APPENDIX – 1

SCIENTIFIC CALCULATOR

PROJECT REPORT

by

NAME OF THE CANDIDATE(S)

Name	Section	Roll Number
Akhil Vibhakar	K19PD	47
Rahul Singh	K19PD	24
Sonu Kumar	K19PD	32

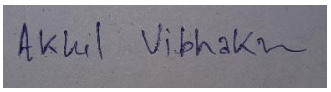
Department of Intelligent Systems
School of Computer Science
Lovely Professional University, Jalandhar

October – 2020

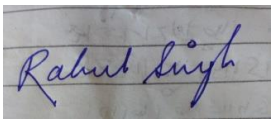
APPENDIX – 2

Student Declaration

This is to declare that this report has been written by me/us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. I/We aver that if any part of the report is found to be copied, I/we are shall take full responsibility for it.



Akhil Vibhakar
47



Rahul Singh
24

Sonu Kumar
32

Lovely Professional University, Jalandhar

24 Oct 2020

APPENDIX 3

TABLE OF CONTENTS

APPENDIX 4

BONAFIDE CERTIFICATE

Certified that this project report " **SCIENTIFIC CALCULATOR** " is the bonafide work of " **AKHIL VIBHAKAR** , **RAHUL SINGH** & **SONU KUMAR** " who carried out the project work under my supervision.

Ashish Shrivastava
Assistance Professor
25703
School of Computer and Engineering

Background and objectives of the project

Introduction:

A scientific calculator is a type of electronic calculator, usually but not always handheld, designed to calculate problems in science, engineering, and mathematics. They have completely replaced slide rules in traditional applications and are widely used in both education and professional settings.
In certain contexts, such as higher education, scientific calculators have been superseded by graphing calculators, which offer a superset of scientific calculator functionality along with the ability to graph input data and write and store programs for the device. There is also some overlap with the financial calculator market.
Scientific calculators are used widely in situations that require quick access to certain mathematical functions, especially those that were once looked up in mathematical tables, such as trigonometric functions or logarithms

Background:

Here by using basic tools of Python GUI (graphical user interface) we have made scientific calculator , plots of basic function ,.for understanding the basic function of the GUI and programming we have made the scientific calculator which will perform the basic function mathematics like addition, multiplication, subtraction , division and some more .which helped lots us to understand tools in in python and programming .
Here by understanding concepts of GUI we have applied to find out bode plot of any system to graphically means without going for programming and which will be easy for anybody to find bode plot of system. Then we have added that these two functions by using graphical method. Main menu is a combination of all function through which we can go to all for operation.

Motivation:

We have studied the ‘Signals and system’ and Control System then we came across many terms like bode plot, impulse response, step response, convolution of two signals. We have implemented this all in the very wonderful matrix laboratory tool python, and then we thought of making it very user friendly so that without knowing the programming for the function one can use it. For that we used the python GUI tool to make it user friendly in this way we come up with this project.

.

Outcomes of the project:

1. we get familiar with python programming language.
2. learnt to implement various mathematical operations in python language.
3. learnt to implement python language to make GUI (graphical user interface)

Description of Project

A **SCIENTIFIC CALCULATOR** is a type of electronic calculator, usually but not always handheld, designed to calculate problems in science, engineering, and mathematics. They have completely replaced slide rules in traditional applications and are widely used in both education and professional settings.

We have divided this project into following 3 modules:

Module 1:

This module will contain all the coding for the basic layout of the scientific calculator or the basic GUI of the scientific calculator. This module will include all the physical buttons which is needed in scientific calculator like addition button, subtraction button, sine function button, cos function button and many more. This module will be developed by only 1 student with the help of tkinter library and basic python only and further all the physical buttons will be linked with their respective function in the other module of this project**.**

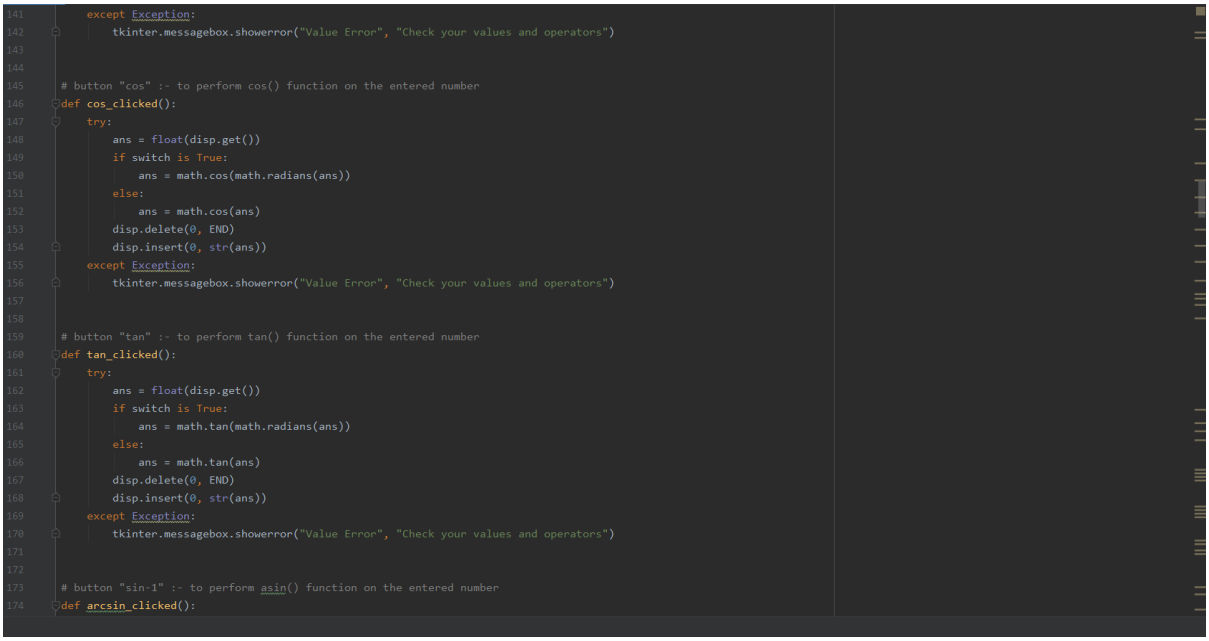
Screenshot:

```
btnrow1 = Frame(root, bg="#000000")
btnrow1.pack(expand=TRUE, fill=BOTH)

pi_btn = Button(btnrow1, text="π", font="Times 18", relief=GROOVE, bd=0, command=pi_clicked, fg="#43FFF6", bg="black")
pi_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

fact_btn = Button(btnrow1, text=" x! ", font="Times 18", relief=GROOVE, bd=0, command=fact_clicked, fg="#43FFF6", bg="black")
fact_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sin_btn = Button(btnrow1, text="sin", font="Times 18", relief=GROOVE, bd=0, command=sin_clicked, fg="#43FFF6", bg="black")
sin_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

cos_btn = Button(btnrow1, text="cos", font="Times 18", relief=GROOVE, bd=0, command=cos_clicked, fg="#43FFF6", bg="black")
cos_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

tan_btn = Button(btnrow1, text="tan", font="Times 18", relief=GROOVE, bd=0, command=tan_clicked, fg="#43FFF6", bg="black")
tan_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn1 = Button(btnrow1, text="1", font="Times 23", relief=GROOVE, bd=0, command=btn1_clicked, fg="#43FFF6", bg="black")
btn1.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn2 = Button(btnrow1, text="2", font="Times 23", relief=GROOVE, bd=0,  command=btn2_clicked, fg="#43FFF6", bg="black")
btn2.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
btn3.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnp = Button(btnrow1, text="+", font="Times 23", relief=GROOVE, bd=0, command=btnp_clicked, fg="#43FFF6", bg="black")
btnp.pack(side=LEFT, expand=TRUE, fill=BOTH)

# Row 2 Buttons

btnrow2 = Frame(root)
btnrow2.pack(expand=TRUE, fill=BOTH)
```

Module 2

This module will contain all the coding for all the mathematical functions needed for scientific calculator like addition function, subtraction function, sine function, cos function and many more functions. This module will be developed by only 1 student with the help of basic python only and further all the functions will be linked with their respective physical buttons in the other module of this project.

Screenshot:

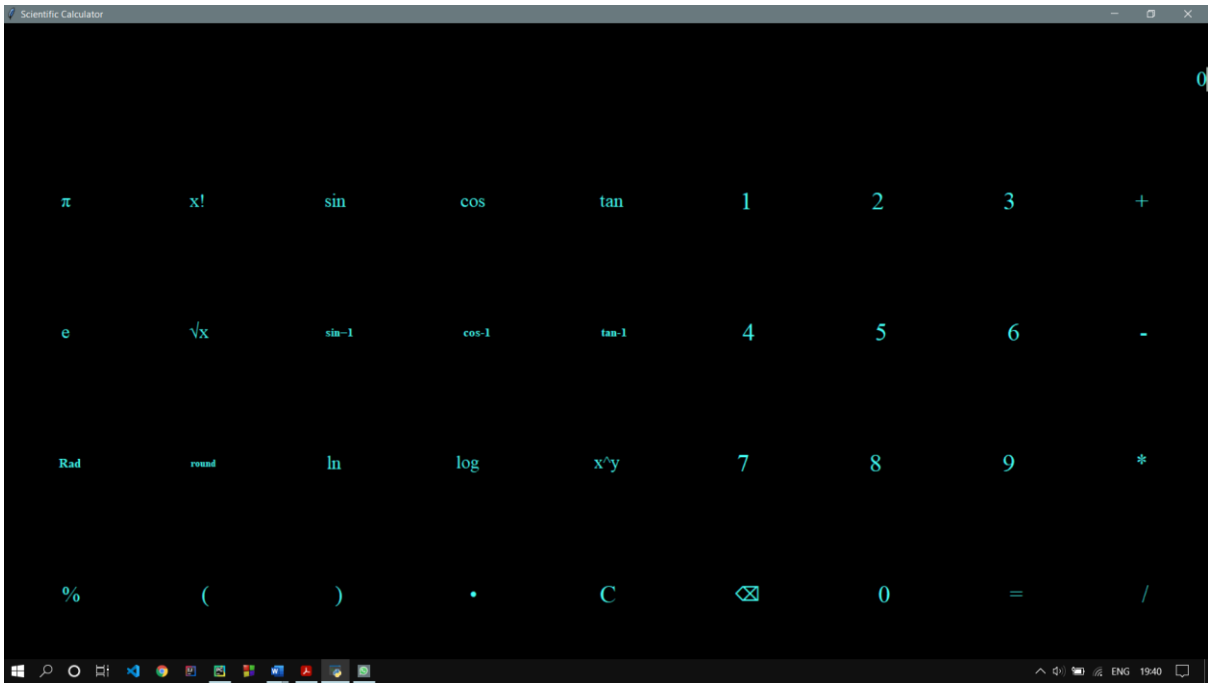


```
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "cos" :- to perform cos() function on the entered number
def cos_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.cos(math.radians(ans))
        else:
            ans = math.cos(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "tan" :- to perform tan() function on the entered number
def tan_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.tan(math.radians(ans))
        else:
            ans = math.tan(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "sin-1" :- to perform asin() function on the entered number
def arcsin_clicked():
```

Module 3:

This module is just the product of the first and the second module. In this module we will link all the physical buttons with their respective mathematical functions. This module will be developed by only 1 student with the help of tkinter library and basic python only and further after some testing we will have a working scientific calculator.

Screenshot:



Description of Work Division in terms of Roles among Students.

Module – 1 (Name – Rahul Singh; Roll No - 24; Reg No- 11905949)

This module is done by **Rahul Singh.** He is responsible for all the coding for the basic layout of the scientific calculator or the basic GUI of the scientific calculator and for all the physical buttons which is needed in scientific calculator like addition button, subtraction button, sine function button, cos function button and many more. He used tkinter library and basic python only and further he handed over the codes to **Akhil Vibhakar.**

Module – 2 (Name – Sonu Kumar; Roll No - 32; Reg No- 11905982)

This module is done by **Sonu Kumar**. He is responsible for all the coding of all the mathematical functions needed for scientific calculator like addition function, subtraction function, sine function, cos function and many more functions. He used basic python only and further he handed over the codes to **Akhil Vibhakar.**

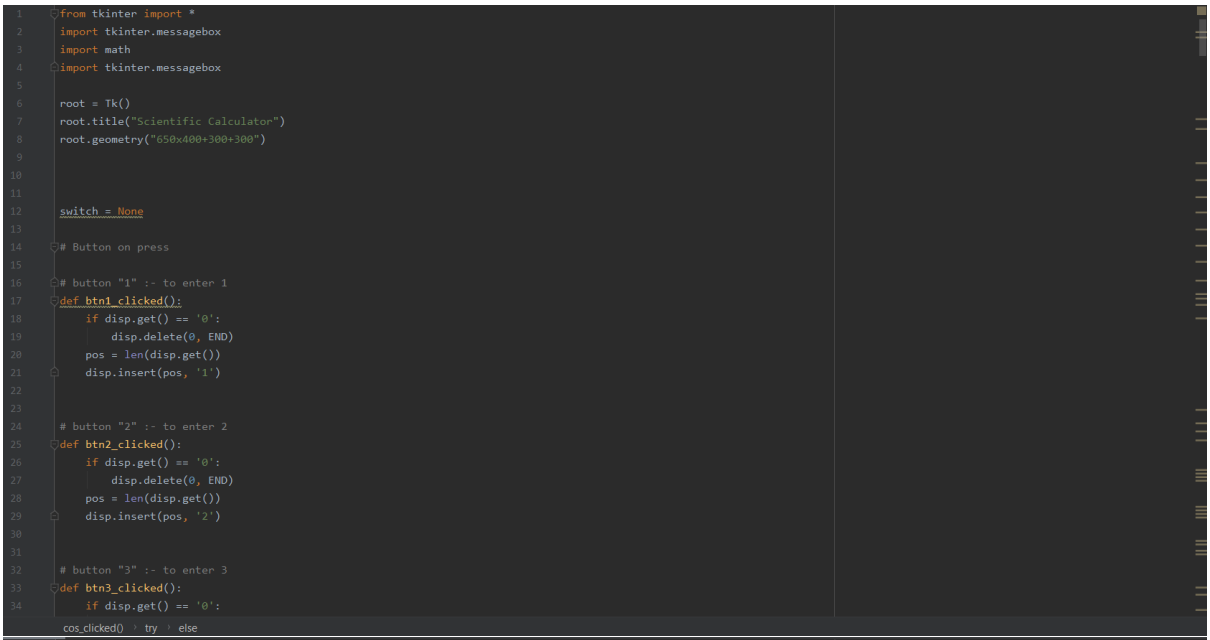
Module – 3 (Name – Akhil Vibhakar; Roll No - 47; Reg No- 11910509)

This module is done by **Akhil Vibhakar** .He is responsible for linking all the physical buttons with their respective mathematical functions . He used tkinter library and basic python only and he is responsible for testing also which he did at the end of the project.

Implementation of scheduled work of Project

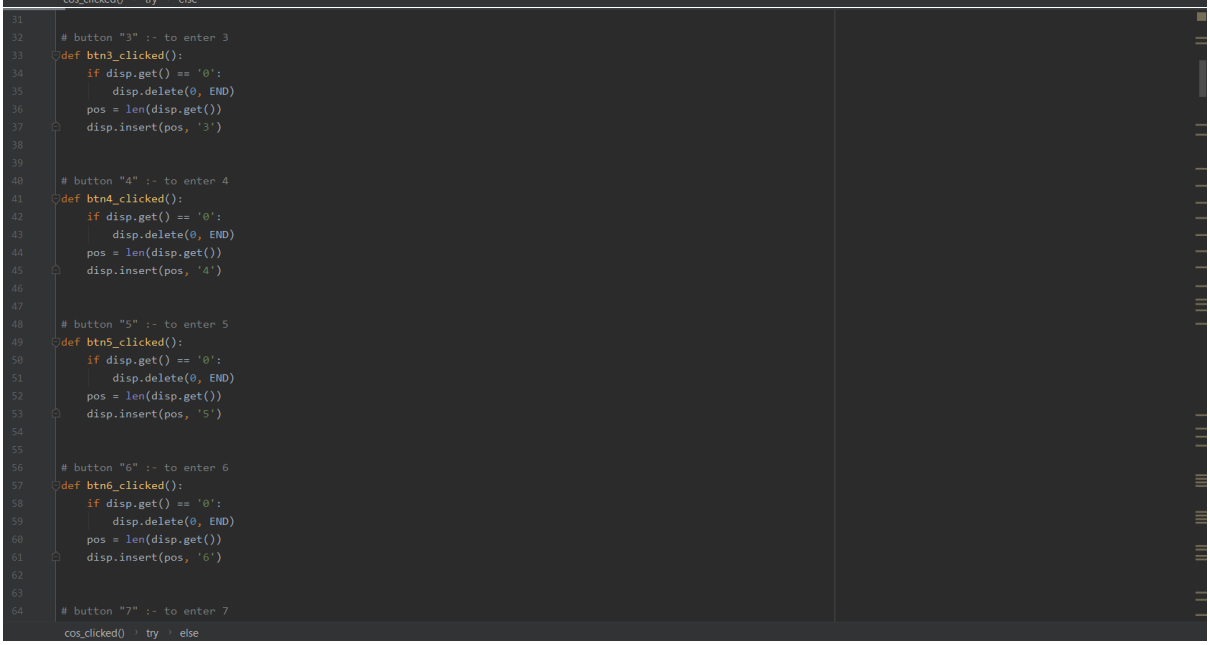
So the project was divided into 3 module on which 1st module was all about the coding of the GUI and the basic layout. 2nd module was all about the coding of the mathematical functions and in 3 module we just linked module1 with module2 and did testing.

CODES:



```
from tkinter import *
import tkinter.messagebox
import math
import tkinter.messagebox

root = Tk()
root.title("Scientific Calculator")
root.geometry("650x400+300+300")



switch = None

# Button on press

# button "1" :- to enter 1
def btn1_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '1')


# button "2" :- to enter 2
def btn2_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '2')


# button "3" :- to enter 3
def btn3_clicked():
    if disp.get() == '0':
```

cos_clicked() › try › else



```

# button "3" :- to enter 3
def btn3_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '3')


# button "4" :- to enter 4
def btn4_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '4')


# button "5" :- to enter 5
def btn5_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '5')


# button "6" :- to enter 6
def btn6_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '6')


# button "7" :- to enter 7
```

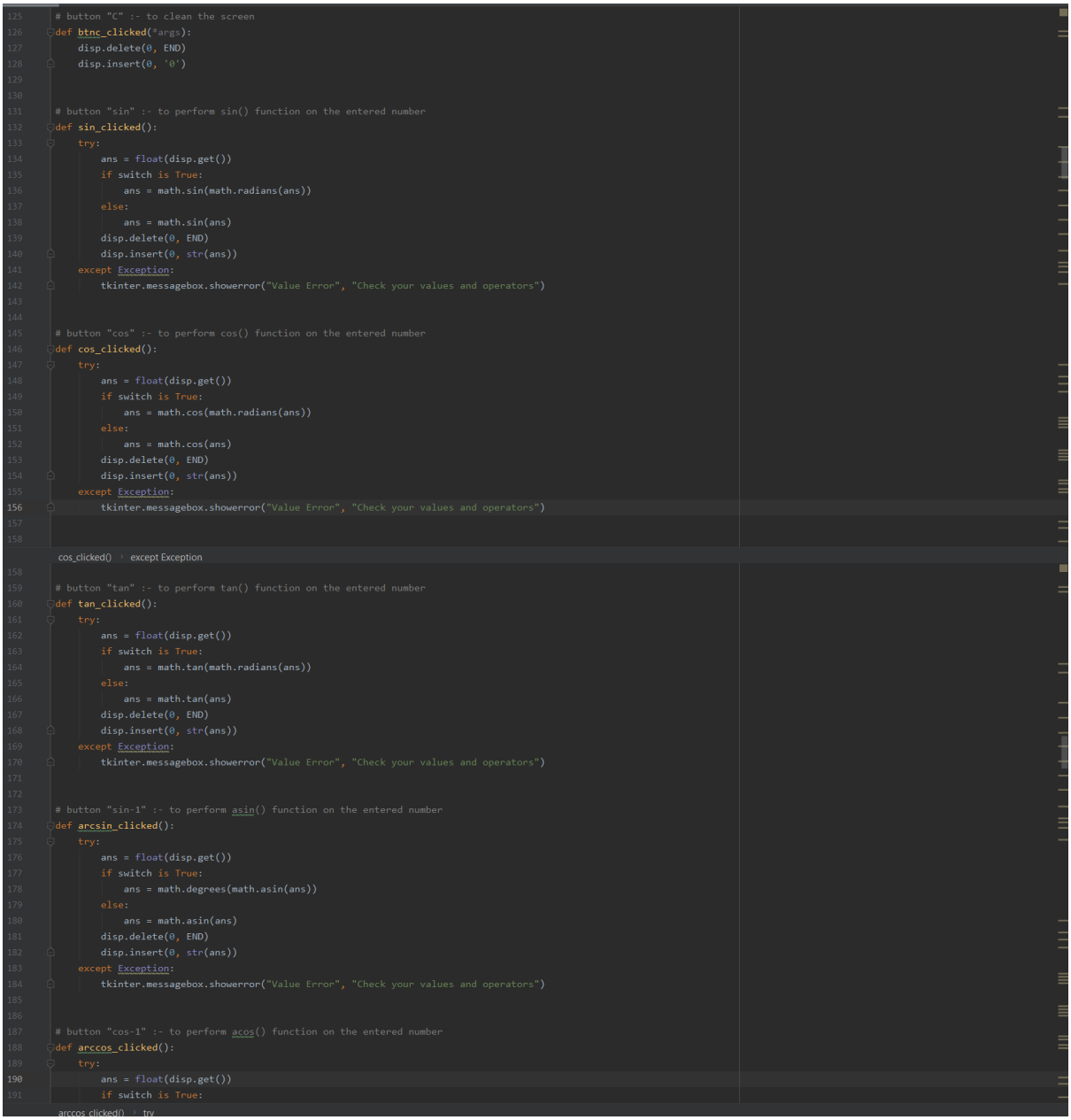
cos_clicked() › try › else

```
# button "7" :- to enter 7
def btn7_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '7')


# button "8" :- to enter 8
def btn8_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '8')


# button "9" :- to enter 9
def btn9_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '9')

# button "0" :- to enter 0
def btn0_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, '0')


# function for entering the numeric characters and point(.)
def key_event(*args):
    if disp.get() == '0':
```

key_event()

```
# function for entering the numeric characters and point(.)
def key_event(*args):
    if disp.get() == '0':
        disp.delete(0, END)


# button "+" :- to perform add operation
def btnp_clicked():
    pos = len(disp.get())
    disp.insert(pos, '+')


# button "-" :- to perform subtraction operation
def btnm_clicked():
    pos = len(disp.get())
    disp.insert(pos, '-')


# button "*" :- to perform multiplication operation
def btnml_clicked():
    pos = len(disp.get())
    disp.insert(pos, '*')


# button "/" :- to perform division operation
def btnd_clicked():
    pos = len(disp.get())
    disp.insert(pos, '/')


# button "C" :- to clean the screen
def btnc_clicked(*args):
    disp.delete(0, END)
```

btnc_clicked()



```
# button "C" :- to clean the screen
def btnc_clicked(*args):
    disp.delete(0, END)
    disp.insert(0, '0')


# button "sin" :- to perform sin() function on the entered number
def sin_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.sin(math.radians(ans))
        else:
            ans = math.sin(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "cos" :- to perform cos() function on the entered number
def cos_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.cos(math.radians(ans))
        else:
            ans = math.cos(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


cos_clicked() › except Exception

# button "tan" :- to perform tan() function on the entered number
def tan_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.tan(math.radians(ans))
        else:
            ans = math.tan(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "sin-1" :- to perform asin() function on the entered number
def arcsin_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.degrees(math.asin(ans))
        else:
            ans = math.asin(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "cos-1" :- to perform acos() function on the entered number
def arccos_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
arccos_clicked() › try
```

```
# button "cos-1" :- to perform acos() function on the entered number
def arccos_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.degrees(math.acos(ans))
        else:
            ans = math.acos(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "tan-1" :- to perform atan() function on the entered number
def arctan_clicked():
    try:
        ans = float(disp.get())
        if switch is True:
            ans = math.degrees(math.atan(ans))
        else:
            ans = math.atan(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "x^y" :- to perform y times x operation on the entered number x(say)
def pow_clicked():
    pos = len(disp.get())
    disp.insert(pos, '**')
```

pow_clicked()

```

# button "round" :- to roundoff the entered number
def round_clicked():
    try:
        ans = float(disp.get())
        ans = round(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "log" :- to perform log() function on the entered number
def logarithm_clicked():
    try:
        ans = float(disp.get())
        ans = math.log10(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")

# button "x!" :- to perform factorial operation on the entered number x(say)
def fact_clicked():
    try:
        ans = float(disp.get())
        ans = math.factorial(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "√x" :- to perform sqare root operation on the entered number x(say)
```

```
# button "√x" :- to perform sqare root operation on the entered number x(say)
def sqr_clicked():
    try:
        ans = float(disp.get())
        ans = math.sqrt(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "dot(.)" :- to enter point(.)
def dot_clicked():
    pos = len(disp.get())
    disp.insert(pos, '.')


# button "π(pi)" :- to enter pi(π)
def pi_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, str(math.pi))


# button "e" :- to enter the value of "e=2.718281828459045"
def e_clicked():
    if disp.get() == '0':
        disp.delete(0, END)
    pos = len(disp.get())
    disp.insert(pos, str(math.e))

# button "(" :- to enter "("
```

```
# button "(" :- to enter "("
def bl_clicked():
    pos = len(disp.get())
    disp.insert(pos, '(')


# button ")" :- to enter ")"
def br_clicked():
    pos = len(disp.get())
    disp.insert(pos, ')')


# button "⌫" :- to delete the last entered character
def del_clicked():
    pos = len(disp.get())
    display = str(disp.get())
    if display == '':
        disp.insert(0, '0')
    elif display == ' ':
        disp.insert(0, '0')
    elif display == '0':
        pass
    else:
        disp.delete(0, END)
        disp.insert(0, display[0:pos-1])


# button "rad/deg" :- to convert from radians to degree and vice-versa
def conv_clicked():
    global switch
    if switch is None:
        switch = True
        conv_btn['text'] = "Deg"
        ans = float(disp.get())
```

conv_clicked() › if switch is None

```python

# button "rad/deg" :- to convert from radians to degree and vice-versa
def conv_clicked():
    global switch
    if switch is None:
        switch = True
        conv_btn['text'] = "Deg"
        ans = float(disp.get())
        ans = math.radians(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))

    else:
        switch = None
        conv_btn['text'] = "Rad"
        ans = float(disp.get())
        ans = math.degrees(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))


# button "ln" :- to perform ln operation on the entered number
def ln_clicked():
    try:
        ans = float(disp.get())
        ans = math.log(ans)
        disp.delete(0, END)
        disp.insert(0, str(ans))
    except Exception:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# button "%" :- to get the remainder (x%y)
```

ln_clicked() › try

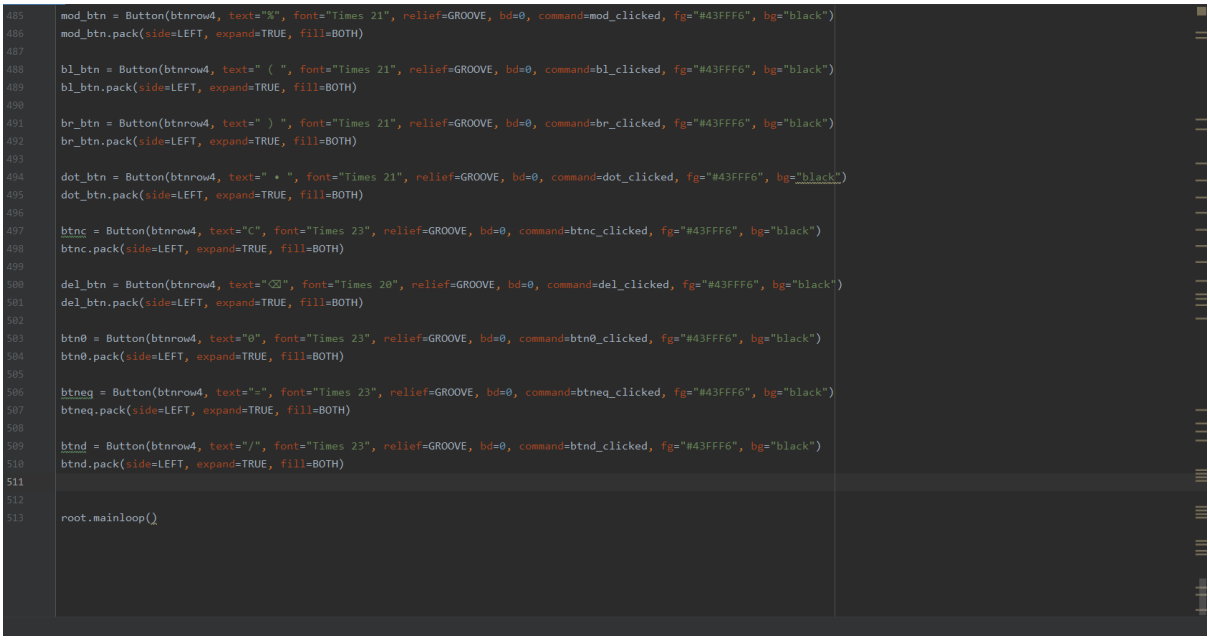
```python
# button "%" :- to get the remainder (x%y)
def mod_clicked():
    pos = len(disp.get())
    disp.insert(pos, '%')


# button "=" :- to get the output of the performed operation
def btneq_clicked(*args):
    try:
        ans = disp.get()
        ans = eval(ans)
        disp.delete(0, END)
        disp.insert(0, ans)

    except:
        tkinter.messagebox.showerror("Value Error", "Check your values and operators")


# Label

data = StringVar()

disp = Entry(root, font="Times 20", fg="#43FFF6", bg="black", bd=0, justify=RIGHT, insertbackground="#abbab1", cursor="arrow")
disp.bind("<Return>", btneq_clicked)
disp.bind("<Escape>", btnc_clicked)
disp.bind("<Key-1>", key_event)
disp.bind("<Key-2>", key_event)
disp.bind("<Key-3>", key_event)
disp.bind("<Key-4>", key_event)
disp.bind("<Key-5>", key_event)
disp.bind("<Key-6>", key_event)
disp.bind("<Key-7>", key_event)
disp.bind("<Key-8>", key_event)
disp.bind("<Key-9>", key_event)
```

```
disp.bind("<Key-9>", key_event)
disp.bind("<Key-0>", key_event)
disp.bind("<Key-.>", key_event)
disp.insert(0, '0')
disp.focus_set()
disp.pack(expand=TRUE, fill=BOTH)

# Row 1 Buttons

btnrow1 = Frame(root, bg="#000000")
btnrow1.pack(expand=TRUE, fill=BOTH)

pi_btn = Button(btnrow1, text="π", font="Times 18", relief=GROOVE, bd=0, command=pi_clicked, fg="#43FFF6", bg="black")
pi_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

fact_btn = Button(btnrow1, text=" x! ", font="Times 18", relief=GROOVE, bd=0, command=fact_clicked, fg="#43FFF6", bg="black")
fact_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sin_btn = Button(btnrow1, text="sin", font="Times 18", relief=GROOVE, bd=0, command=sin_clicked, fg="#43FFF6", bg="black")
sin_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

cos_btn = Button(btnrow1, text="cos", font="Times 18", relief=GROOVE, bd=0, command=cos_clicked, fg="#43FFF6", bg="black")
cos_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

tan_btn = Button(btnrow1, text="tan", font="Times 18", relief=GROOVE, bd=0, command=tan_clicked, fg="#43FFF6", bg="black")
tan_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn1 = Button(btnrow1, text="1", font="Times 23", relief=GROOVE, bd=0, command=btn1_clicked, fg="#43FFF6", bg="black")
btn1.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn2 = Button(btnrow1, text="2", font="Times 23", relief=GROOVE, bd=0,  command=btn2_clicked, fg="#43FFF6", bg="black")
btn2.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
```

```
btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
btn3.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnp = Button(btnrow1, text="+", font="Times 23", relief=GROOVE, bd=0, command=btnp_clicked, fg="#43FFF6", bg="black")
btnp.pack(side=LEFT, expand=TRUE, fill=BOTH)

# Row 2 Buttons

btnrow2 = Frame(root)
btnrow2.pack(expand=TRUE, fill=BOTH)

e_btn = Button(btnrow2, text="e", font="Times 18", relief=GROOVE, bd=0, command=e_clicked, fg="#43FFF6", bg="black")
e_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sqr_btn = Button(btnrow2, text=" √x ", font="Times 18", relief=GROOVE, bd=0, command=sqr_clicked, fg="#43FFF6", bg="black")
sqr_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sinh_btn = Button(btnrow2, text="sin-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arcsin_clicked, fg="#43FFF6", bg="black")
sinh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

cosh_btn = Button(btnrow2, text="cos-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arccos_clicked, fg="#43FFF6", bg="black")
cosh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

tanh_btn = Button(btnrow2, text="tan-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arctan_clicked, fg="#43FFF6", bg="black")
tanh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn4 = Button(btnrow2, text="4", font="Times 23", relief=GROOVE, bd=0, command=btn4_clicked, fg="#43FFF6", bg="black")
btn4.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn5 = Button(btnrow2, text="5", font="Times 23", relief=GROOVE, bd=0, command=btn5_clicked, fg="#43FFF6", bg="black")
btn5.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn6 = Button(btnrow2, text="6", font="Times 23", relief=GROOVE, bd=0, command=btn6_clicked, fg="#43FFF6", bg="black")
btn6.pack(side=LEFT, expand=TRUE, fill=BOTH)
```

```
444
445     btnm = Button(btnrow2, text="-", font="Times 23", relief=GROOVE, bd=0, command=btnm_clicked, fg="#43FFF6", bg="black")
446     btnm.pack(side=LEFT, expand=TRUE, fill=BOTH)
447
448     # Row 3 Buttons
449
450     btnrow3 = Frame(root)
451     btnrow3.pack(expand=TRUE, fill=BOTH)
452
453     conv_btn = Button(btnrow3, text="Rad", font="Times 12 bold", relief=GROOVE, bd=0, command=conv_clicked, fg="#43FFF6", bg="black")
454     conv_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
455
456     round_btn = Button(btnrow3, text="round", font="Times 10 bold", relief=GROOVE, bd=0, command=round_clicked, fg="#43FFF6", bg="black")
457     round_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
458
459     ln_btn = Button(btnrow3, text="ln", font="Times 18", relief=GROOVE, bd=0, command=ln_clicked, fg="#43FFF6", bg="black")
460     ln_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
461
462     logarithm_btn = Button(btnrow3, text="log", font="Times 17", relief=GROOVE, bd=0, command=logarithm_clicked, fg="#43FFF6", bg="black")
463     logarithm_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
464
465     pow_btn = Button(btnrow3, text="x^y", font="Times 17", relief=GROOVE, bd=0, command=pow_clicked, fg="#43FFF6", bg="black")
466     pow_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
467
468     btn7 = Button(btnrow3, text="7", font="Times 23", relief=GROOVE, bd=0, command=btn7_clicked, fg="#43FFF6", bg="black")
469     btn7.pack(side=LEFT, expand=TRUE, fill=BOTH)
470
471     btn8 = Button(btnrow3, text="8", font="Times 23", relief=GROOVE, bd=0, command=btn8_clicked, fg="#43FFF6", bg="black")
472     btn8.pack(side=LEFT, expand=TRUE, fill=BOTH)
473
474     btn9 = Button(btnrow3, text="9", font="Times 23", relief=GROOVE, bd=0, command=btn9_clicked, fg="#43FFF6", bg="black")
475     btn9.pack(side=LEFT, expand=TRUE, fill=BOTH)
476
477     btnml = Button(btnrow3, text="*", font="Times 23", relief=GROOVE, bd=0, command=btnml_clicked, fg="#43FFF6", bg="black")
```

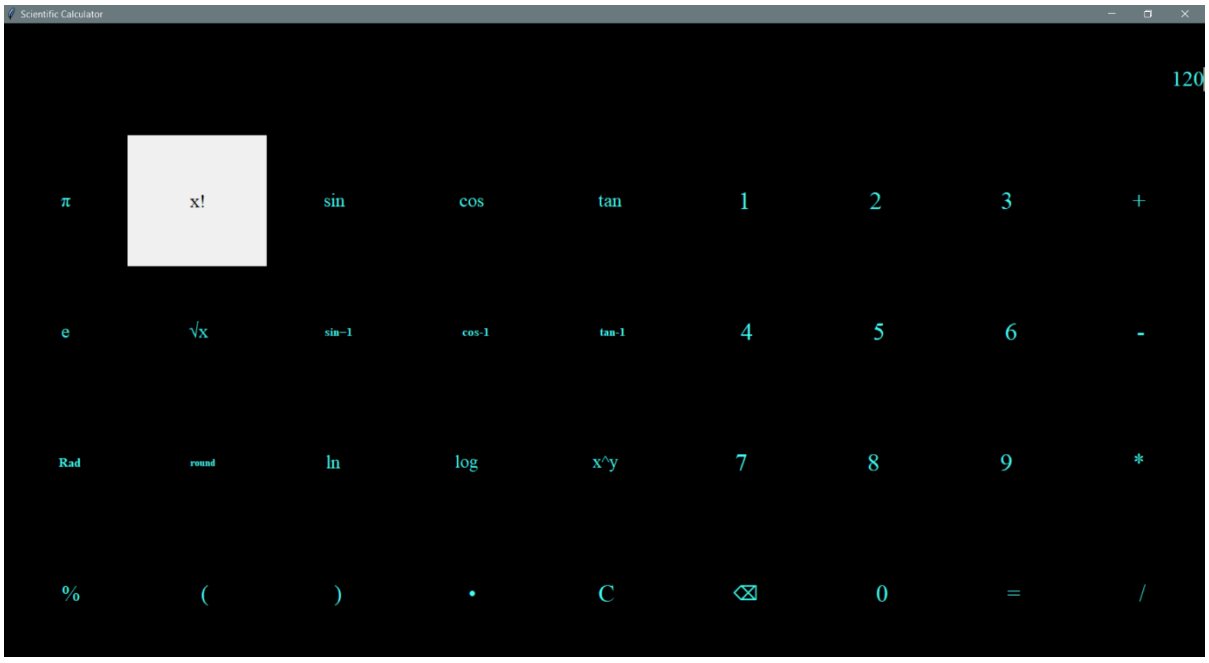
```
477     btnml = Button(btnrow3, text="*", font="Times 23", relief=GROOVE, bd=0, command=btnml_clicked, fg="#43FFF6", bg="black")
478     btnml.pack(side=LEFT, expand=TRUE, fill=BOTH)
479
480     # Row 4 Buttons
481
482     btnrow4 = Frame(root)
483     btnrow4.pack(expand=TRUE, fill=BOTH)
484
485     mod_btn = Button(btnrow4, text="%", font="Times 21", relief=GROOVE, bd=0, command=mod_clicked, fg="#43FFF6", bg="black")
486     mod_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
487
488     bl_btn = Button(btnrow4, text=" ( ", font="Times 21", relief=GROOVE, bd=0, command=bl_clicked, fg="#43FFF6", bg="black")
489     bl_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
490
491     br_btn = Button(btnrow4, text=" ) ", font="Times 21", relief=GROOVE, bd=0, command=br_clicked, fg="#43FFF6", bg="black")
492     br_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
493
494     dot_btn = Button(btnrow4, text=" • ", font="Times 21", relief=GROOVE, bd=0, command=dot_clicked, fg="#43FFF6", bg="black")
495     dot_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
496
497     btnc = Button(btnrow4, text="C", font="Times 23", relief=GROOVE, bd=0, command=btnc_clicked, fg="#43FFF6", bg="black")
498     btnc.pack(side=LEFT, expand=TRUE, fill=BOTH)
499
500     del_btn = Button(btnrow4, text="⌫", font="Times 20", relief=GROOVE, bd=0, command=del_clicked, fg="#43FFF6", bg="black")
501     del_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
502
503     btn0 = Button(btnrow4, text="0", font="Times 23", relief=GROOVE, bd=0, command=btn0_clicked, fg="#43FFF6", bg="black")
504     btn0.pack(side=LEFT, expand=TRUE, fill=BOTH)
505
506     btneq = Button(btnrow4, text="=", font="Times 23", relief=GROOVE, bd=0, command=btneq_clicked, fg="#43FFF6", bg="black")
507     btneq.pack(side=LEFT, expand=TRUE, fill=BOTH)
508
509     btnd = Button(btnrow4, text="/", font="Times 23", relief=GROOVE, bd=0, command=btnd_clicked, fg="#43FFF6", bg="black")
510     btnd.pack(side=LEFT, expand=TRUE, fill=BOTH)
```



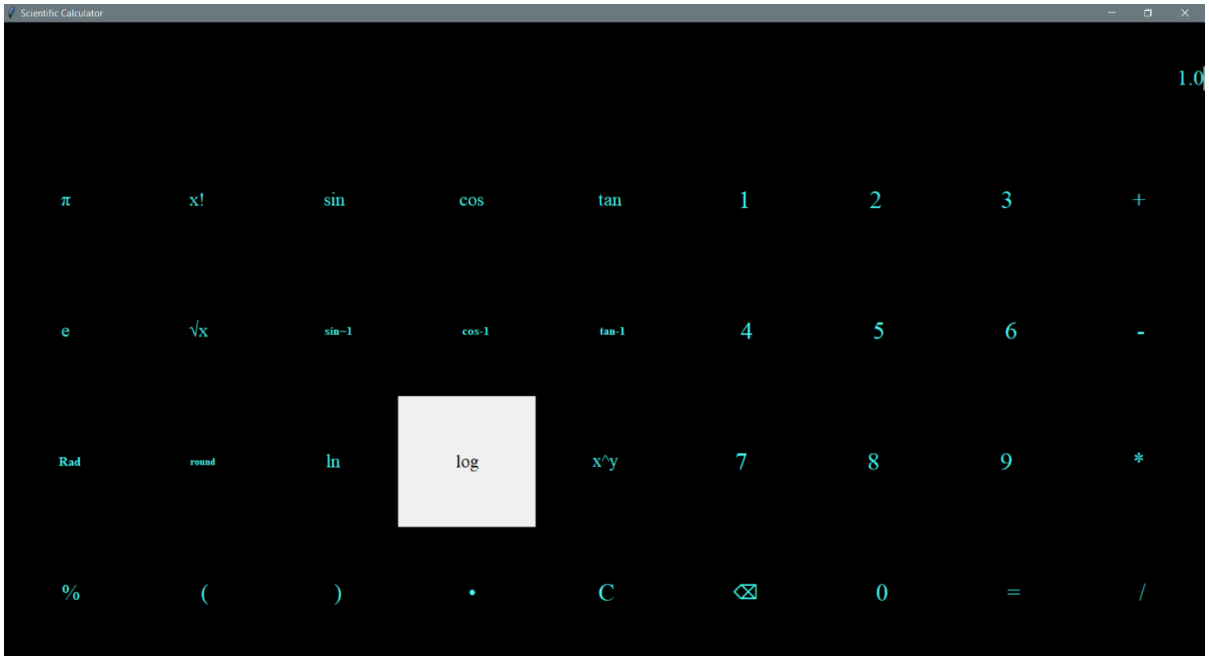
```
mod_btn = Button(btnrow4, text="%", font="Times 21", relief=GROOVE, bd=0, command=mod_clicked, fg="#43FFF6", bg="black")
mod_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

bl_btn = Button(btnrow4, text=" ( ", font="Times 21", relief=GROOVE, bd=0, command=bl_clicked, fg="#43FFF6", bg="black")
bl_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

br_btn = Button(btnrow4, text=" ) ", font="Times 21", relief=GROOVE, bd=0, command=br_clicked, fg="#43FFF6", bg="black")
br_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

dot_btn = Button(btnrow4, text=" * ", font="Times 21", relief=GROOVE, bd=0, command=dot_clicked, fg="#43FFF6", bg="black")
dot_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnc = Button(btnrow4, text="C", font="Times 23", relief=GROOVE, bd=0, command=btnc_clicked, fg="#43FFF6", bg="black")
btnc.pack(side=LEFT, expand=TRUE, fill=BOTH)

del_btn = Button(btnrow4, text="⌫", font="Times 20", relief=GROOVE, bd=0, command=del_clicked, fg="#43FFF6", bg="black")
del_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn0 = Button(btnrow4, text="0", font="Times 23", relief=GROOVE, bd=0, command=btn0_clicked, fg="#43FFF6", bg="black")
btn0.pack(side=LEFT, expand=TRUE, fill=BOTH)

btneq = Button(btnrow4, text="=", font="Times 23", relief=GROOVE, bd=0, command=btneq_clicked, fg="#43FFF6", bg="black")
btneq.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnd = Button(btnrow4, text="/", font="Times 23", relief=GROOVE, bd=0, command=btnd_clicked, fg="#43FFF6", bg="black")
btnd.pack(side=LEFT, expand=TRUE, fill=BOTH)


root.mainloop()
```

Outputs

Factorial of 5 = 125



Log 10 = 1



Technologies and Framework to be used.

System Requirements

- Processor: Intel® Pentium® 4 or AMD Athlon™ desktop processor 2 GHz or faster
- Memory: 512 MB RAM or more
- Storage space: 512 MB or more available hard disk space
- Monitor: Resolution of 1024 x 768 or higher
- Operating system: Windows® XP SP3 32-bit, Windows® 7 32-bit or 64- bit, or Mac OS X 10.6.4 or later

Ide and Framework Used

- IDE used – PyCharm Community Edition
- Frameworks – Tkinter 24.1

SWOT Analysis achieved in project.

Strength:

- Quick answer of tough mathematical equation and sums.
- User friendly easy to use.
- All the mathematical commands and equation is already compiled just we need to use as per our need.

Weakness:

- Sometimes some mathematical function does not give 100% exact answer it always contains some error in final answer, so we need to take always its round figure

Opportunities:

- Gives us a platform to make our tough work and problem solved easily.

Threats:

- It effects individuals thinking ability instead of solving equation and sums by its own students or and individuals get totally dependent or get relied on machine(scientific calculator) for making work easy.