PYTHON PROJECT

Name : Raunak Dey (Roll Number : 19)

Department : Computer Science and Engineering Year : 2 (Semester : 3)

Batch : 2021-2025

ST. THOMAS COLLEGE OF ENGINEERING AND TECHNOLOGY

Objective:

To create a scientific calculator using GUI in Python.

Synopsis:

This code is a calculator program written in Python using the Tkinter library. It has a GUI interface that displays a calculator with a text field for input and buttons for various operations. The Calc class contains methods for performing arithmetic operations, displaying the input and result, and handling various other functionality such as clearing entries, taking the square root, and finding the cosine of an angle. The program also imports the math library, which is used in some of the methods.

The Calc class has an init method that initializes several instance variables including total, current, input_value, check_sum, op, and result. The numberEnter method is used to update the current value when the user inputs a number. The total_sum method calculates the result of an operation and displays it in the text field. The display method is used to update the text field with a specified value. The Clear_Entry and All_Clear_Entry methods are used to reset the current value and total value, respectively. The backspace method removes the last character from the current value.

The pi and e methods set the current value to the mathematical constants of pi and e, respectively. The mathPM method negates the current value. The squared method calculates the square root of the current value. The cos, cosh, tan, and tanh methods calculate the cosine, hyperbolic cosine, tangent, and hyperbolic tangent of the current value in radians.

The final block of code creates the buttons for the calculator and specifies their position, text, and command. When the program is run, the calculator window is displayed and the user can interact with it to perform various calculations.

Code:

```
from tkinter import *
import math
import tkinter.messagebox
windowr = Tk()
windowr.title("Calculator")
windowr.configure(background = 'gray63')
windowr.geometry("780x550")
calc = Frame(windowr)
calc.grid()
class Calc():
  def __init__(self):
    self.total=0
    self.current=''
    self.input_value=True
    self.check_sum=False
    self.op=''
    self.result=False
  def numberEnter(self, num):
    self.result=False
    firstnum=txtDisplay.get()
    secondnum=str(num)
```

```
        if self.input_value:
            self.current = secondnum
            self.input_value=False
        else:
            if secondnum == '.':
                if secondnum in firstnum:
                    return
            self.current = firstnum+secondnum
        self.display(self.current)
    def total_sum(self):
        self.result=True
        self.current=float(self.current)
        if self.check_sum==True:
            self.valid_function()
        else:
            self.total=float(txtDisplay.get())
    def display(self, value):
        txtDisplay.delete(0, END)
        txtDisplay.insert(0, value)
    def valid_function(self):
        if self.op == "add":
            self.total += self.current
        if self.op == "sub":
            self.total -= self.current
        if self.op == "multi":
            self.total *= self.current
        if self.op == "divide":
            self.total /= self.current
        if self.op == "mod":
            self.total %= self.current
        self.input_value=True
        self.check_sum=False
        self.display(self.total)
    def operation(self, op):
        self.current = float(self.current)
        if self.check_sum:
            self.valid_function()
        elif not self.result:
            self.total=self.current
            self.input_value=True
```

```
        self.check_sum=True
        self.op=op
        self.result=False
    def Clear_Entry(self):
        self.result = False
        self.current = "0"
        self.display(0)
        self.input_value=True
    def All_Clear_Entry(self):
        self.Clear_Entry()
        self.total=0
    def backspace(self):
        if len(self.current) > 0:
            self.current = self.current[:-1]
            self.display(self.current)
    def pi(self):
        self.result = False
        self.current = math.pi
        self.display(self.current)
    def e(self):
        self.result = False
        self.current = math.e
        self.display(self.current)
    def mathPM(self):
        self.result = False
        self.current = -(float(txtDisplay.get()))
        self.display(self.current)
    def squared(self):
        self.result = False
        self.current = math.sqrt(float(txtDisplay.get()))
        self.display(self.current)
    def cos(self):
        self.result = False
        self.current = math.cos(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def cosh(self):
        self.result = False
        self.current = math.cosh(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def tan(self):
```

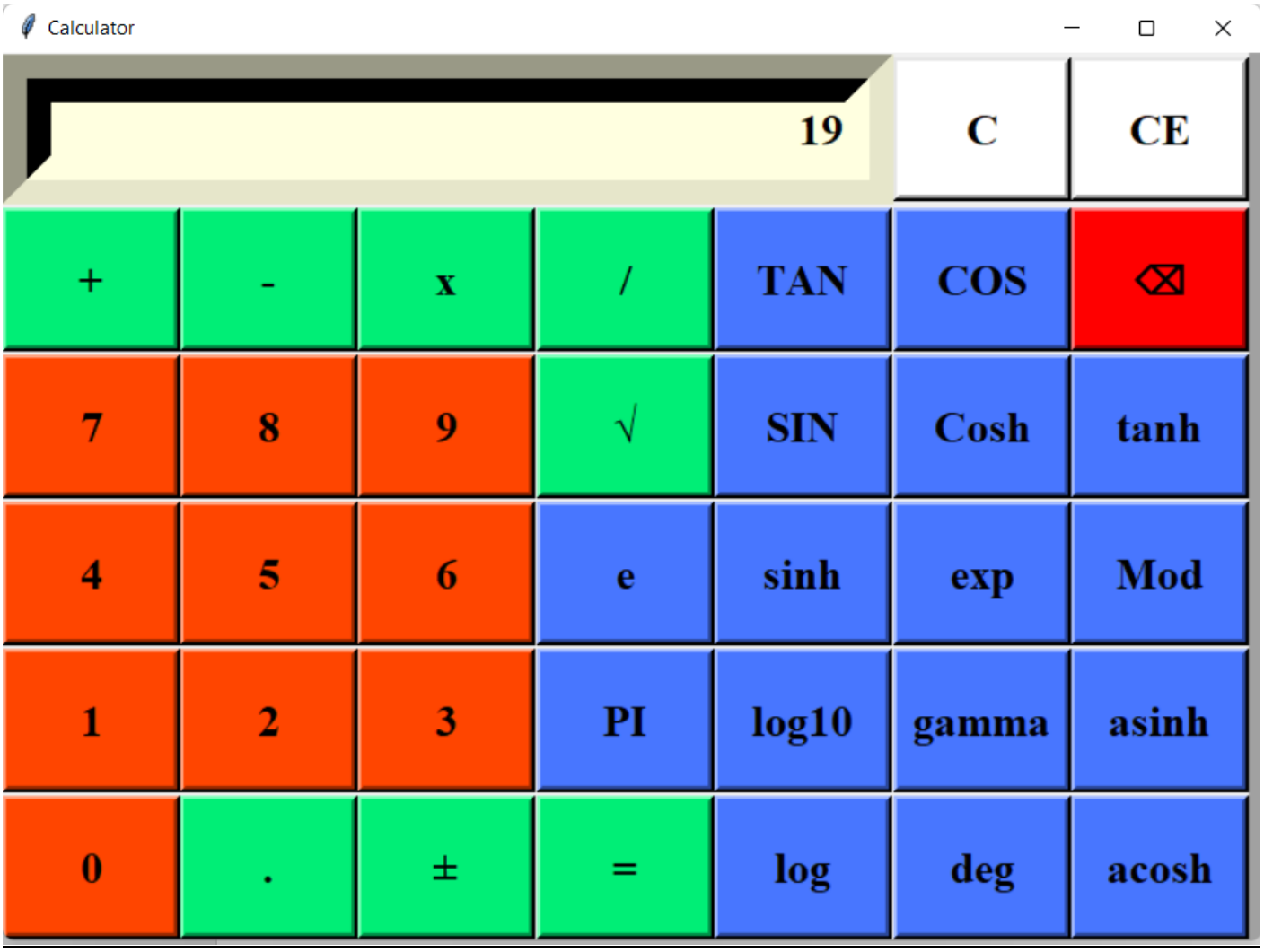
```
        self.result = False
        self.current = math.tan(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def tanh(self):
        self.result = False
        self.current = math.tanh(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def sin(self):
        self.result = False
        self.current = math.sin(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def sinh(self):
        self.result = False
        self.current = math.sinh(math.radians(float(txtDisplay.get())))
        self.display(self.current)
    def log(self):
        self.result = False
        self.current = math.log(float(txtDisplay.get()))
        self.display(self.current)
    def exp(self):
        self.result = False
        self.current = math.exp(float(txtDisplay.get()))
        self.display(self.current)
    def acosh(self):
        self.result = False
        self.current = math.acosh(float(txtDisplay.get()))
        self.display(self.current)
    def asinh(self):
        self.result = False
        self.current = math.asinh(float(txtDisplay.get()))
        self.display(self.current)
    def gamma(self):
        self.result = False
        self.current = math.gamma(float(txtDisplay.get()))
        self.display(self.current)
    def degrees(self):
        self.result = False
        self.current = math.degrees(float(txtDisplay.get()))
        self.display(self.current)
    def log10(self):
```

```
        self.result = False
        self.current = math.log10(float(txtDisplay.get()))
        self.display(self.current)
numbers = Calc()
txtDisplay = Entry(calc, font=('Times New Roman',20,'bold'),bg='light yellow',fg='black',bd=30,width=35,justify=RIGHT)
txtDisplay.grid(row=0,column=0, columnspan=5, pady=1)
txtDisplay.insert(0,"0")
numeric_keypad = "789456123"
i=0
btn = []
for r in range(2,5):
    for d in range(3):
        btn.append(Button(calc, width=6, height=2,bg='orange red',fg='black',font=('Times New Roman',20,'bold'),
                  bd=4,text=numeric_keypad[i]))
        btn[i].grid(row=r, column= d, pady = 1)
        btn[i]["command"]=lambda x=numeric_keypad[i]:numbers.numberEnter(x)
        i+=1


Clearall = Button(calc, text=chr(67)+chr(69),width=6, height=2,fg='black',bg='white',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.All_Clear_Entry).grid(row=0, column= 6, pady = 1)
Clear = Button(calc, text=chr(67),width=6, height=2,bg='white',fg='black',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.Clear_Entry).grid(row=0, column= 5, pady = 1)
backspace = Button(calc, text = "\u232B", width = 6, height = 2,
 font = ('Times New Roman',20,'bold'),bd=4, bg = "red", command = lambda:numbers.backspace()).grid(row=1,column=6,pady=1)
button_squareroot = Button(calc, text="\u221A",width=6, height=2,bg='SpringGreen2', font=('Times New Roman',  20,'bold'),
bd=4,command=numbers.squared).grid(row=2, column= 3, pady = 1)
Addition = Button(calc, text="+",width=6, height=2,bg='SpringGreen2',font=('Times New Roman',20,'bold'),
bd=4,command=lambda:numbers.operation("add")).grid(row=1, column= 0, pady = 1)
Subtract = Button(calc, text="-",width=6,  height=2,bg='SpringGreen2',font=('Times New
Roman',20,'bold'),bd=4,command=lambda:numbers.operation("sub")).grid(row=1, column= 1, pady = 1)
Multiplication = Button(calc, text="x",width=6, height=2,bg='SpringGreen2',font=('Times New Roman',20,'bold'),
bd=4,command=lambda:numbers.operation("multi")).grid(row=1, column= 2, pady = 1)
Division = Button(calc, text="/",width=6,height=2,bg='SpringGreen2',  font=('Times New Roman',20,'bold'),
bd=4,command=lambda:numbers.operation("divide")).grid(row=1, column= 3, pady = 1)
zero = Button(calc, text="0",width=6,height=2,bg='orange red',fg='black',font=('Times New Roman',20,'bold'),
bd=4,command=lambda:numbers.numberEnter(0)).grid(row=5, column= 0, pady = 1)
Dot = Button(calc, text=".",width=6,height=2,bg='SpringGreen2',font=('Times New Roman',20,'bold'),
bd=4,command=lambda:numbers.numberEnter(".")).grid(row=5, column= 1, pady = 1)
buttonPM = Button(calc, text=chr(177),width=6,  height=2,bg='SpringGreen2', font=('Times New Roman',20,'bold'),
bd=4,command=numbers.mathPM).grid(row=5, column= 2, pady = 1)
```

```
Equal = Button(calc, text="=",width=6,height=2,bg='SpringGreen2',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.total_sum).grid(row=5, column= 3, pady = 1)
Pi_button = Button(calc, text="PI",width=6, height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.pi).grid(row=4, column= 3, pady = 1)
Cos_button = Button(calc, text="COS",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.cos).grid(row=1, column= 5, pady = 1)
Tan_button = Button(calc, text="TAN",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'),bd=4,command=numbers.tan).grid(row=1, column= 4, pady = 1)
sin_button = Button(calc, text="SIN",width=6,height=2,bg='RoyalBlue1',fg='black',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.sin).grid(row=2, column= 4, pady = 1)
Cosh_button = Button(calc, text="Cosh",width=6, height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.cosh).grid(row=2, column= 5, pady = 1)
Tanh_button = Button(calc, text="tanh",width=6, height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.tanh).grid(row=2, column= 6, pady = 1)
sinh_button = Button(calc, text="sinh",width=6, height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'),bd=4,command=numbers.sinh).grid(row=3, column= 4, pady = 1)
logOption = Button(calc, text="log",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.log).grid(row=5, column= 4, pady = 1)
Exp_button = Button(calc, text="exp",width=6, height=2, bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.exp).grid(row=3, column= 5, pady = 1)
Mod_button = Button(calc, text="Mod",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=lambda:numbers.operation("mod")).grid(row=3, column= 6, pady = 1)
ButtonE = Button(calc, text="e",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.e).grid(row=3, column= 3, pady = 1)
Button_log10 = Button(calc, text="log10",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'),bd=4,command=numbers.log10).grid(row=4, column= 4, pady = 1)
gamma_button = Button(calc, text="gamma",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'),bd=4,command=numbers.gamma).grid(row=4, column= 5, pady = 1)
Button_deg = Button(calc, text="deg",width=6,height=2,bg='RoyalBlue1',fg='black',
font=('Times New Roman',20,'bold'), bd=4,command=numbers.degrees).grid(row=5, column= 5, pady = 1)
Button_acosh = Button(calc, text="acosh",width=6,height=2,bg='RoyalBlue1',fg='black',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.acosh).grid(row=5, column= 6, pady = 1)
Button_asinh = Button(calc, text="asinh",width=6,height=2,bg='RoyalBlue1',fg='black',font=('Times New Roman',20,'bold'),
bd=4,command=numbers.asinh).grid(row=4, column= 6, pady = 1)
def on_closing():
    if tkinter.messagebox.askokcancel("EXIT", "Do you want to Exit?"):
        windowr.destroy()
windowr.protocol("WM_DELETE_WINDOW", on_closing)
windowr.mainloop()
```

Output:


Calculator
19
C
CE
+
-
x
/
TAN
COS
⌫
7
8
9
√
SIN
Cosh
tanh
4
5
6
e
sinh
exp
Mod
1
2
3
PI
log10
gamma
asinh
0
.
±
=
log
deg
acosh