Batch: A3 **Roll No.:** 1911034

Experiment / assignment / tutorial No._______

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE: Scientific Calculator using Tkinter

AIM: To understand the concepts of Tkinter

Expected OUTCOME of Experiment:

CO -

Books/ Journals/ Websites referred:

1. W3Schools.com
2. Geeksforgeeks.com
3. tutorialspoint
4. powerpoints

Pre Lab/ Prior Concepts:

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

- **Tkinter** − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.
- **wxPython** − This is an open-source Python interface for wxWindows http://wxpython.org.
- **JPython** − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine http://www.jython.org.

There are many other interfaces available, which you can find them on the net.

Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

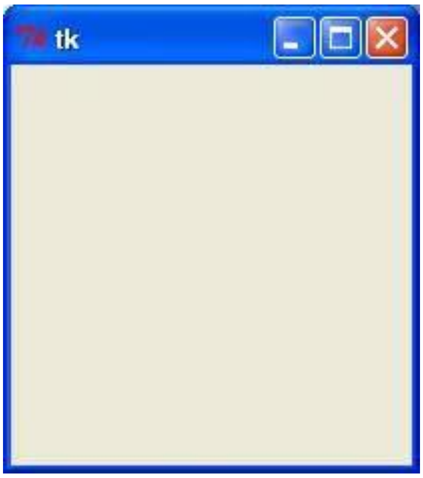
Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Example

```
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

This would create a following window −



Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter. We present these widgets as well as a brief description in the following table −

Sr.No.	Operator & Description
1	Button The Button widget is used to display buttons in your application.
2	Canvas The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.
3	Checkbutton The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time.
4	Entry The Entry widget is used to display a single-line text field for accepting values from a user.
5	Frame The Frame widget is used as a container widget to organize other widgets.
6	Label The Label widget is used to provide a single-line caption for other widgets. It can also contain images.
7	Listbox The Listbox widget is used to provide a list of options to a user.
8	Menubutton The Menubutton widget is used to display menus in your application.

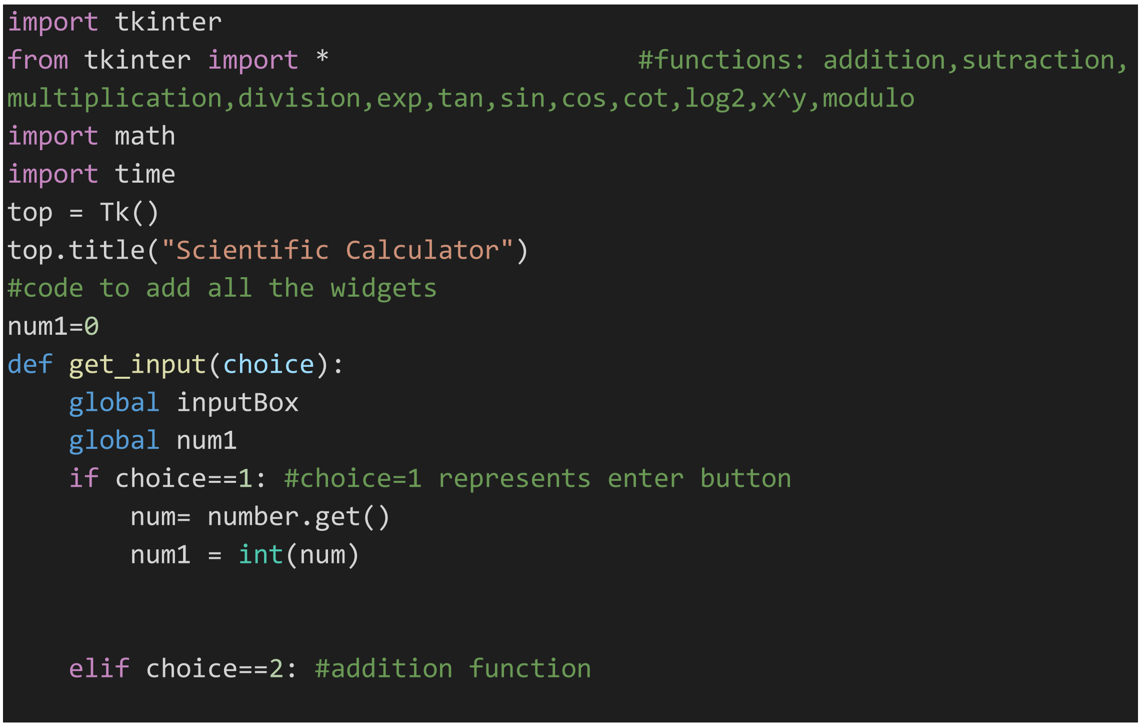
9	Menu The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.
10	Message The Message widget is used to display multiline text fields for accepting values from a user.
11	Radiobutton The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.
12	Scale The Scale widget is used to provide a slider widget.
13	Scrollbar The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes.
14	Text The Text widget is used to display text in multiple lines.
15	Toplevel The Toplevel widget is used to provide a separate window container.
16	Spinbox The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values.
17	PanedWindow A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically.

18	LabelFrame A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.
19	tkMessageBox This module is used to display message boxes in your applications.

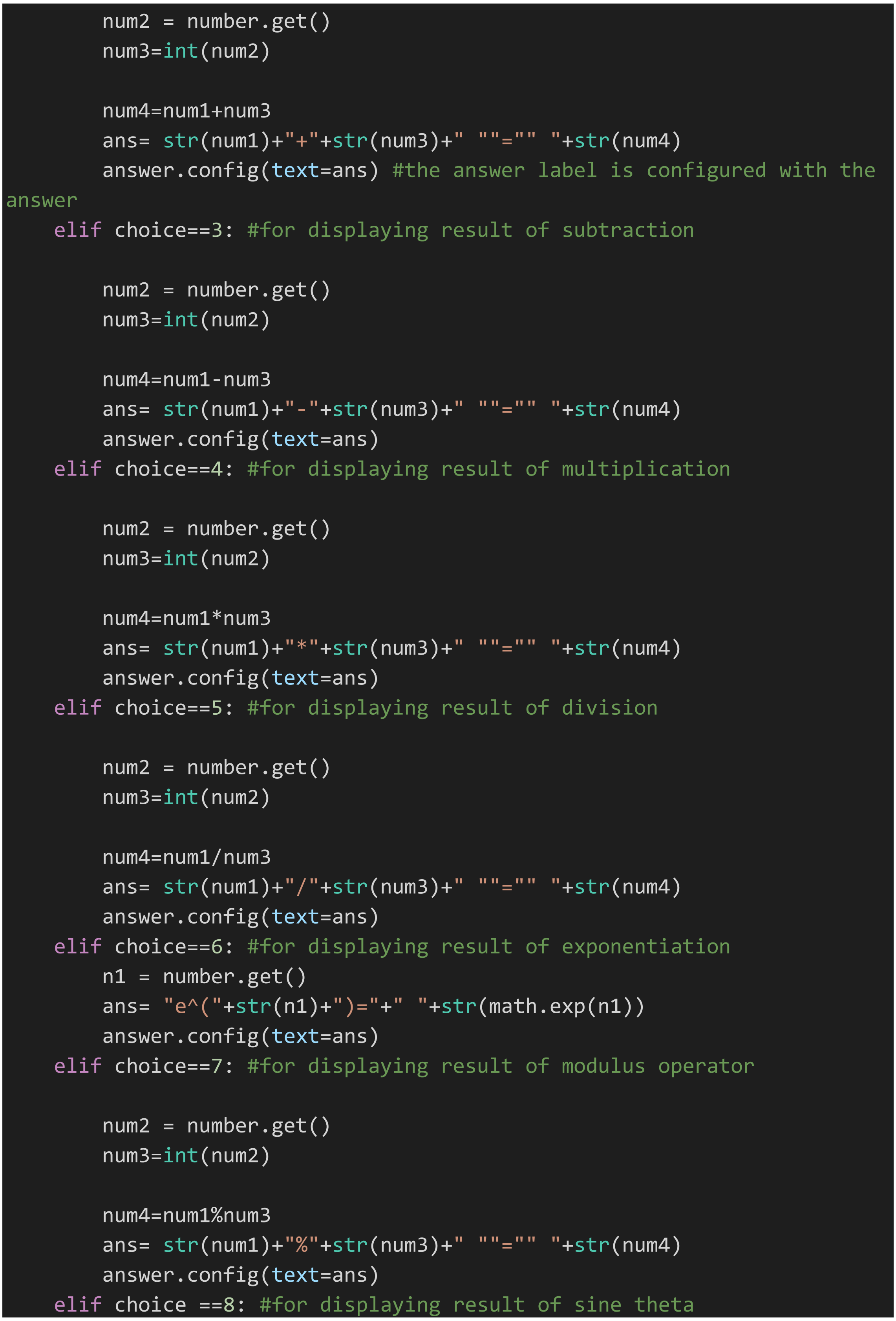
Problem statement:

Demonstrate the use of GUI Widgets to create Scientific Calculator (Min. 10 functionalities).

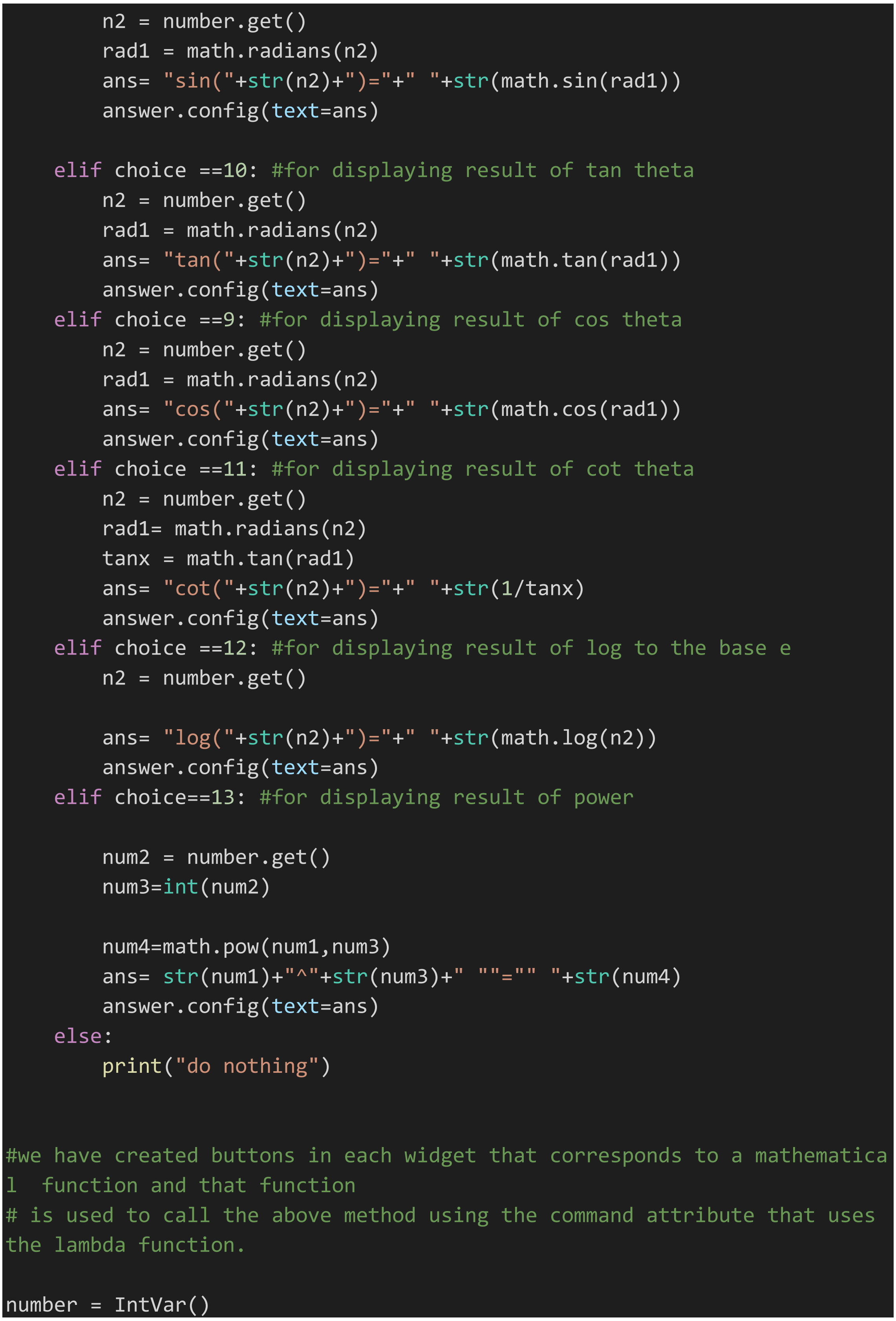
Program (with comments) (Write comments to your program to explain the implementation)



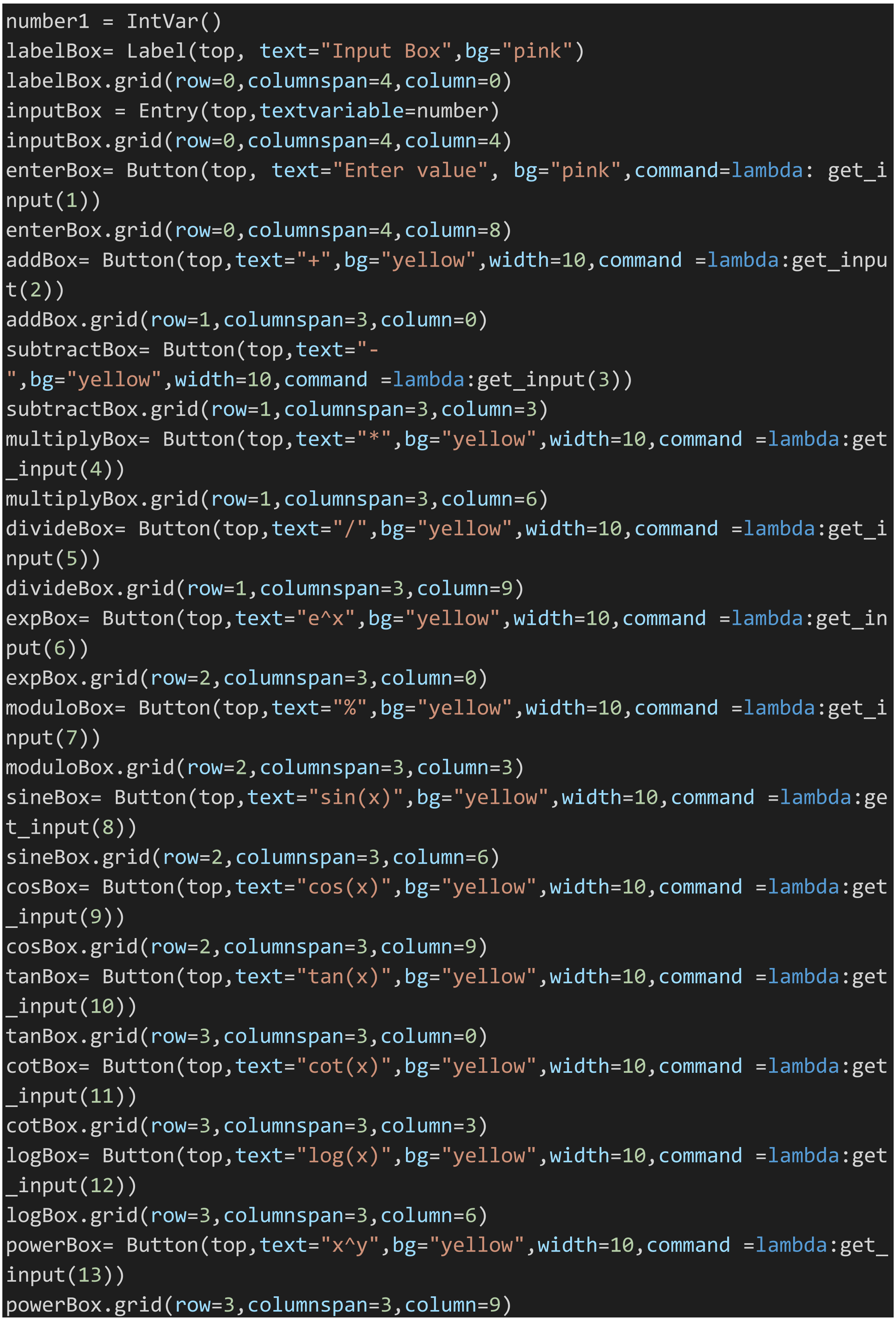
```
import tkinter
from tkinter import *                    #functions: addition,sutraction,
multiplication,division,exp,tan,sin,cos,cot,log2,x^y,modulo
import math
import time
top = Tk()
top.title("Scientific Calculator")
#code to add all the widgets
num1=0
def get_input(choice):
    global inputBox
    global num1
    if choice==1: #choice=1 represents enter button
        num= number.get()
        num1 = int(num)


    elif choice==2: #addition function
```



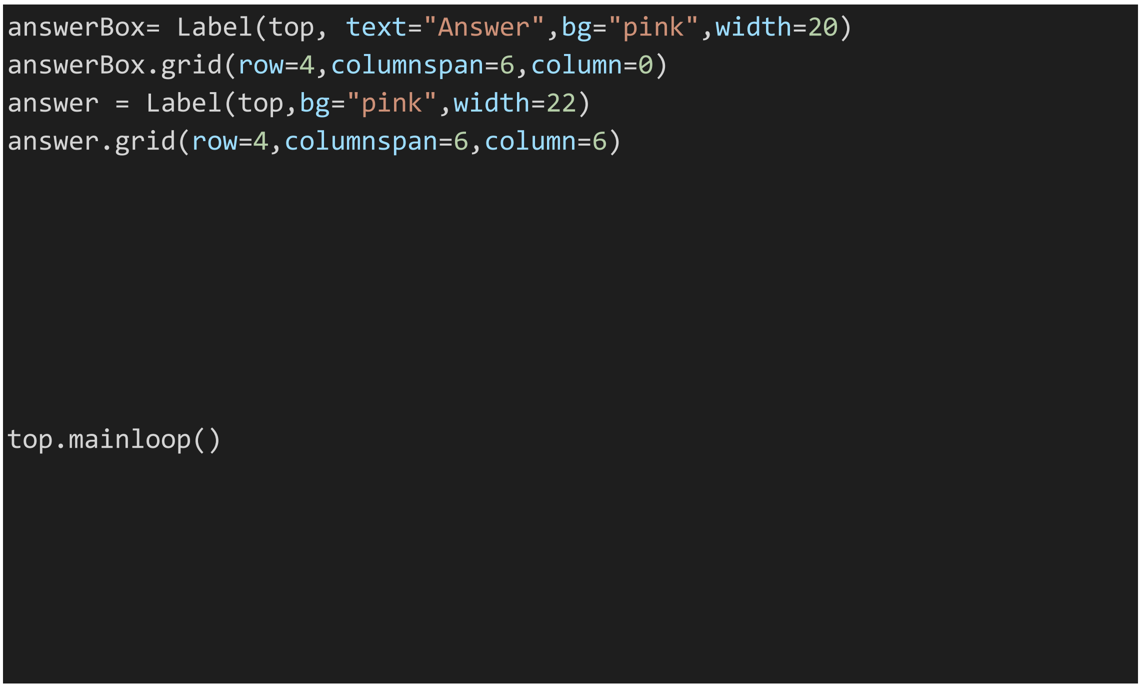
```
        num2 = number.get()
        num3=int(num2)

        num4=num1+num3
        ans= str(num1)+"+"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans) #the answer label is configured with the
answer
    elif choice==3: #for displaying result of subtraction

        num2 = number.get()
        num3=int(num2)

        num4=num1-num3
        ans= str(num1)+"-"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans)
    elif choice==4: #for displaying result of multiplication

        num2 = number.get()
        num3=int(num2)

        num4=num1*num3
        ans= str(num1)+"*"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans)
    elif choice==5: #for displaying result of division

        num2 = number.get()
        num3=int(num2)

        num4=num1/num3
        ans= str(num1)+"/"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans)
    elif choice==6: #for displaying result of exponentiation
        n1 = number.get()
        ans= "e^("+str(n1)+")="+" "+str(math.exp(n1))
        answer.config(text=ans)
    elif choice==7: #for displaying result of modulus operator

        num2 = number.get()
        num3=int(num2)

        num4=num1%num3
        ans= str(num1)+"%"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans)
    elif choice ==8: #for displaying result of sine theta
```



```
        n2 = number.get()
        rad1 = math.radians(n2)
        ans= "sin("+str(n2)+")="+" "+str(math.sin(rad1))
        answer.config(text=ans)

    elif choice ==10: #for displaying result of tan theta
        n2 = number.get()
        rad1 = math.radians(n2)
        ans= "tan("+str(n2)+")="+" "+str(math.tan(rad1))
        answer.config(text=ans)
    elif choice ==9: #for displaying result of cos theta
        n2 = number.get()
        rad1 = math.radians(n2)
        ans= "cos("+str(n2)+")="+" "+str(math.cos(rad1))
        answer.config(text=ans)
    elif choice ==11: #for displaying result of cot theta
        n2 = number.get()
        rad1= math.radians(n2)
        tanx = math.tan(rad1)
        ans= "cot("+str(n2)+")="+" "+str(1/tanx)
        answer.config(text=ans)
    elif choice ==12: #for displaying result of log to the base e
        n2 = number.get()

        ans= "log("+str(n2)+")="+" "+str(math.log(n2))
        answer.config(text=ans)
    elif choice==13: #for displaying result of power

        num2 = number.get()
        num3=int(num2)

        num4=math.pow(num1,num3)
        ans= str(num1)+"^"+str(num3)+" ""="" "+str(num4)
        answer.config(text=ans)
    else:
        print("do nothing")


#we have created buttons in each widget that corresponds to a mathematica
l  function and that function
# is used to call the above method using the command attribute that uses
the lambda function.

number = IntVar()
```



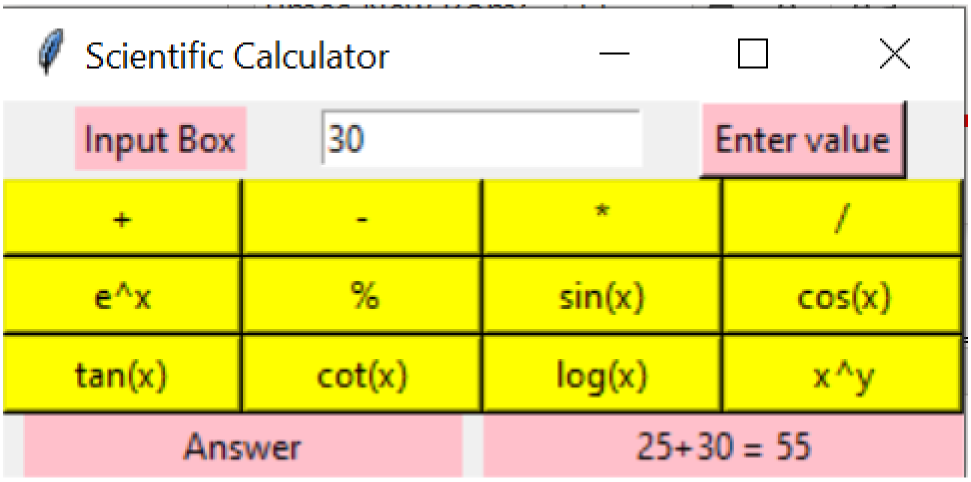
```
number1 = IntVar()
labelBox= Label(top, text="Input Box",bg="pink")
labelBox.grid(row=0,columnspan=4,column=0)
inputBox = Entry(top,textvariable=number)
inputBox.grid(row=0,columnspan=4,column=4)
enterBox= Button(top, text="Enter value", bg="pink",command=lambda: get_input(1))
enterBox.grid(row=0,columnspan=4,column=8)
addBox= Button(top,text="+",bg="yellow",width=10,command =lambda:get_input(2))
addBox.grid(row=1,columnspan=3,column=0)
subtractBox= Button(top,text="-",bg="yellow",width=10,command =lambda:get_input(3))
subtractBox.grid(row=1,columnspan=3,column=3)
multiplyBox= Button(top,text="*",bg="yellow",width=10,command =lambda:get_input(4))
multiplyBox.grid(row=1,columnspan=3,column=6)
divideBox= Button(top,text="/",bg="yellow",width=10,command =lambda:get_input(5))
divideBox.grid(row=1,columnspan=3,column=9)
expBox= Button(top,text="e^x",bg="yellow",width=10,command =lambda:get_input(6))
expBox.grid(row=2,columnspan=3,column=0)
moduloBox= Button(top,text="%",bg="yellow",width=10,command =lambda:get_input(7))
moduloBox.grid(row=2,columnspan=3,column=3)
sineBox= Button(top,text="sin(x)",bg="yellow",width=10,command =lambda:get_input(8))
sineBox.grid(row=2,columnspan=3,column=6)
cosBox= Button(top,text="cos(x)",bg="yellow",width=10,command =lambda:get_input(9))
cosBox.grid(row=2,columnspan=3,column=9)
tanBox= Button(top,text="tan(x)",bg="yellow",width=10,command =lambda:get_input(10))
tanBox.grid(row=3,columnspan=3,column=0)
cotBox= Button(top,text="cot(x)",bg="yellow",width=10,command =lambda:get_input(11))
cotBox.grid(row=3,columnspan=3,column=3)
logBox= Button(top,text="log(x)",bg="yellow",width=10,command =lambda:get_input(12))
logBox.grid(row=3,columnspan=3,column=6)
powerBox= Button(top,text="x^y",bg="yellow",width=10,command =lambda:get_input(13))
powerBox.grid(row=3,columnspan=3,column=9)
```



```
answerBox= Label(top, text="Answer",bg="pink",width=20)
answerBox.grid(row=4,columnspan=6,column=0)
answer = Label(top,bg="pink",width=22)
answer.grid(row=4,columnspan=6,column=6)







top.mainloop()
```

Output Screenshots

1. **Addition**

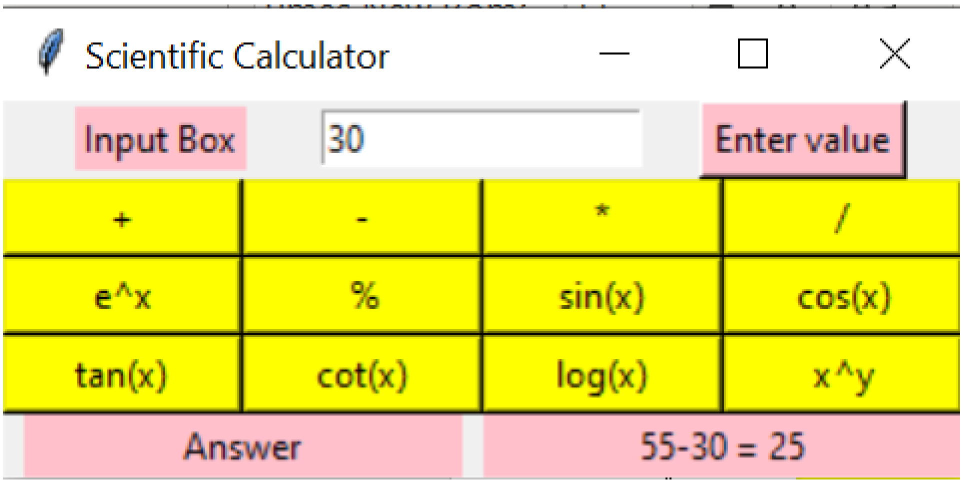


Scientific Calculator

Input Box: 30

Answer: 25+30 = 55

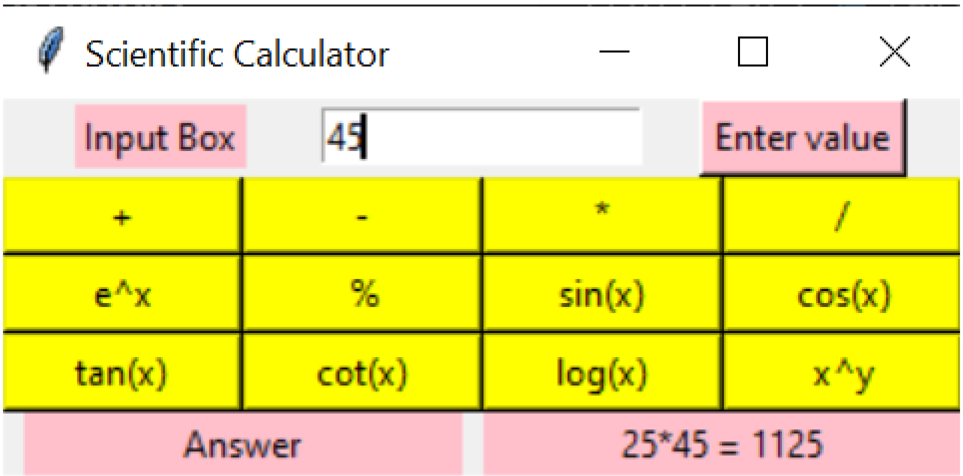
2. **Subtraction**



Input Box 30

Answer 55-30 = 25

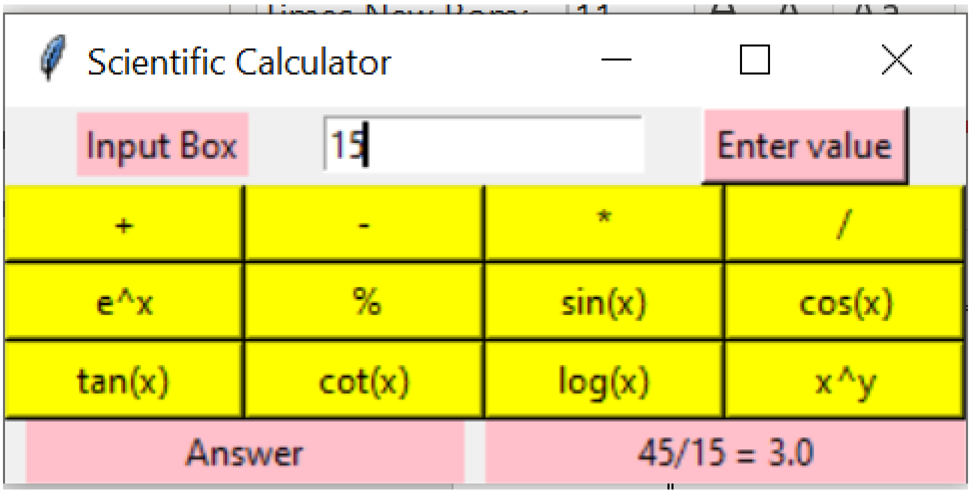
3. **Multiplication**



Input Box 45

Answer 25*45 = 1125

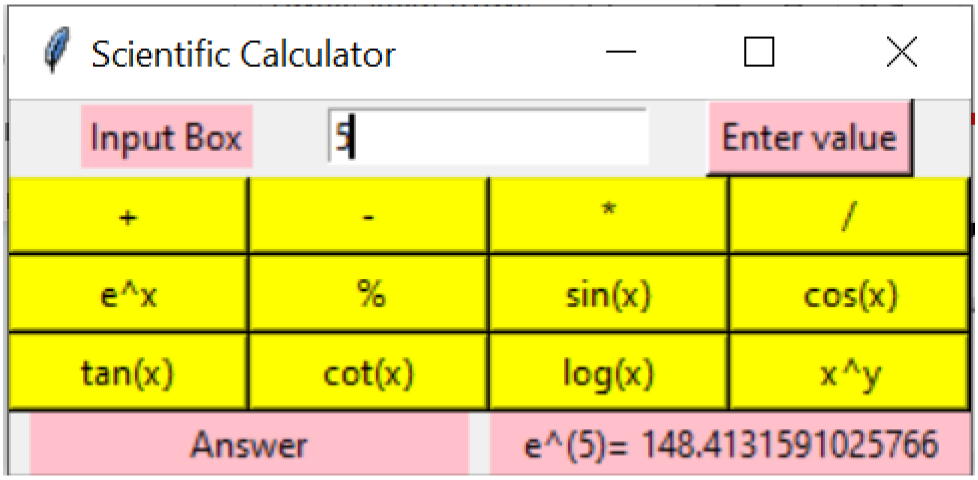
4. **Division:**



15

45/15 = 3.0

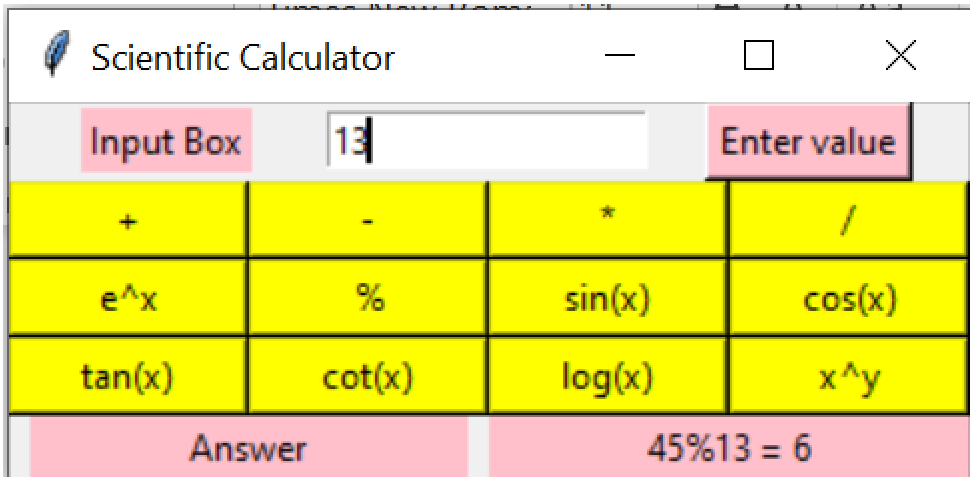
5. Exponentiation :



5

e^(5)= 148.4131591025766

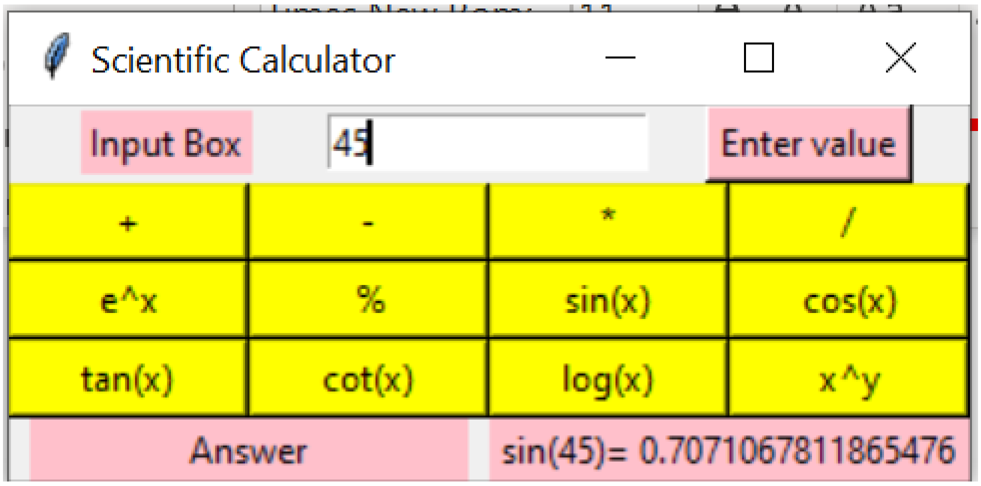
6. Modulus



13

45%13 = 6

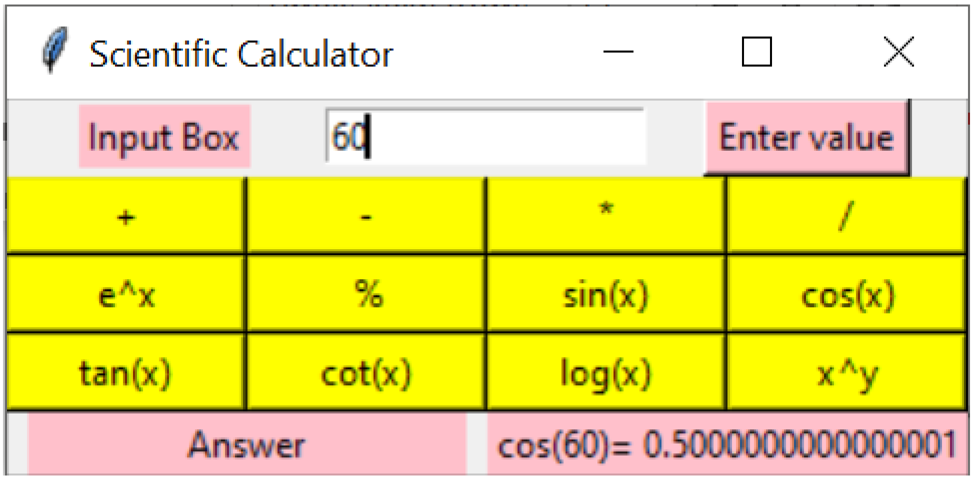
7. **Sin(x)**



45

sin(45)= 0.7071067811865476

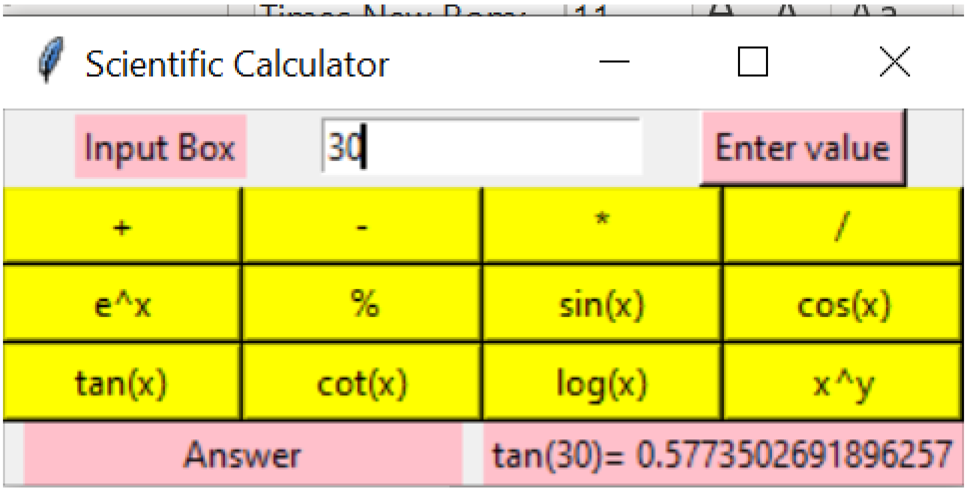
8. **Cos(x)**



60

cos(60)= 0.5000000000000001

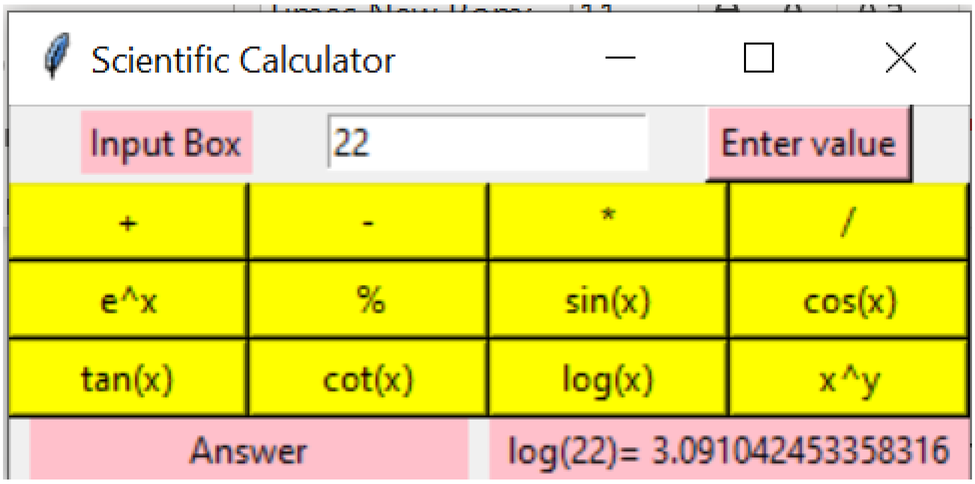
9. **Tan(x)**



30

Answer tan(30)= 0.5773502691896257

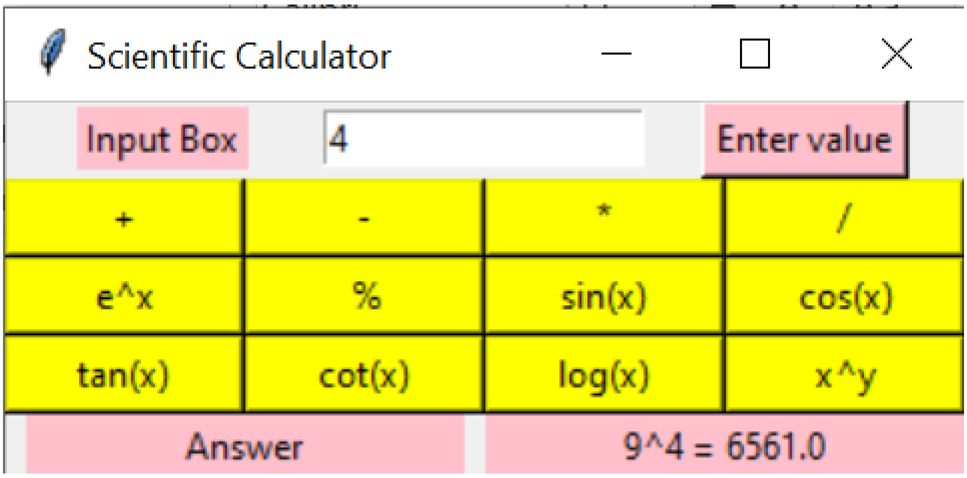
10. Log(x)



22

Answer log(22)= 3.091042453358316

11. x^y (x to the power y)



4

Answer 9^4 = 6561.0

Conclusion: In this experiment we were able to understand and implement the different functionalities of Tkinter in order to create a scientific calculator.

Date: _____________ **Signature of faculty in-charge**