K. J. Somaiya College of Engineering, Mumbai-77

Department of Computer Engineering

Roll No.: 1911027

Experiment No. 6

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

Signature of the Staff In-charge with date

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

AIM: Use concepts of tkinter GUI widgets

Expected OUTCOME of Experiment:

CO1: Describe the Numbers, Math functions, Strings, List, Tuples and Dictionaries in Python.

CO2: Interpret different Decision Making statements, Functions, Object oriented programming in Python.

CO4: Explain how to design GUI Applications in Python and evaluate different database operations.

Books/ Journals/ Websites referred:

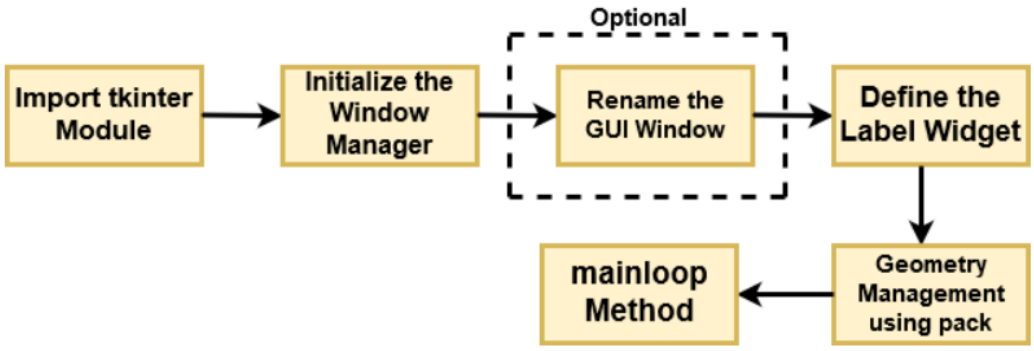
1) https://www.tutorialspoint.com/python/python_gui_programming
2) https://docs.python.org/3/library/tkinter.html
3) https://www.geeksforgeeks.org/python-gui-tkinter/
4) https://www.datacamp.com/community/tutorials/gui-tkinter-python

---

Pre Lab/ Prior Concepts:

Tkinter:

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. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at ActiveState). Most of the time, tkinter is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named _tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter. In addition to the Tk interface module, tkinter includes a number of Python modules, tkinter.constants being one of the most important. Importing tkinter will automatically import tkinter.constants



Flow Diagram for Rendering a Basic GUI

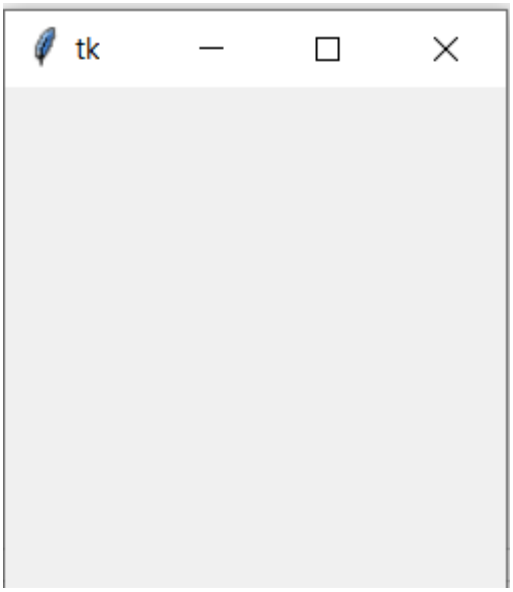
Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

- 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:

```
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.

Standard attributes:

Let us take a look at how some of their common attributes such as sizes, colours and fonts are specified.

Dimensions: 1) Colours 2) Fonts 3) Anchors 4) Relief styles 5) Bitmaps 6) Cursors

Geometry Management:

All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area. Tkinter exposes the following geometry manager classes: pack, grid, and place.

The pack() Method − This geometry manager organizes widgets in blocks before placing them in the parent widget.

The grid() Method − This geometry manager organizes widgets in a table-like structure in the parent widget.

The place() Method − This geometry manager organizes widgets by placing them in a specific position in the parent widget.

Examples:

1) Check Button: To select any number of options by displaying a number of options to a user as toggle buttons. The general syntax is:

w = CheckButton(master, option=value)

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

Title: To set the title of the widget.

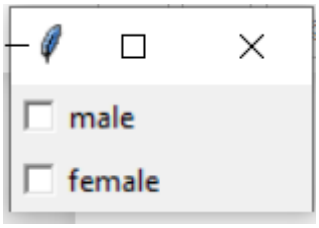
activebackground: to set the background colour when widget is under the cursor.

activeforeground: to set the foreground colour when widget is under the cursor.

Code:

```
from tkinter import *

master = Tk()

var1 = IntVar()

Checkbutton(master, text='male', variable=var1).grid(row=0, sticky=W)

var2 = IntVar()

Checkbutton(master, text='female', variable=var2).grid(row=1, sticky=W)

mainloop()
```

Output:



2) Entry: It is used to input the single line text entry from the user.. For multi-line text input, Text widget is used. The general syntax is:

w=Entry(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

bd: to set the border width in pixels.

bg: to set the normal background colour.

cursor: to set the cursor used.

command: to call a function.

Program:

```
from tkinter import *
import math
import re
def preced(sym):
    if(sym=='^'):
        return 3
    elif(sym=='*' or sym=='/'):
        return 2
    elif(sym=='+' or sym=='-'):
        return 1
    else:
```

```
        return 0


def cal(str1):
    flag=0
    flag1=0
    str1=str1+")"
    stack=list()
    stack=[None]*100
    stack[0]="("
    post=list()
    post=[None]*100
    j=0
    k=0
    i=0
    while(i<len(str1)):
        match=re.search(r'[0-9]',str1[i])
        if(match):
            tem=""
            for t in range(i,len(str1)):
                match1=re.search(r'[0-9]|[.]',str1[t])
                if(match1):
                    tem=tem+str1[t]
                    i=i+1
                else:
                    break
            post[j]=tem
            j=j+1
        elif(str1[i]=='('):
            k=k+1
```

```
        stack[k]=str1[i]
        i=i+1
    elif(str1[i]=='^' or str1[i]=='+' or str1[i]=='-' or str1[i]=='/' or str1[i]=="*"):
        temp=stack[k]
        k=k-1
        while(preced(temp)>=preced(str1[i]) and (temp=='^' or temp=='+' or
temp=='-' or temp=='*' or temp=='/')):
            post[j]=temp
            j=j+1
            temp=stack[k]
            k=k-1
        k=k+1
        stack[k]=temp
        k=k+1
        stack[k]=str1[i]
        i=i+1
    elif(str1[i]==')'):
        temp=stack[k]
        k=k-1
        while(temp!='('):
            post[j]=temp
            j=j+1
            temp=stack[k]
            k=k-1
        i=i+1
    elif(re.search(r'[a-z]',str1[i])):
        exp=""
        temp=""
        for j in range(i,len(str1)):
```

```
    if(re.search(r'[a-z]',str1[j])):
        temp=temp+str1[j]
        i=i+1
    else:
        break
for m in range(i,len(str1)):
    if(str1[m]=='('):
        i=i+1
        continue
    elif(str1[m]!=")"):
        exp=exp+str1[m]
        i=i+1
    elif(str1[m]==")"):
        i=i+1
        break
if(temp=="sin"):
    cal1=float(exp)*math.pi/180
    calans=math.sin(cal1)
elif(temp=="cos"):
    cal1=float(exp)*math.pi/180
    calans=math.cos(cal1)
elif(temp=="tan"):
    if(float(exp)!=90.0):
        cal1=float(exp)*math.pi/180
        calans=math.tan(cal1)
    else:
        e.delete(0,END)
        e.insert(0,"Invalid Operation")
        flag1=1
```

```
            break
        elif(temp=="log"):
            calans=math.log(float(exp))
        elif(temp=="rot"):
            calans=math.sqrt(float(exp))
        elif(temp=="sinh"):
            calans=math.asin(float(exp))*180/math.pi
        elif(temp=="cosh"):
            calans=math.acos(float(exp))*180/math.pi
        elif(temp=="tanh"):
            calans=math.atan(float(exp))*180/math.pi
        elif(temp=="e"):
            calans=math.pow(2.718,float(exp))
        post[j]=calans
        j=j+1
    if(flag1!=1):
        calcupo(post)

def calcupo(post):
    stack=list()
    stack=[None]*100
    k=-1
    flag=0
    for i in range(0,len(post)):
        if(post[i]!=None):
            match=re.search(r'[0-9]|[0-9]*[.][0-9]*',str(post[i]))
            if(match):
                k=k+1
                stack[k]=float(post[i])
```

```
        else:
            num1=float(stack[k])
            k=k-1
            num2=float(stack[k])
            k=k-1
            if(post[i]=='/'):
                if(num1==0):
                    e.delete(0,END)
                    e.insert(0,"Division by zero occured")
                    flag=1
                    break
                ans=num2/num1
            elif(post[i]=='*'):
                ans=num2*num1
            elif(post[i]=='+'):
                ans=num2+num1
            elif(post[i]=='-'):
                ans=num2-num1
            else:
                ans=num2**num1
            k=k+1
            stack[k]=ans
    if(flag==0):
        e.delete(0,END)
        e.insert(0,str(stack[k]))


def cleartex():
    e.delete('0', END)
```

```
def setTextInput(text):
    cur=e.get()
    e.delete(0,END)
    e.insert(0,str(cur)+str(text))

def gettext():
    str1=e.get()
    if(not re.search(r'[0-9]',str1)):
        e.delete(0,END)
        e.insert(0,"Invalid Operation")
    elif(re.search(r'[!]$',str1)):
        if(re.search(r'[.]',str1)):
            e.delete(0,END)
            e.insert(0,"Factorial Not Possible")
        else:
            sum=1
            act=int(str1[:-1])
            for i in range(1,act+1):
                sum=sum*i
            e.delete(0,END)
            e.insert(0,str(sum))
    else:
        cal(str1)


root=Tk()
root.title('Scientific Caclulator')
root.geometry('680x350')
```

```
e=Entry(root,borderwidth=5,width=85)
e.grid(row=0,column=0,columnspan=9,padx=20,pady=20)
b0=Button(root,text='0',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("0"))
b1=Button(root,text='1',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("1"))
b2=Button(root,text='2',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("2"))
b3=Button(root,text='3',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("3"))
b4=Button(root,text='4',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("4"))
b5=Button(root,text='5',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("5"))
b6=Button(root,text='6',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("6"))
b7=Button(root,text='7',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("7"))
b8=Button(root,text='8',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("8"))
b9=Button(root,text='9',padx=30,pady=20,borderwidth=2,command=lambda:set
TextInput("9"))

equal=Button(root,text='=',padx=30,pady=20,borderwidth=2,command=gettext)
clear=Button(root,text='Clear',padx=60,pady=20,borderwidth=2,command=clear
tex)
decimal=Button(root,text='.',padx=30,pady=20,borderwidth=2,command=lambd
a:setTextInput("."))
```

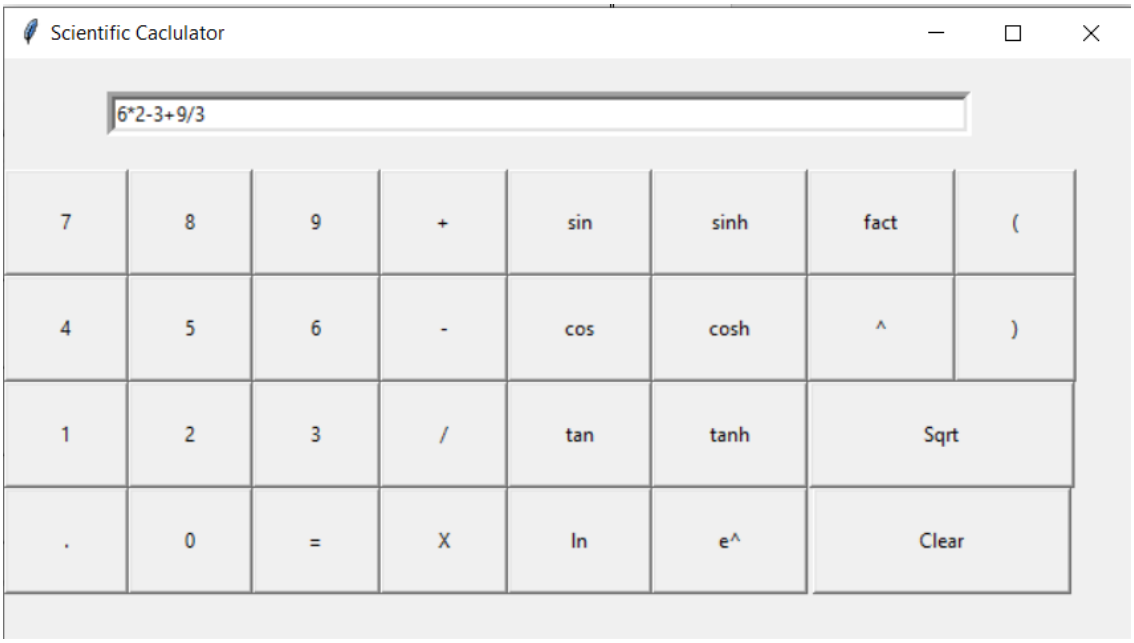
```
plus=Button(root,text='+',padx=30,pady=20,borderwidth=2,command=lambda:s
etTextInput("+"))
subtract=Button(root,text='-
',padx=30,pady=20,borderwidth=2,command=lambda:setTextInput("-"))
divide=Button(root,text='/',padx=30,pady=20,borderwidth=2,command=lambda:
setTextInput("/"))
multiply=Button(root,text='X',padx=30,pady=20,borderwidth=2,command=lamb
da:setTextInput("*"))
sin=Button(root,text='sin',padx=30,pady=20,borderwidth=2,command=lambda:s
etTextInput("sin("))
cos=Button(root,text='cos',padx=30,pady=20,borderwidth=2,command=lambda:
setTextInput("cos("))
tan=Button(root,text='tan',padx=30,pady=20,borderwidth=2,command=lambda:s
etTextInput("tan("))
log=Button(root,text='ln',padx=30,pady=20,borderwidth=2,command=lambda:se
tTextInput("log("))
sinh=Button(root,text='sinh',padx=30,pady=20,borderwidth=2,command=lambd
a:setTextInput("sinh("))
cosh=Button(root,text='cosh',padx=30,pady=20,borderwidth=2,command=lamb
da:setTextInput("cosh("))
tanh=Button(root,text='tanh',padx=30,pady=20,borderwidth=2,command=lambd
a:setTextInput("tanh("))
eraise=Button(root,text='e^',padx=30,pady=20,borderwidth=2,command=lambd
a:setTextInput("e("))
x_y=Button(root,text=' ^
',padx=30,pady=20,borderwidth=2,command=lambda:setTextInput("^"))
sqrt=Button(root,text='Sqrt',padx=65,pady=20,borderwidth=2,command=lambda
:setTextInput("rot("))
```

```
fact=Button(root,text='fact',padx=30,pady=20,borderwidth=2,command=lambda
:setTextInput("!"))
brack_ope=Button(root,text='(',padx=30,pady=20,borderwidth=2,command=lam
bda:setTextInput("("))
brack_clo=Button(root,text=')',padx=30,pady=20,borderwidth=2,command=lam
bda:setTextInput(")"))



b1.grid(row=3,column=0,sticky='nesw')
b2.grid(row=3,column=1,sticky='nesw')
b3.grid(row=3,column=2,sticky='nesw')
b4.grid(row=2,column=0,sticky='nesw')
b5.grid(row=2,column=1,sticky='nesw')
b6.grid(row=2,column=2,sticky='nesw')
b7.grid(row=1,column=0,sticky='nesw')
b8.grid(row=1,column=1,sticky='nesw')
b9.grid(row=1,column=2,sticky='nesw')
equal.grid(row=4,column=2,sticky='nesw')
b0.grid(row=4,column=1,sticky='nesw')

plus.grid(row=1,column=4,sticky='nesw')
subtract.grid(row=2,column=4,sticky='nesw')
divide.grid(row=3,column=4,sticky='nesw')
multiply.grid(row=4,column=4,sticky='nesw')

sin.grid(row=1,column=5,sticky='nesw')
cos.grid(row=2,column=5,sticky='nesw')
tan.grid(row=3,column=5,sticky='nesw')
log.grid(row=4,column=5,sticky='nesw')
```

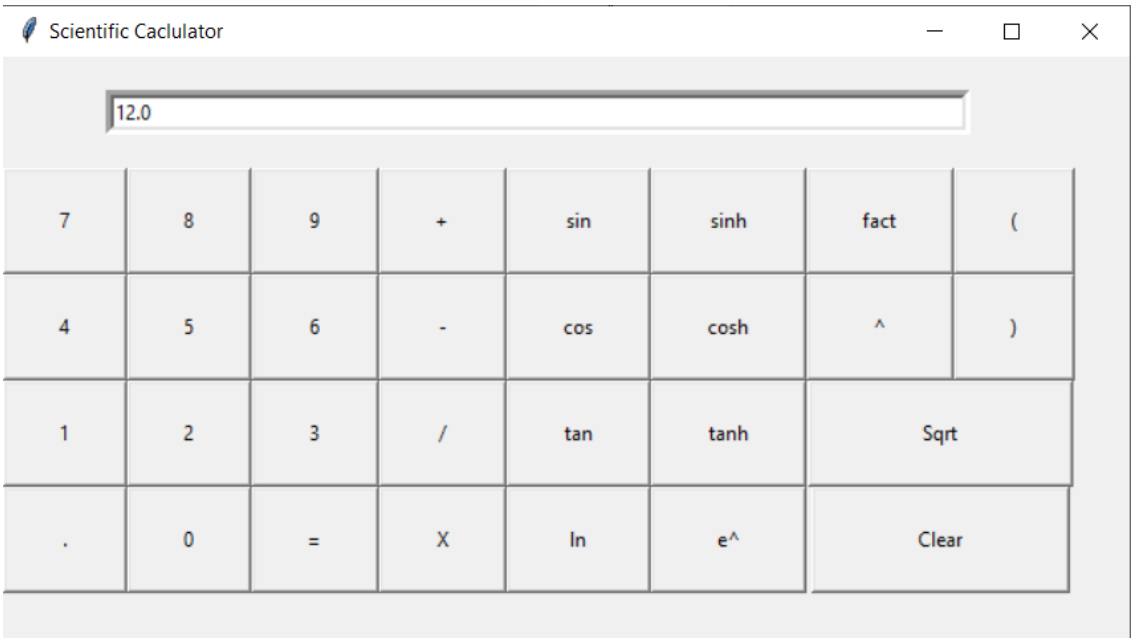
```
sinh.grid(row=1,column=6,sticky='nesw')
cosh.grid(row=2,column=6,sticky='nesw')
tanh.grid(row=3,column=6,sticky='nesw')
eraise.grid(row=4,column=6,sticky='nesw')

fact.grid(row=1,column=7,sticky='nesw')
x_y.grid(row=2,column=7,sticky='nesw')
sqrt.grid(row=3,column=7,columnspan=10)
decimal.grid(row=4,column=0,sticky='nesw')
clear.grid(row=4,column=7,columnspan=10)
brack_ope.grid(row=1,column=8,sticky='nesw')
brack_clo.grid(row=2,column=8,sticky='nesw')
root.mainloop()
```

Output:

Basic Math operations:

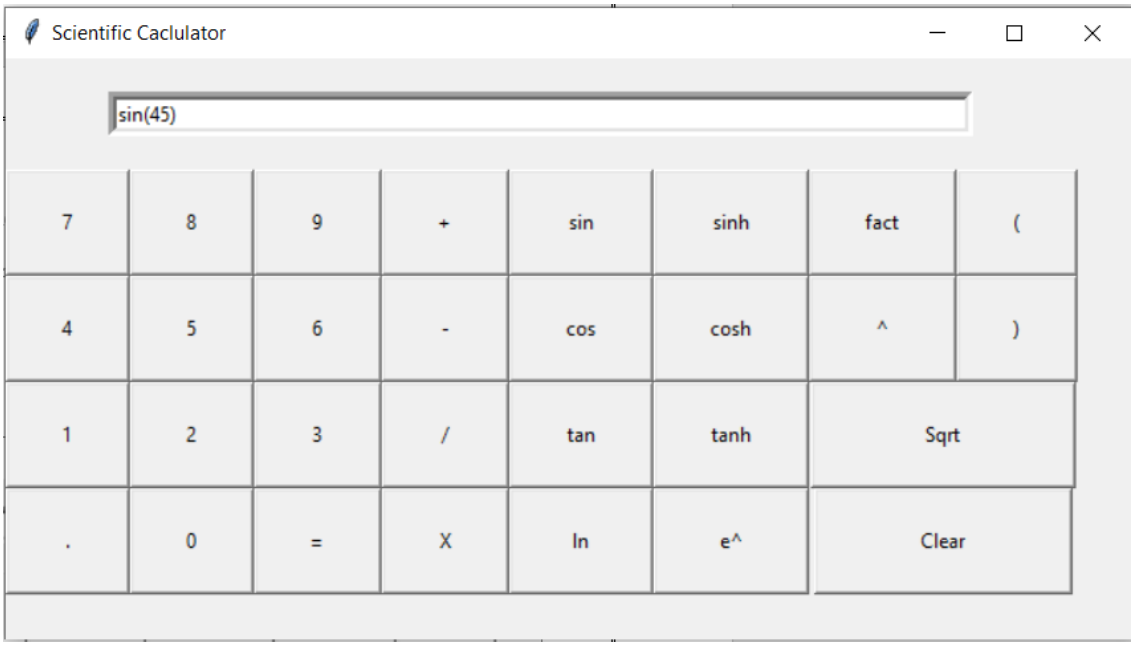


Result:

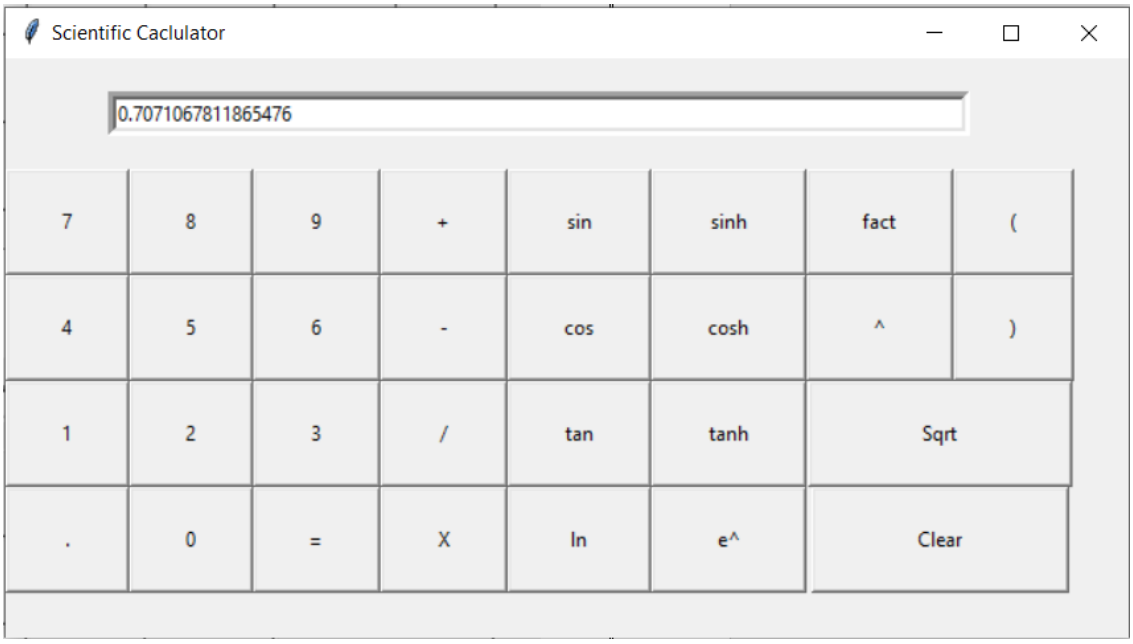


Trigonometric Functions:

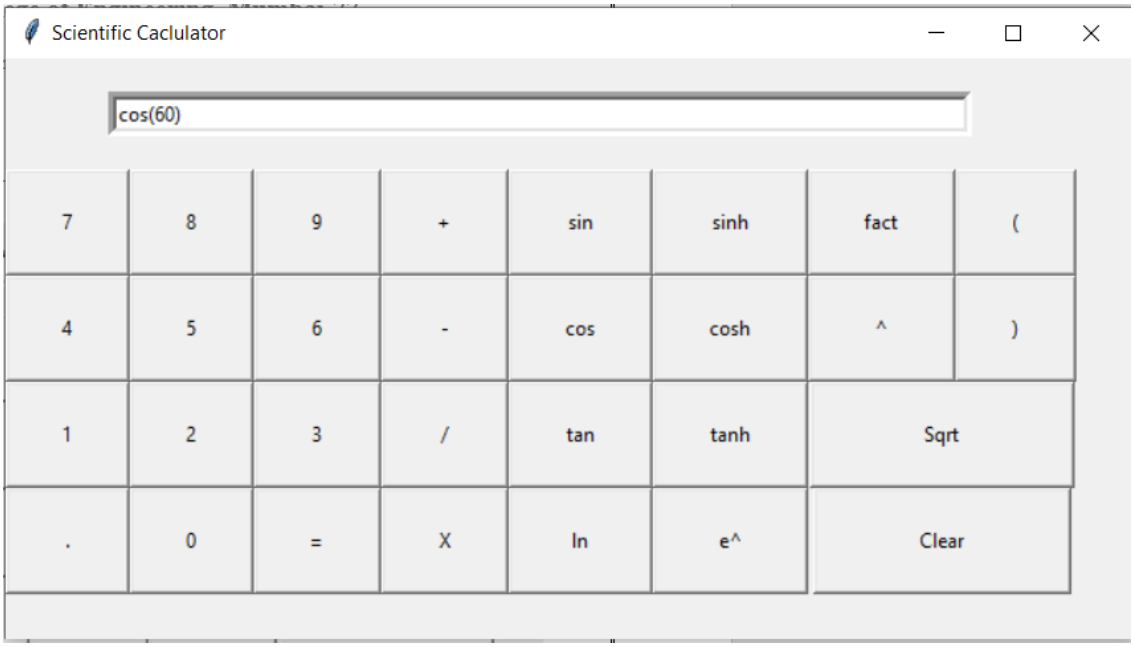
1) Sine function:



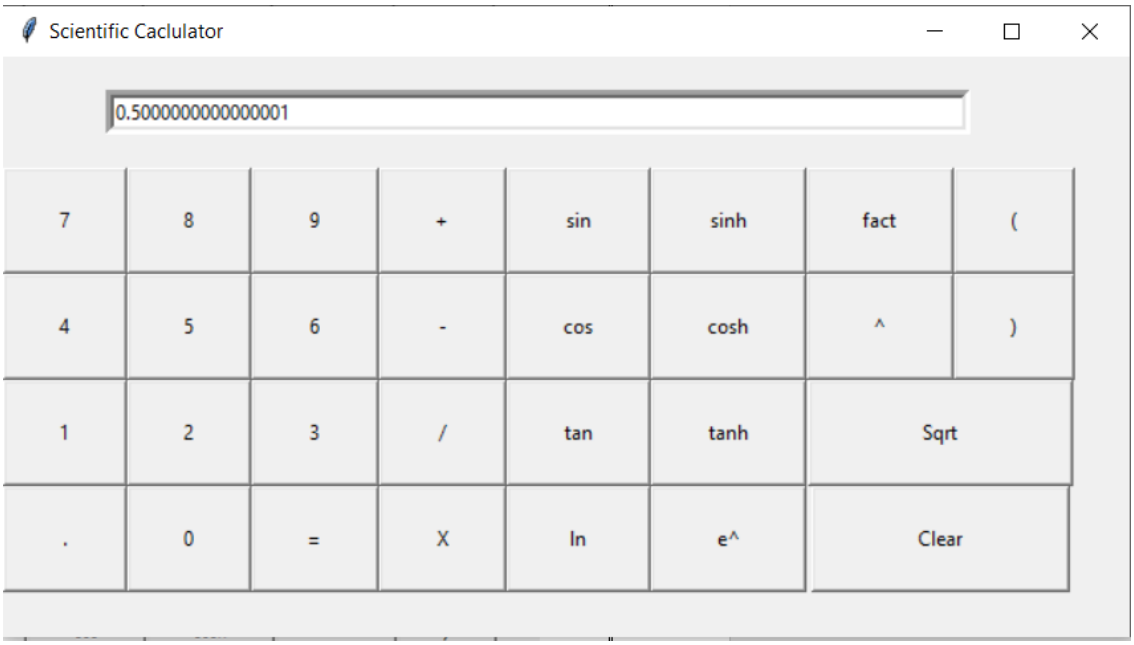
Result:



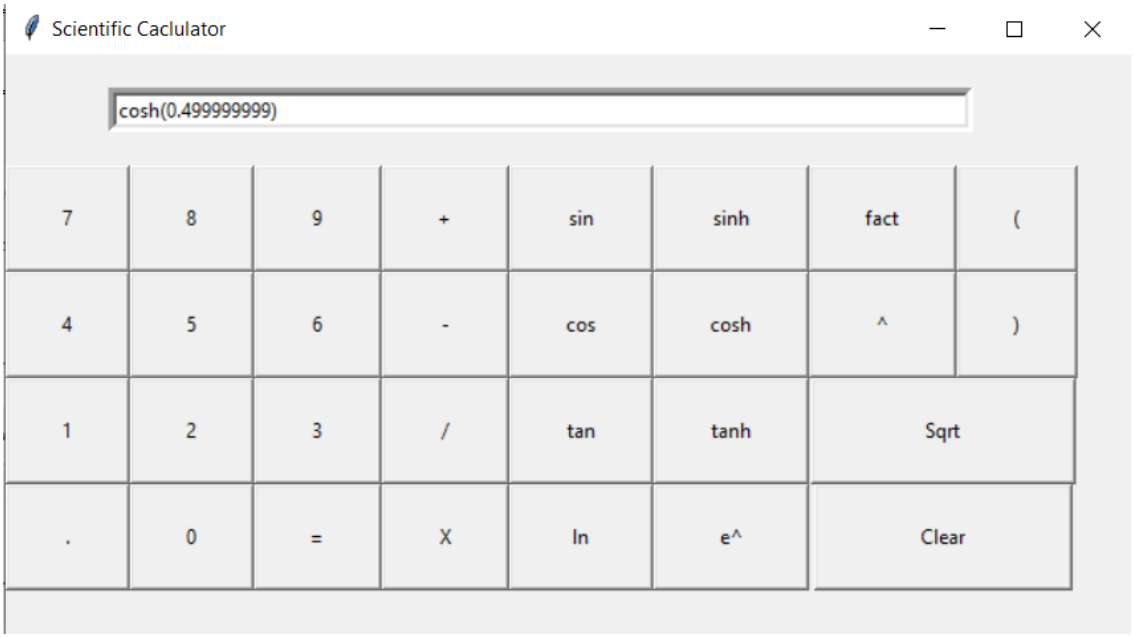
2) Cosine function:



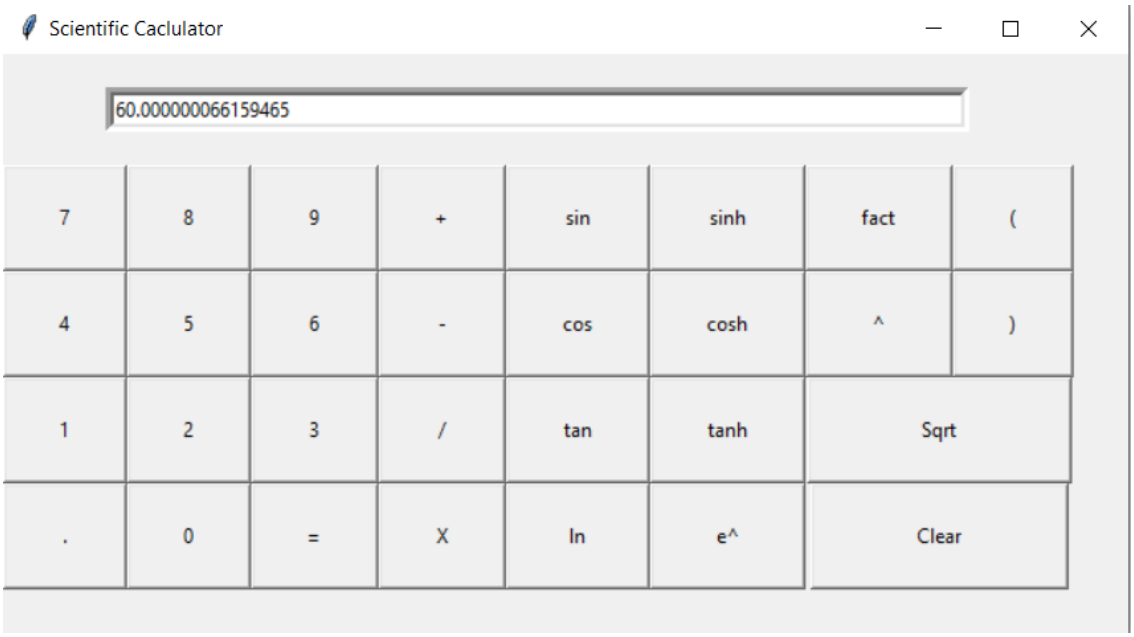
Result:



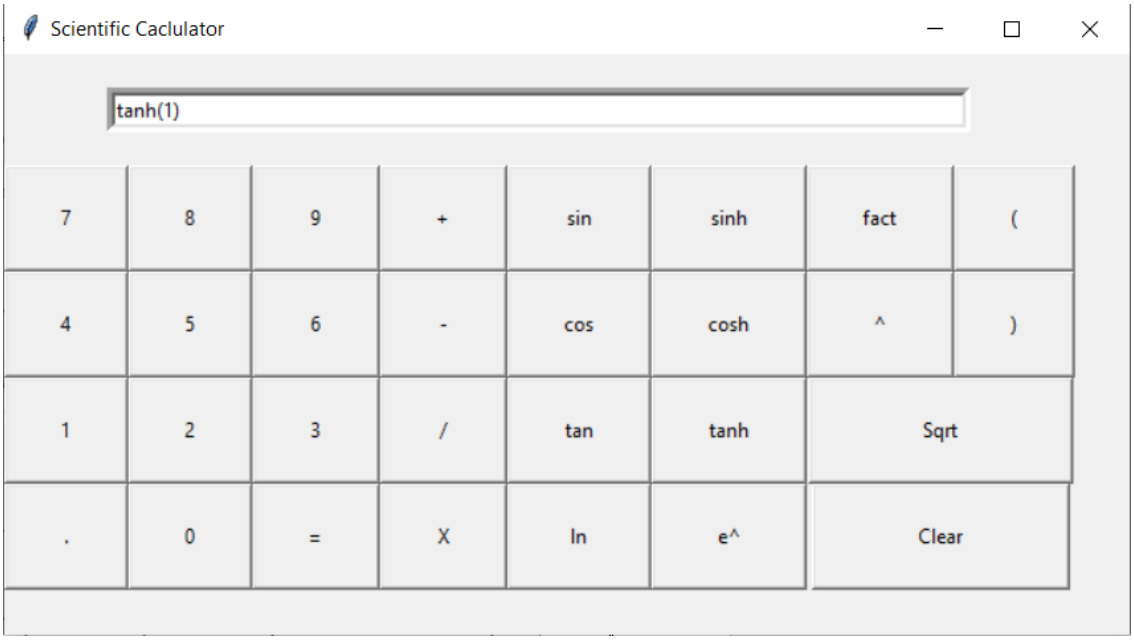
3) Cosine inverse function:



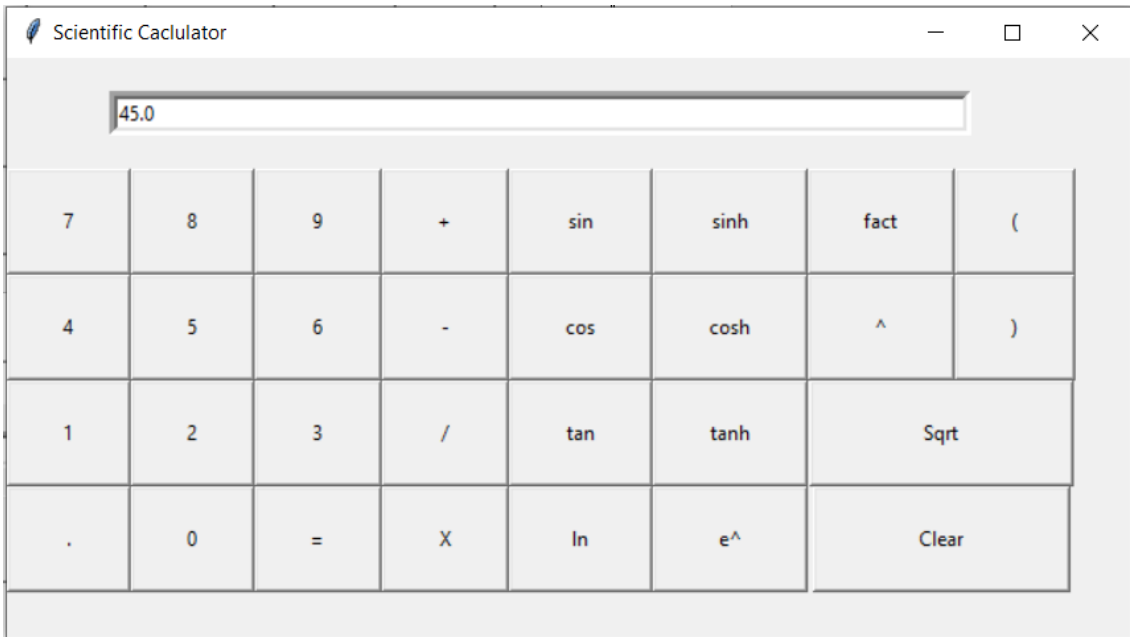
Result:



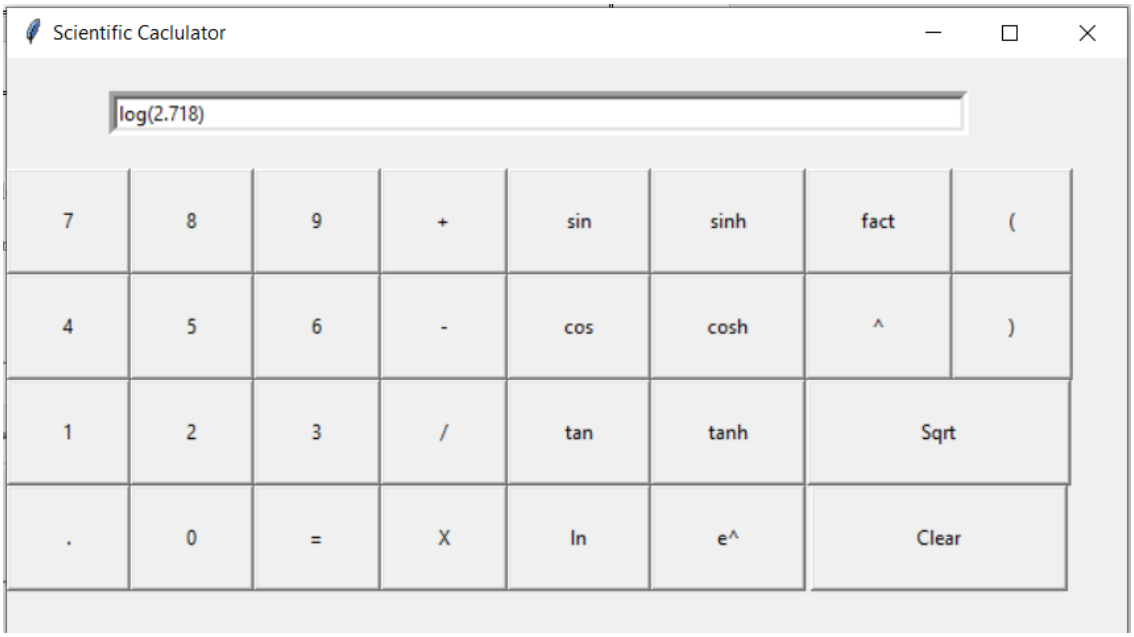
4) Tangent inverse function:



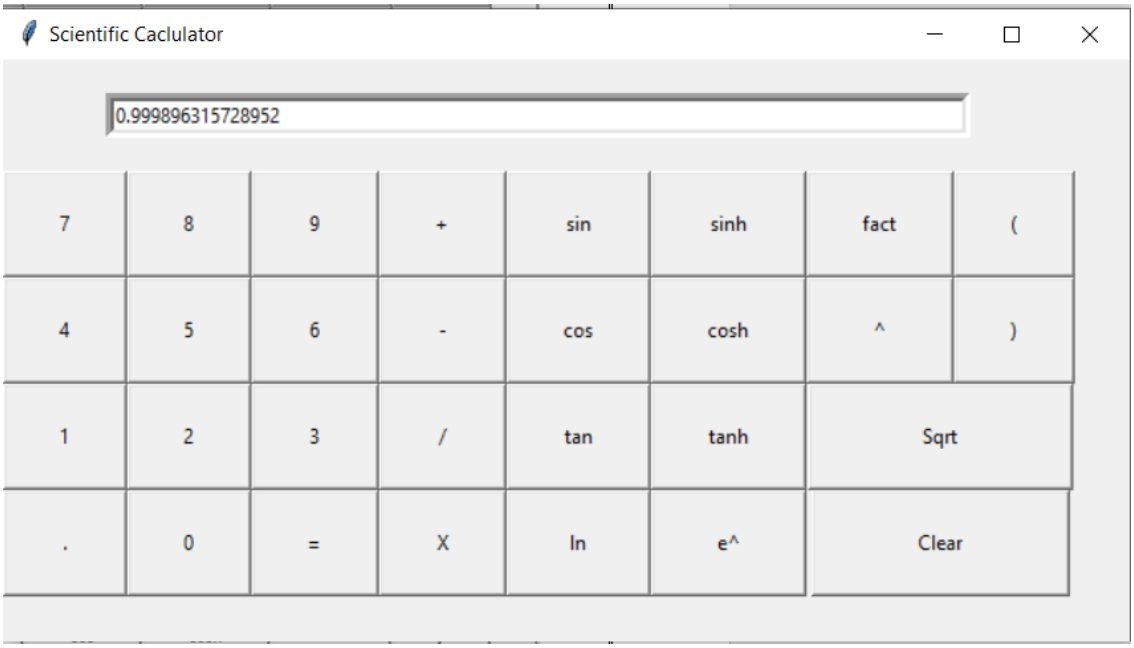
Result:



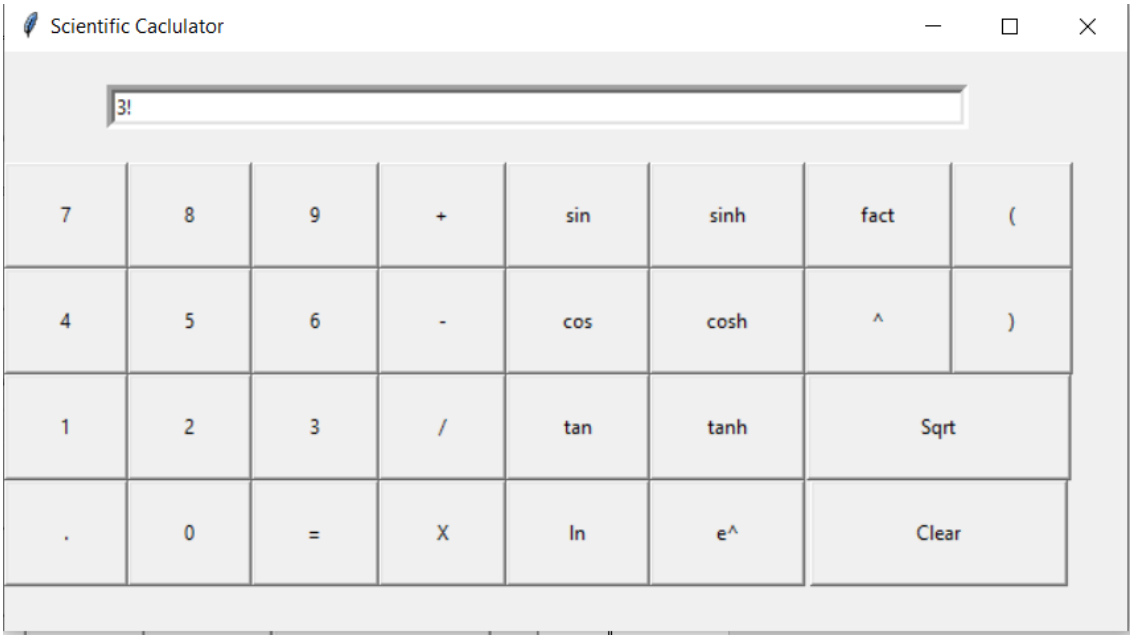
Logartithmic function:



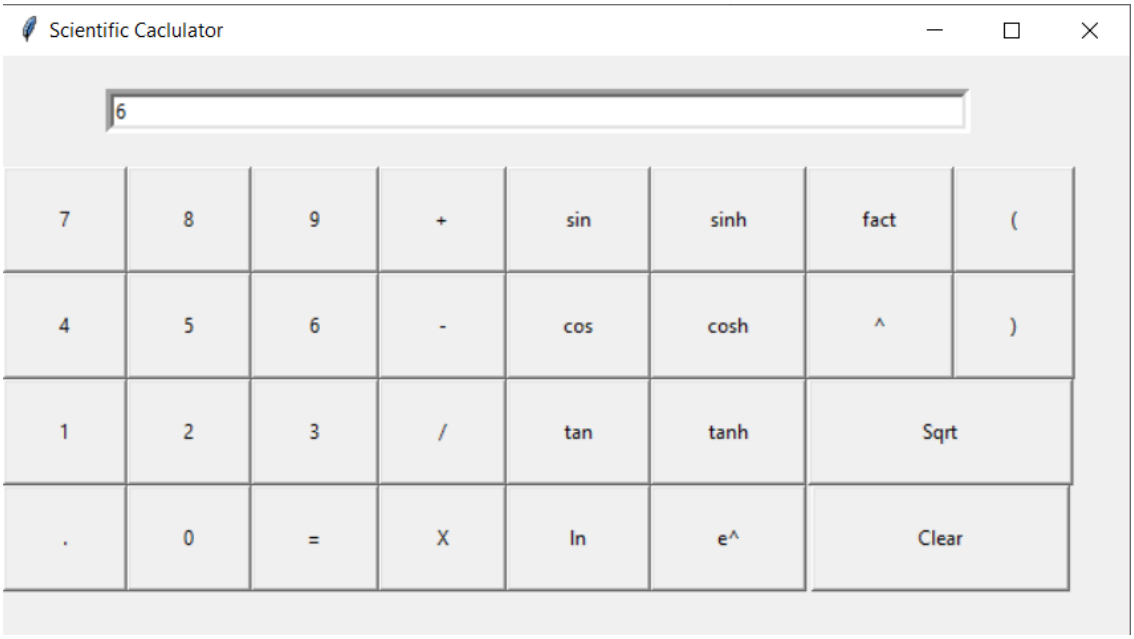
Result:



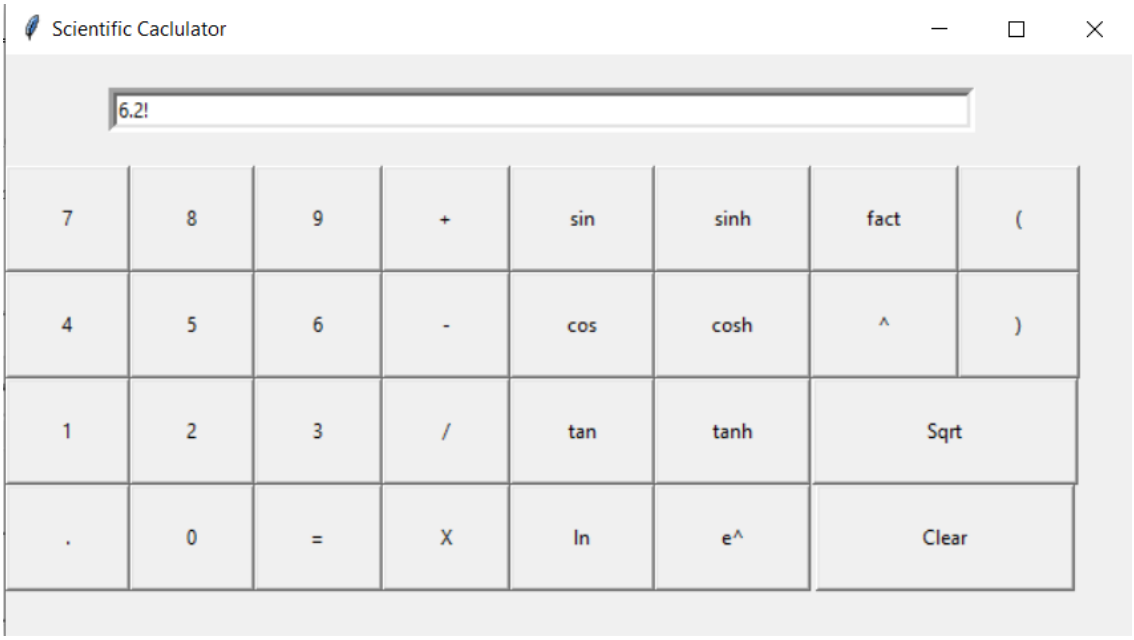
Factorial function:



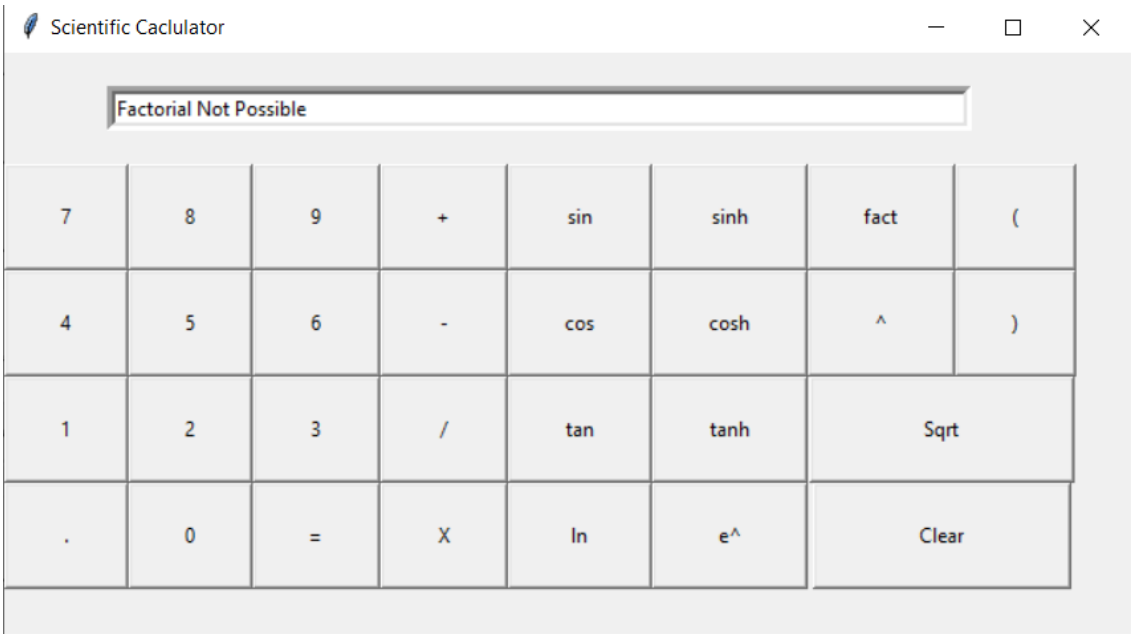
Result:



Invalid factorial operation:

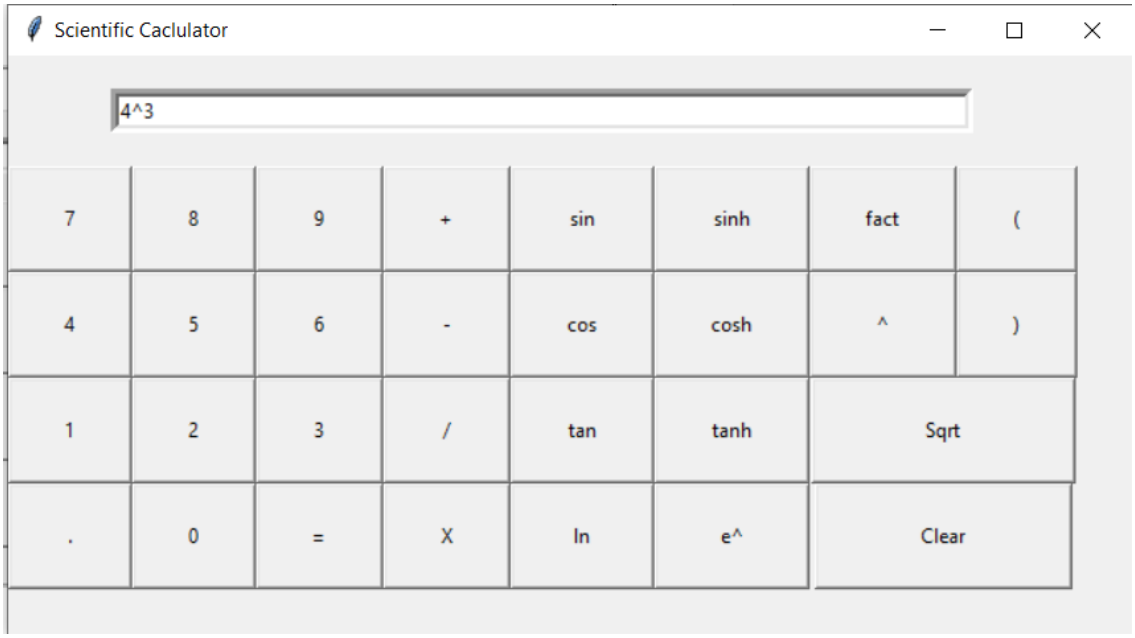


Result:

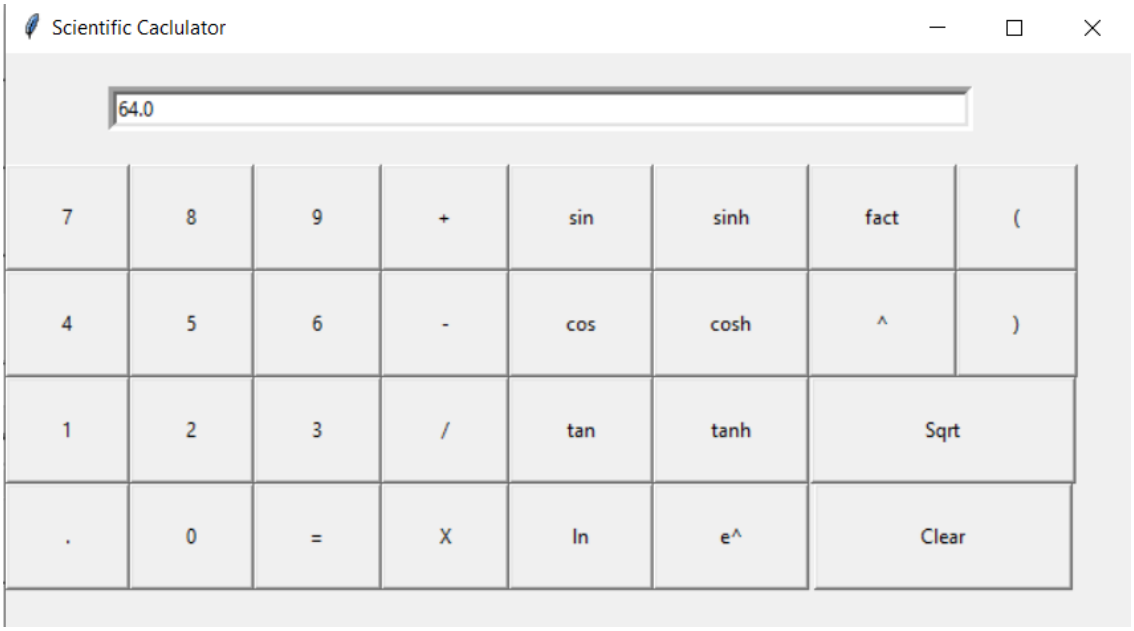


Exponential functions:

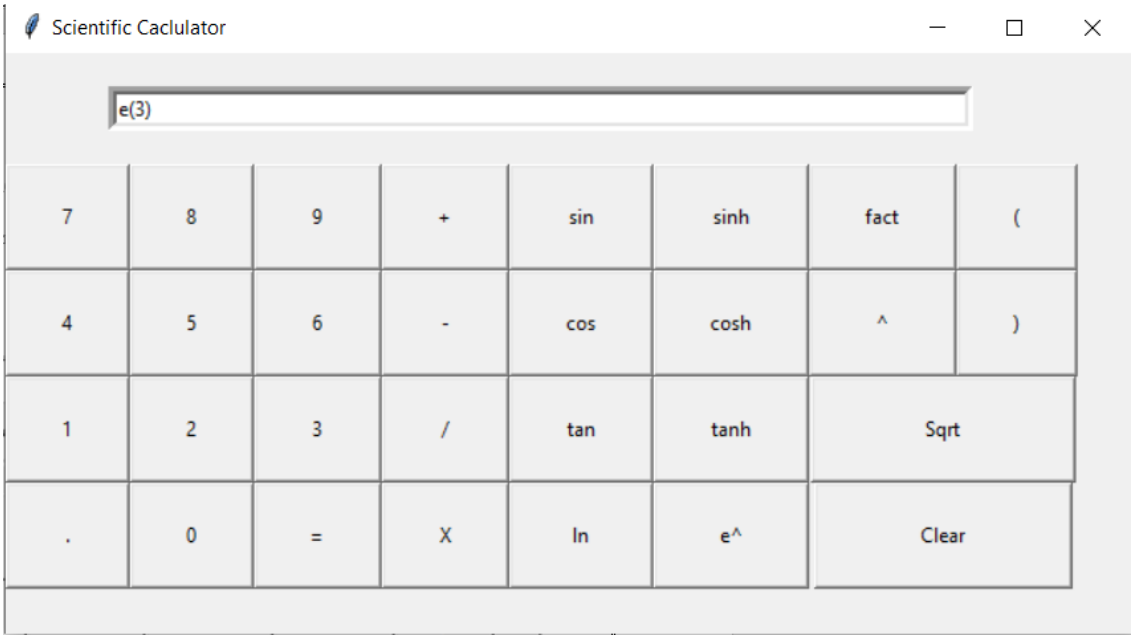
1) x^y:



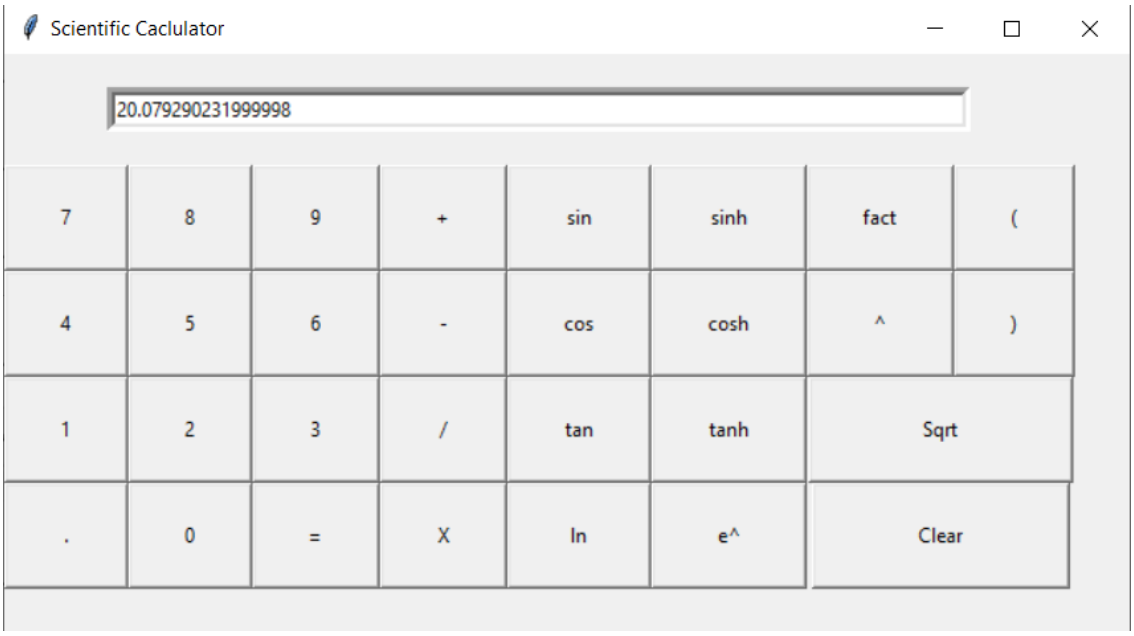
Result:



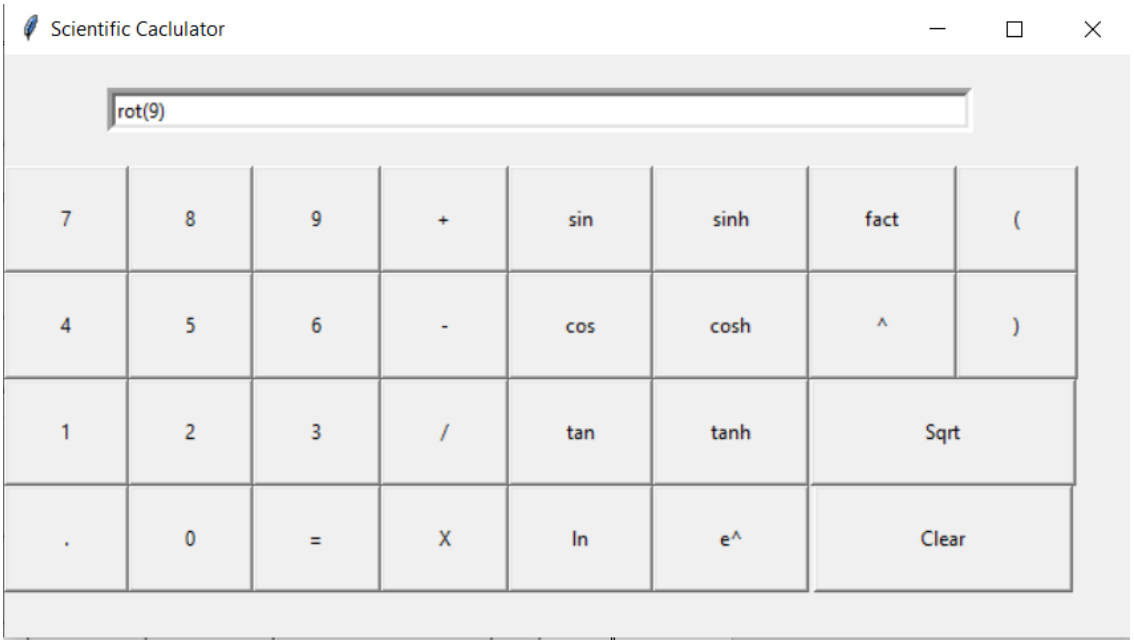
2) e^(x):


Scientific Caclulator
e(3)
7
8
9
+
sin
sinh
fact
(
4
5
6
-
cos
cosh
^
)
1
2
3
/
tan
tanh
Sqrt
.
0
=
X
ln
e^
Clear

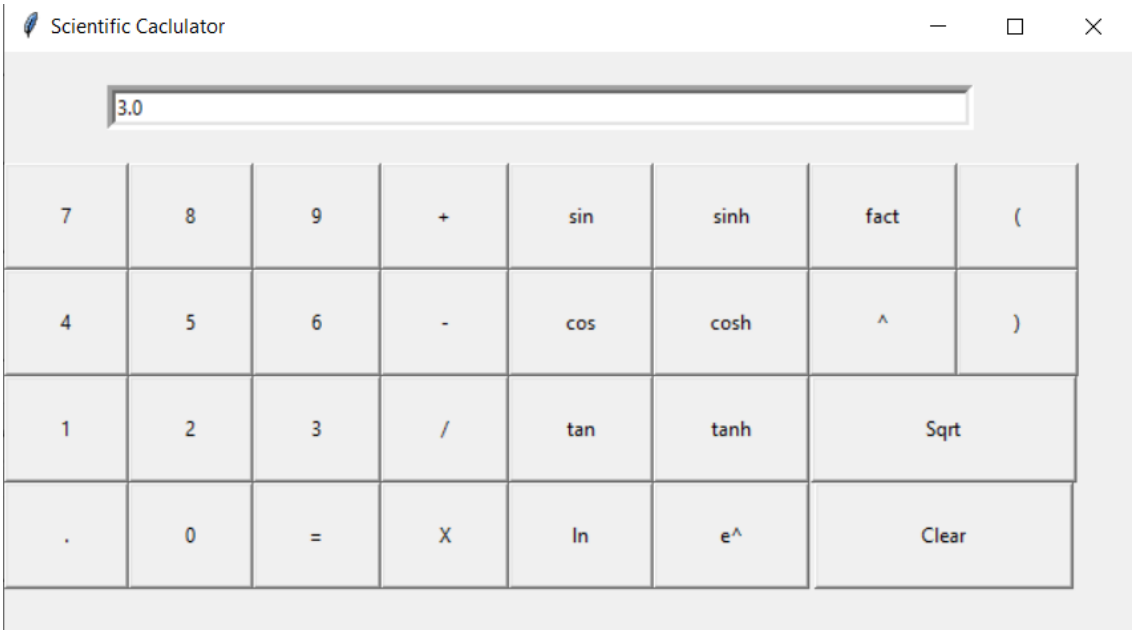
Result:


Scientific Caclulator
20.079290231999998
7
8
9
+
sin
sinh
fact
(
4
5
6
-
cos
cosh
^
)
1
2
3
/
tan
tanh
Sqrt
.
0
=
X
ln
e^
Clear

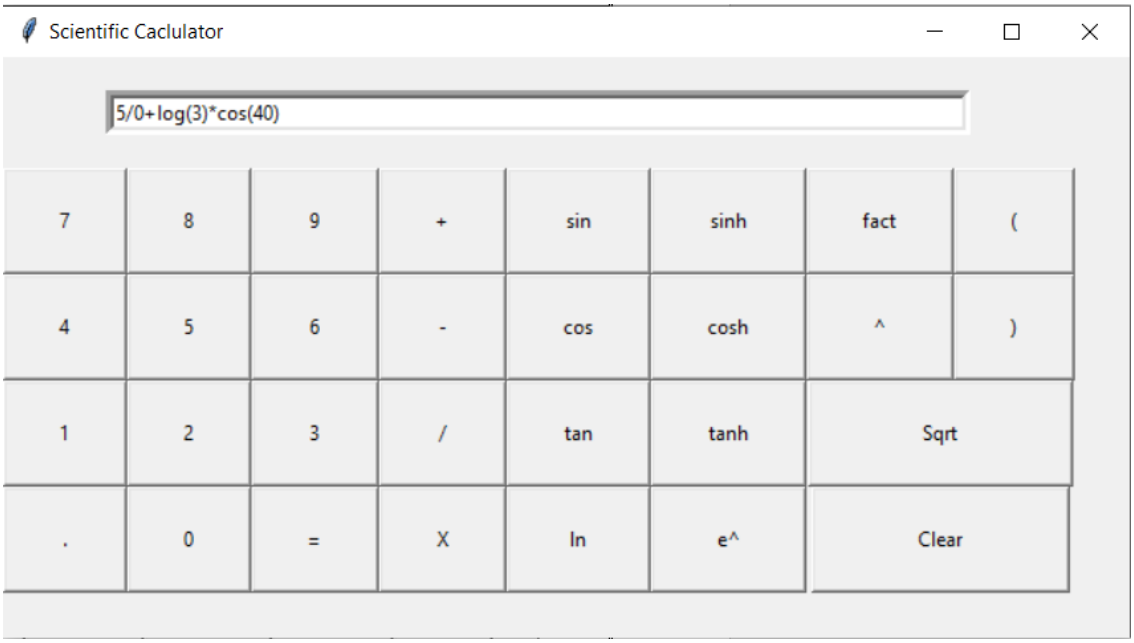
Square root function:



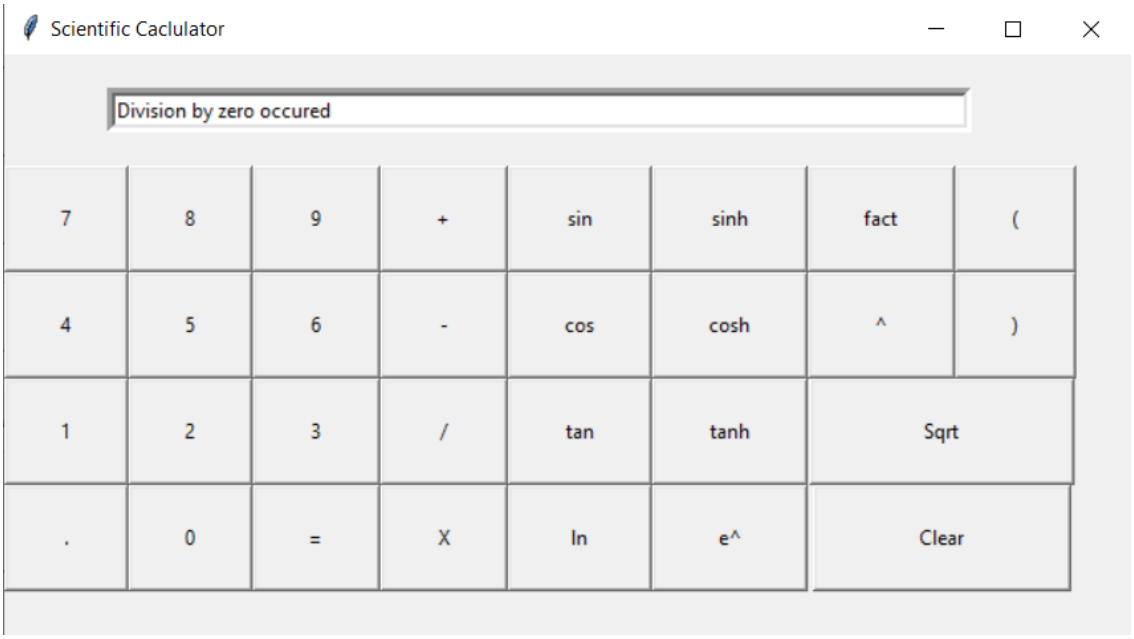
Result:



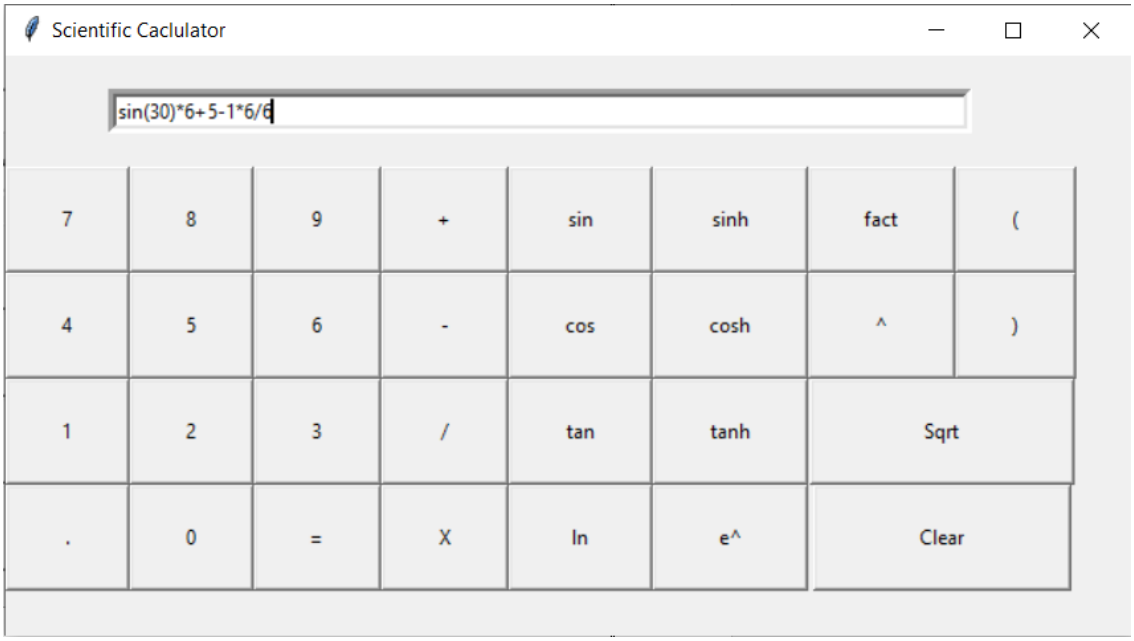
Complex operation 1:



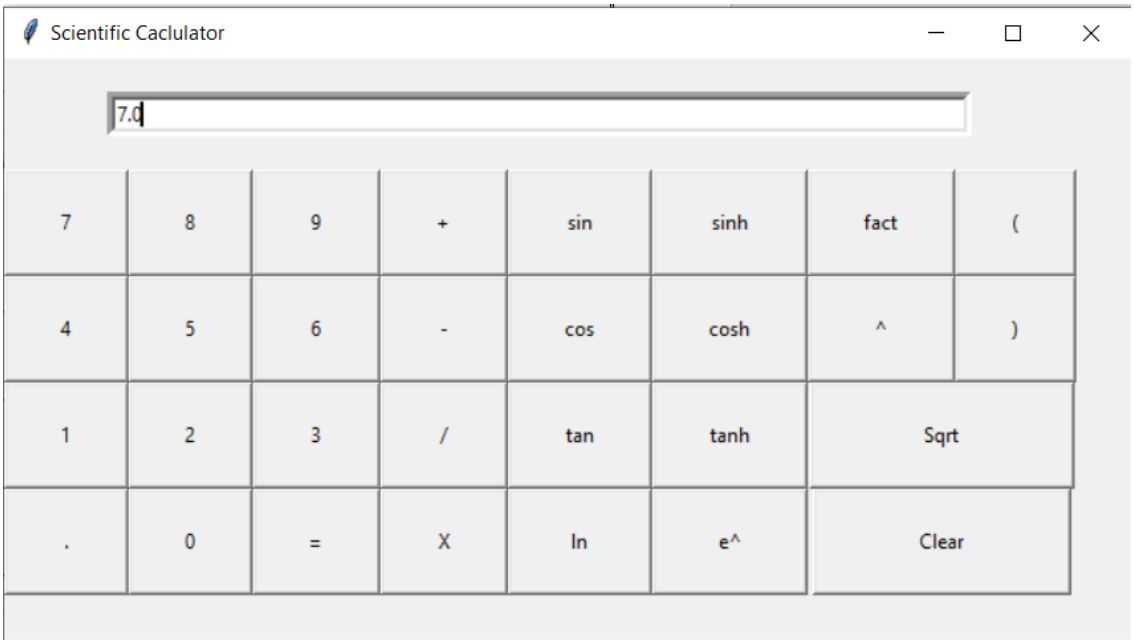
Result:



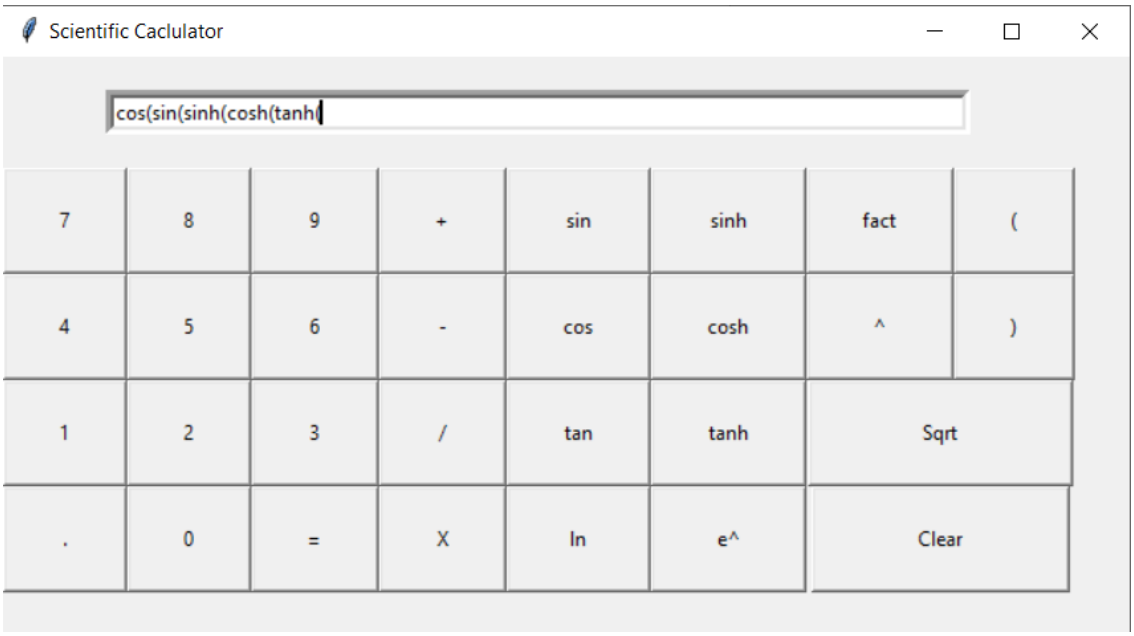
Complex operation 2:



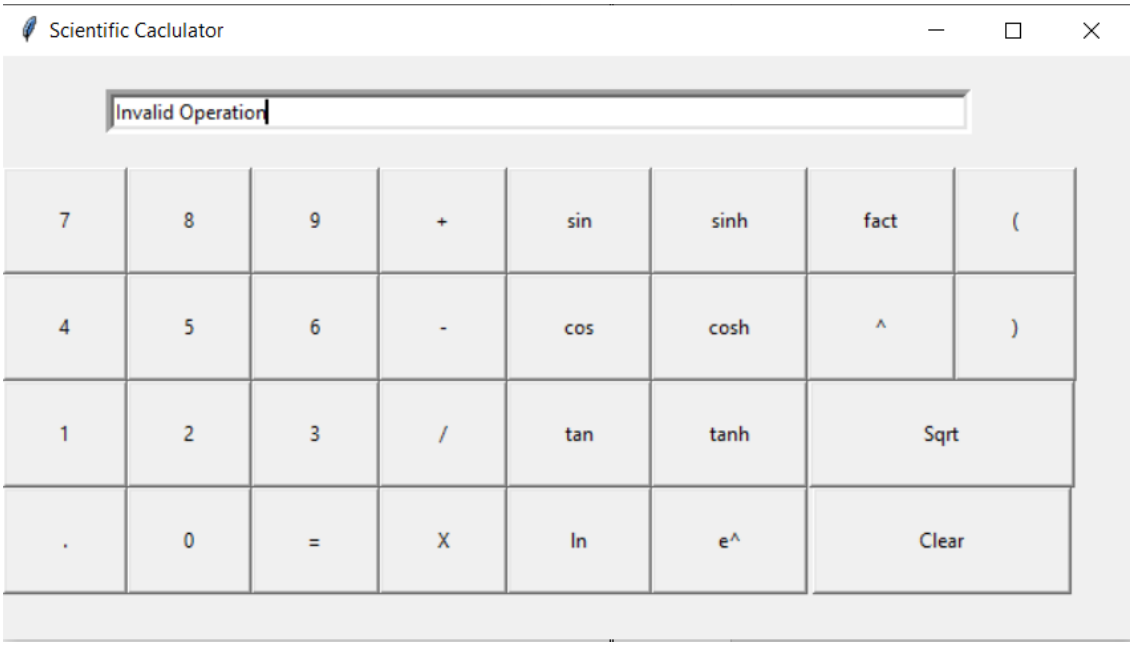
Result:



Invalid operation:



Result:



Conclusion: By performing this experiment completely understood how to make a GUI application using tkinter. Also gained knowledge about how calculators internally work. Used many functionalities of tkinter and made a scientific calculator.

Date: 15 / 4 / 2021 **Signature of faculty in-charge**

Post Lab Descriptive Questions

1. Config() in Python Tkinter are used for

(A) destroy the widget

(B) place the widget

(C) change property of the widget

(D) configure the widget

ANS) (C) change property of the widget

2. Correct way to draw a line in canvas tkinter ?

(A) line()

(B) canvas.create_line()

(C) create_line(canvas)

(D) None of the above

ANS) (B) canvas.create_line()

3. Essential thing to create a window screen using tkinter python?

(A) call tk() function

(B) create a button

(C) To define a geometry

(D) All of the above

ANS) (A) call tk() function

4. fg in tkinter widget is stands for ?

(A) foreground

(B) background

(C) forgap

(D) None of the above

ANS) (A) foreground

5. For user Entry data, which widget we use in tkinter ?

(A) Entry

(B) Text

(C) Both of the above

(D) None of the above

ANS) (B) Text

6. How pack() function works on tkinter widget ?

(A) According to x,y coordinate

(B) According to row and column vise

(C) According to left,right,up,down

(D) None of the above

ANS) (C) According to left,right,up,down

7. How the grid() function put the widget on the screen ?

(A) According to x,y coordinate

(B) According to row and column vise

(C) According to left,right,up,down

(D) None of the above

ANS) (B) According to row and column vise

8. Minimum number of argument we pass in a function to create a rectangle using canvas tkinter ?

(A) 2

(B) 4

(C) 6

(D) 5

ANS) (A) 2

9. Which of the following is clickable in GUI programming ?

(A) Button

(B) Checkbutton

(C) Lable

(D) 1 and 2

ANS) (D) 1 and 2

10. How we import a tkinter in python program ?

(A) import tkinter

(B) import tkinter as t

(C) from tkinter import *

(D) All of the above

ANS) (D) All of the above