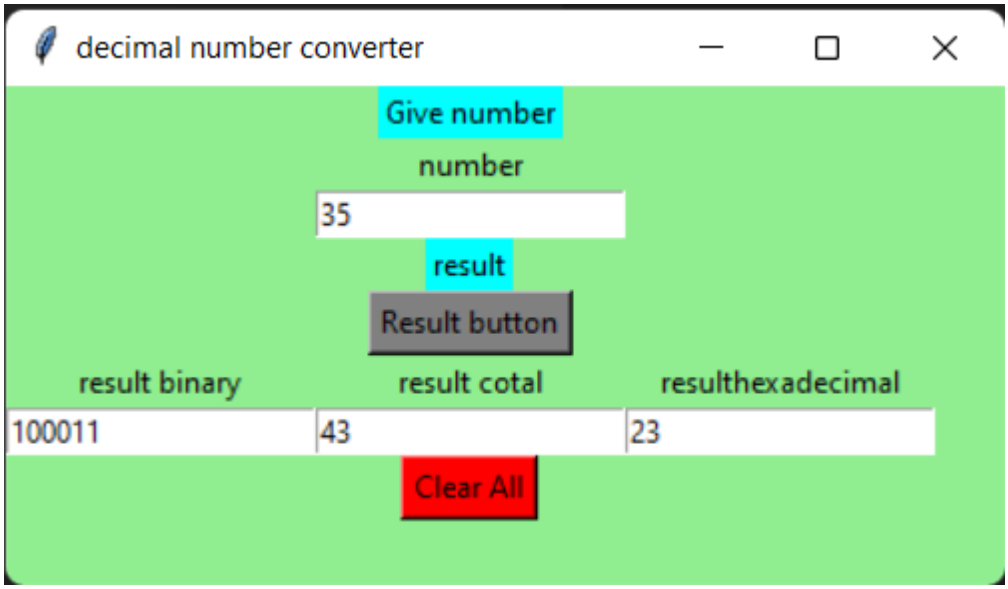
```
        return -1


def calculateAge() :


    value = checkError()


    if value == -1 :

        return


    else :


        number0 = int(numberField.get())

        binary=(bin(number0)[2:])

        octal =oct(number0)[2:]
```

```
        hexadecimal=hex(number0)[2:]



        binaryField.insert(10, str(binary))

        octalField.insert(10, str(octal))

        hexadecimalField.insert(10, str(hexadecimal))



if __name__ == "__main__" :



    gui = Tk()

    gui.configure(background = "light green")

    gui.title("decimal number converter")

    gui.geometry("400x200")
```

```
    number = Label(gui, text = "Give number", bg = "#00ffff")

    number1 = Label(gui, text = "number", bg = "light green")

    numberField = Entry(gui)



    result = Label(gui, text = "result", bg = "#00ffff")



    resultbutton = Button(gui, text = "Result button", fg = "Black",

    bg = "gray", command = calculateAge)



    resultbinary = Label(gui, text = "result binary", bg = "light green")

    resultoctal = Label(gui, text = "result cotal", bg = "light green")

    resulthexadecimal = Label(gui,text ="resulthexadecimal",bg = "light
green")
```

```
binaryField = Entry(gui)

octalField = Entry(gui)

hexadecimalField = Entry(gui)



clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

bg = "Red", command = clearAll)



number.grid(row = 0, column = 1)

number1.grid(row = 1, column = 1)

numberField.grid(row = 2, column = 1)



result.grid(row = 3, column = 1)

resultbutton.grid(row = 4, column = 1)
```

```
resultbinary.grid(row = 5, column = 0)

binaryField.grid(row = 6, column = 0)



resultoctal.grid(row = 5, column = 1)

octalField.grid(row = 6, column = 1)



resulthexadecimal.grid(row = 5, column = 2)

hexadecimalField.grid(row = 6, column = 2)



clearAllEntry.grid(row = 7, column = 1)



gui.mainloop()
```



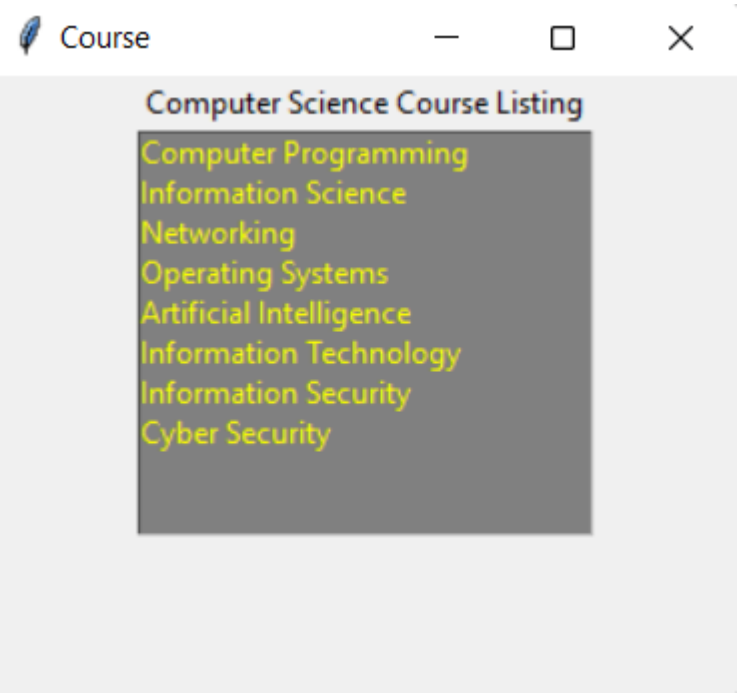
Python-Slip 28

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

```
from tkinter import *



top = Tk()

top.title('Course')
```

```
top.geometry("300x250")

Lb1 =
Listbox(top,fg='yellow',width=30,bg='gray',bd=1,activestyle='dotbox')

label=Label(top,text='Computer Science Course Listing').pack()

Lb1.insert(1, "Computer Programming")

Lb1.insert(2, "Information Science")

Lb1.insert(3, "Networking")

Lb1.insert(4, "Operating Systems")

Lb1.insert(5, "Artificial Intelligence")

Lb1.insert(6, "Information Technology")

Lb1.insert(7,'Information Security')

Lb1.insert(8, "Cyber Security")



Lb1.pack()
```

top.mainloop()



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

```
list1 =[]

n=int(input('Enter number of elements in first list : '))

for i in range(n):

    value=int(input('enter {} value of list : '.format(i+1)))
```

```
    list1.append(value)



list2 =[]

n=int(input('Enter number of elements in second list : '))

for i in range(n):

    value=int(input('enter {} value of list : '.format(i+1)))

    list2.append(value)



print('list 1 : ',list1)

print('list 2 : ',list2)

tuple1=tuple(list1+list2)

print(tuple1)
```

```
Enter number of elements in first list : 5
enter 1 value of list : 1
enter 2 value of list : 5
enter 3 value of list : 8
enter 4 value of list : 8
enter 5 value of list : 5
```

```
Enter number of elements in second list : 5
enter 1 value of list : 5
enter 2 value of list : 6
enter 3 value of list : 7
enter 4 value of list : 8
enter 5 value of list : 9
list 1 :  [1, 5, 8, 8, 5]
list 2 :  [5, 6, 7, 8, 9]
(1, 5, 8, 8, 5, 5, 6, 7, 8, 9)
```

Python-Slip 29-

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

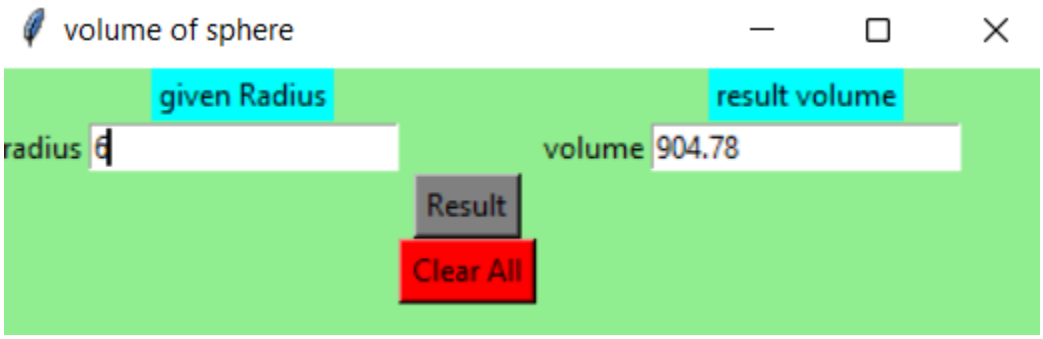
```
from tkinter import *

from tkinter import messagebox

import math

def clearAll() :

   radiusField.delete(0, END)

   volumeField.delete(0, END)
```

```
def checkError() :
    if (radiusField.get() == "") :
        messagebox.showerror("Input Error")
        clearAll()
        return -1


def getvolume() :
    value = checkError()
    if value == -1 :
        return
    else :
        radius0 = int(radiusField.get())
        volume0=round((4/3)*math.pi*radius0*radius0*radius0,2)
```

```
        volumeField.insert(10, str(volume0))



if __name__ == "__main__" :



    gui = Tk()

    gui.configure(background = "light green")

    gui.title("volume of sphere")

    gui.geometry("425x200")









    Radiuslabel = Label(gui, text = "given Radius", bg = "#00ffff")

    volumelabel = Label(gui, text = "result volume", bg = "#00ffff")
```

```
    Radius1 = Label(gui, text = "radius", bg = "light green")

    volume1 = Label(gui, text = "volume", bg = "light green")



    result = Button(gui, text = "Result", fg = "Black",

    bg = "gray", command = getvolume)

    clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

    bg = "Red", command = clearAll)



    radiusField = Entry(gui)



    volumeField = Entry(gui)



Radiuslabel.grid(row = 0, column = 1)

Radius1.grid(row = 1, column = 0)
```

```
radiusField.grid(row = 1, column = 1)




volumelabel.grid(row = 0, column = 4)

volume1.grid(row = 1, column = 3)

volumeField.grid(row = 1, column = 4)



result.grid(row = 4, column = 2)

clearAllEntry.grid(row = 12, column = 2)



gui.mainloop()
```



B)Write a Python script to sort (ascending and descending) a dictionary by key and Value.

```
dict1 = {}
n=int (input('Enter a number or pair in dict :'))
for i in range(n):
    key=input('enter {0} key :'.format(i+1))
    value=input('enter value of {}:'.format(key))
    dict1[key]=value

sorted_result = dict(sorted(dict1.items()))
print("\nSorting key in alphabetically ascending order:-")
print(sorted_result)

sorted_result = dict(sorted(dict1.items(), reverse=True))
print("\nSorting Dictionary by Key in Descending Order:-")
print(sorted_result)


a = dict(sorted(dict1.items(), key=lambda x: x[1]))
```

```
print("\nsort dictionary by value Ascending:-")
print(a)


a = dict(sorted(dict1.items(), key=lambda x: x[1], reverse=True))
print("\nSort dictionary by value descending:-")
print(a)
```

```
Enter a number or pair in dict :5
enter 1 key :6
enter value of 6:3
enter 2 key :6
enter value of 6:7
enter 3 key :8
enter value of 8:9
enter 4 key :8
enter value of 8:2
enter 5 key :14
enter value of 14:52

Sorting key in alphabetically ascending order:-
{'14': '52', '6': '7', '8': '2'}

Sorting Dictionary by Key in Descending Order:-
{'8': '2', '6': '7', '14': '52'}

sort dictionary by value Ascending:-
{'8': '2', '14': '52', '6': '7'}

Sort dictionary by value descending:-
{'6': '7', '14': '52', '8': '2'}
```

Python-Slip 30

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

```
from tkinter import *
```

```
from tkinter import messagebox



def clearAll() :

    str1Field.delete(0, END)

    char1Field.delete(0, END)

    resultField.delete(0, END)



def checkError() :

    if (str1Field.get() == "" or char1Field.get() == "") :

        messagebox.showerror("Input Error")

        clearAll()

        return -1



def occurrences() :
```

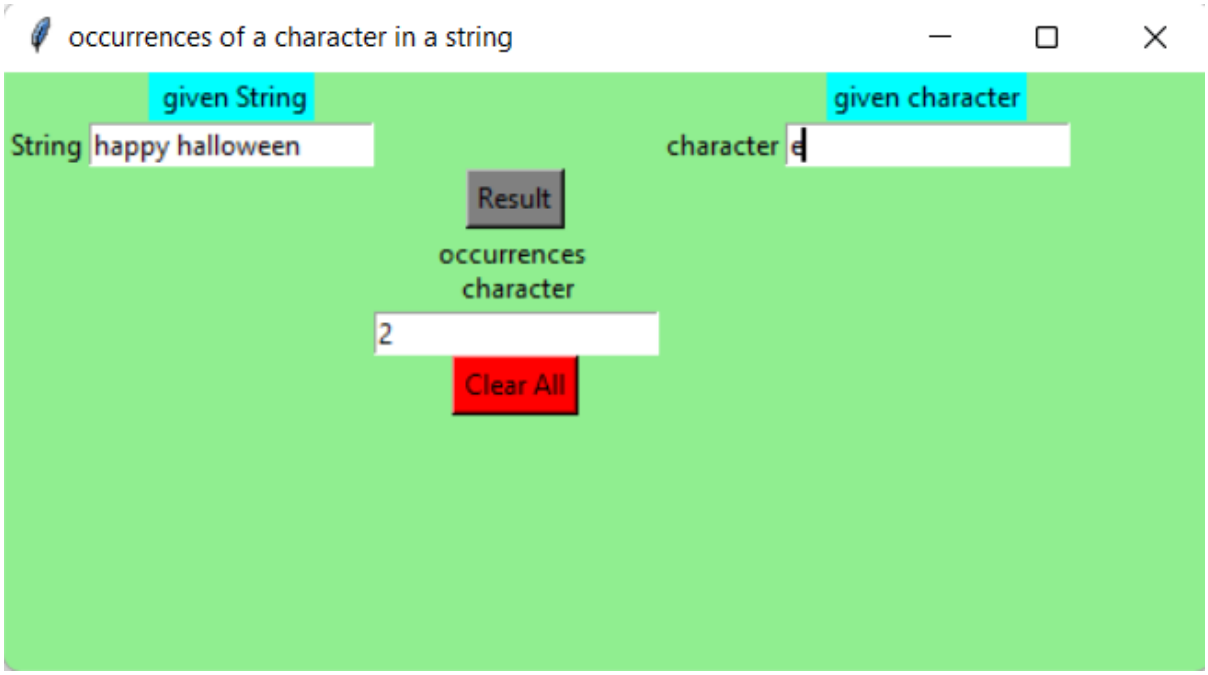
```
value = checkError()

if value == -1 :

    return

else :

    String0 = (str1Field.get())

    char0 = (char1Field.get())



    i=0

    count=0

    while(i<len(String0)):

        if(String0[i]==char0):

            count=count+1

        i=i+1
```

```
        resultField.insert(10, str(count))


if __name__ == "__main__" :


    gui = Tk()

    gui.configure(background = "light green")

    gui.title("occurrences of a character in a string")

    gui.geometry("525x260")




    Stringin = Label(gui, text = " given String", bg = "#00ffff")

    char = Label(gui, text = "given character", bg = "#00ffff")

    str1 = Label(gui, text = "String", bg = "light green")
```

```
char1 = Label(gui, text = "character", bg = "light green")



occurrenceslabel = Label(gui, text = "occurrences \n character",

bg = "light green")



result = Button(gui, text = "Result", fg = "Black",

bg = "gray", command = occurrences)

clearAllEntry = Button(gui, text = "Clear All", fg = "Black",

bg = "Red", command = clearAll)



str1Field = Entry(gui)

char1Field = Entry(gui)

resultField = Entry(gui)
```

```
Stringin.grid(row = 0, column = 1)

str1.grid(row = 1, column = 0)

str1Field.grid(row = 1, column = 1)




char.grid(row = 0, column = 4)

char1.grid(row = 1, column = 3)

char1Field.grid(row = 1, column = 4)



result.grid(row = 4, column = 2)

occurrenceslabel.grid(row = 5, column = 2)

resultField.grid(row = 6, column = 2)

clearAllEntry.grid(row = 12, column = 2)
```

gui.mainloop()



B) 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.

```
class stringmethod():

    def __init__(self):

        self.string=""
```

```
    def get(self):
        self.string=input("Enter string: ")

    def put(self):
        print("String is:")
        print(self.string)

obj=stringmethod()
obj.get()
obj.put()
```

```
Enter string: hello
String is:
hello
```