

# 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



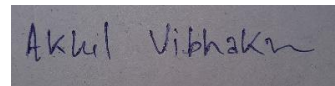
**L** OVELY  
**P** ROFESSIONAL  
**U** NIVERSITY

**Department of Intelligent Systems**  
**School of Computer Science**  
**Lovely Professional University, Jalandhar**  
October – 2020

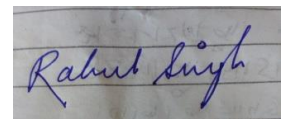
## APPENDIX – 2

**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

Sonu Kumar  
32

Lovely Professional University, Jalandhar

24 Oct 2020

## APPENDIX 3

## TABLE OF CONTENTS

<b>TITLE</b>	<b>PAGE NO.</b>
<b>1. Background and objectives of project assigned .....</b>	<b>5</b>
<b>a. Introduction .....</b>	<b>5</b>
<b>b. Background .....</b>	<b>5</b>
<b>c. Motivation .....</b>	<b>6</b>
<b>d. Outcomes of the project .....</b>	<b>6</b>
<b>2. Description of Project.....</b>	<b>7</b>
<b>a. Module 1.....</b>	<b>7</b>
<b>b. Module 2.....</b>	<b>8</b>
<b>c. Module 3.....</b>	<b>9</b>
<b>3. Description of work division in terms of role among students</b>	<b>10</b>
<b>a. Module 1.....</b>	<b>10</b>
<b>b. Module 2.....</b>	<b>10</b>
<b>c. Module 3.....</b>	<b>10</b>
<b>4. Implementation of scheduled work of project.....</b>	<b>11</b>
<b>a. Codes.....</b>	<b>11-19</b>
<b>b. Output.....</b>	<b>20</b>
<b>5. Technologies and framework to be used.....</b>	<b>21</b>
<b>6. SWOT analysis achieved in project.....</b>	<b>22</b>

## 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:

```

386 btnrow1 = Frame(root, bg="#000000")
387 btnrow1.pack(expand=TRUE, fill=BOTH)
388
389 pi_btn = Button(btnrow1, text="π", font="Times 18", relief=GROOVE, bd=0, command=pi_clicked, fg="#43FFF6", bg="black")
390 pi_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
391
392 fact_btn = Button(btnrow1, text="x!", font="Times 18", relief=GROOVE, bd=0, command=fact_clicked, fg="#43FFF6", bg="black")
393 fact_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
394
395 sin_btn = Button(btnrow1, text="sin", font="Times 18", relief=GROOVE, bd=0, command=sin_clicked, fg="#43FFF6", bg="black")
396 sin_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
397
398 cos_btn = Button(btnrow1, text="cos", font="Times 18", relief=GROOVE, bd=0, command=cos_clicked, fg="#43FFF6", bg="black")
399 cos_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
400
401 tan_btn = Button(btnrow1, text="tan", font="Times 18", relief=GROOVE, bd=0, command=tan_clicked, fg="#43FFF6", bg="black")
402 tan_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
403
404 btn1 = Button(btnrow1, text="1", font="Times 23", relief=GROOVE, bd=0, command=btn1_clicked, fg="#43FFF6", bg="black")
405 btn1.pack(side=LEFT, expand=TRUE, fill=BOTH)
406
407 btn2 = Button(btnrow1, text="2", font="Times 23", relief=GROOVE, bd=0, command=btn2_clicked, fg="#43FFF6", bg="black")
408 btn2.pack(side=LEFT, expand=TRUE, fill=BOTH)
409
410 btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
411 btn3.pack(side=LEFT, expand=TRUE, fill=BOTH)
412
413 btnp = Button(btnrow1, text="+", font="Times 23", relief=GROOVE, bd=0, command=btnp_clicked, fg="#43FFF6", bg="black")
414 btnp.pack(side=LEFT, expand=TRUE, fill=BOTH)
415
416 # Row 2 Buttons
417
418 btnrow2 = Frame(root)
419 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:

```

141         except Exception:
142             tkinter.messagebox.showerror("Value Error", "Check your values and operators")
143
144
145     # button "cos" :- to perform cos() function on the entered number
146     def cos_clicked():
147         try:
148             ans = float(dis.get())
149             if switch is True:
150                 ans = math.cos(math.radians(ans))
151             else:
152                 ans = math.cos(ans)
153             disp.delete(0, END)
154             disp.insert(0, str(ans))
155         except Exception:
156             tkinter.messagebox.showerror("Value Error", "Check your values and operators")
157
158
159     # button "tan" :- to perform tan() function on the entered number
160     def tan_clicked():
161         try:
162             ans = float(dis.get())
163             if switch is True:
164                 ans = math.tan(math.radians(ans))
165             else:
166                 ans = math.tan(ans)
167             disp.delete(0, END)
168             disp.insert(0, str(ans))
169         except Exception:
170             tkinter.messagebox.showerror("Value Error", "Check your values and operators")
171
172
173     # button "sin-1" :- to perform asin() function on the entered number
174     def arcsin_clicked():

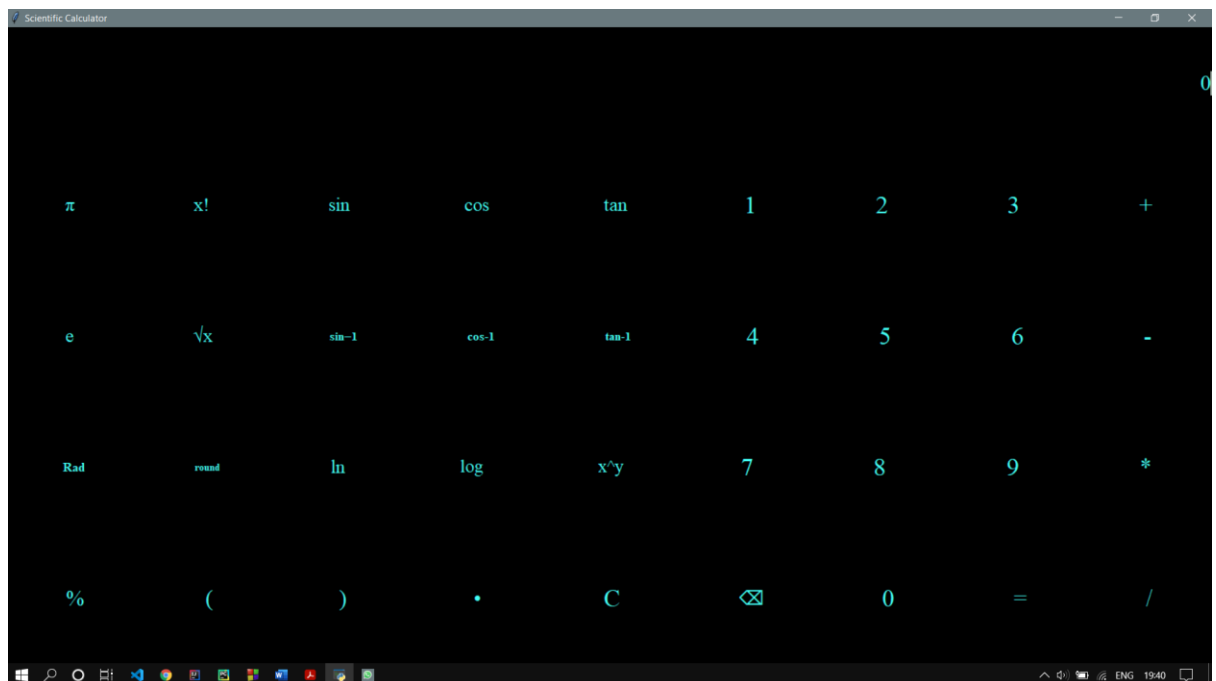
```



### 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 1<sup>st</sup> module was all about the coding of the GUI and the basic layout. 2<sup>nd</sup> module was all about the coding of the mathematical functions and in 3 module we just linked module1 with module2 and did testing.

**CODES:**

```

1 from tkinter import *
2 import tkinter.messagebox
3 import math
4 import tkinter.messagebox
5
6 root = Tk()
7 root.title("Scientific Calculator")
8 root.geometry("650x400+300+300")
9
10
11
12 switch = None
13
14 # Button on press
15
16 # button "1" :- to enter 1
17 def btn1_clicked():
18     if disp.get() == '0':
19         disp.delete(0, END)
20         pos = len(disp.get())
21         disp.insert(pos, '1')
22
23
24 # button "2" :- to enter 2
25 def btn2_clicked():
26     if disp.get() == '0':
27         disp.delete(0, END)
28         pos = len(disp.get())
29         disp.insert(pos, '2')
30
31
32 # button "3" :- to enter 3
33 def btn3_clicked():
34     if disp.get() == '0':
35         disp.delete(0, END)
36         pos = len(disp.get())
37         disp.insert(pos, '3')
38
39
40 # button "4" :- to enter 4
41 def btn4_clicked():
42     if disp.get() == '0':
43         disp.delete(0, END)
44         pos = len(disp.get())
45         disp.insert(pos, '4')
46
47
48 # button "5" :- to enter 5
49 def btn5_clicked():
50     if disp.get() == '0':
51         disp.delete(0, END)
52         pos = len(disp.get())
53         disp.insert(pos, '5')
54
55
56 # button "6" :- to enter 6
57 def btn6_clicked():
58     if disp.get() == '0':
59         disp.delete(0, END)
60         pos = len(disp.get())
61         disp.insert(pos, '6')
62
63
64 # button "7" :- to enter 7
65 def btn7_clicked():
66     if disp.get() == '0':
67         disp.delete(0, END)
68         pos = len(disp.get())
69         disp.insert(pos, '7')
70
71
72 # button "8" :- to enter 8
73 def btn8_clicked():
74     if disp.get() == '0':
75         disp.delete(0, END)
76         pos = len(disp.get())
77         disp.insert(pos, '8')
78
79
80 # button "9" :- to enter 9
81 def btn9_clicked():
82     if disp.get() == '0':
83         disp.delete(0, END)
84         pos = len(disp.get())
85         disp.insert(pos, '9')
86
87
88 # button "+" :- to enter +
89 def btn_plus_clicked():
90     if disp.get() == '0':
91         disp.delete(0, END)
92         pos = len(disp.get())
93         disp.insert(pos, '+')
94
95
96 # button "-" :- to enter -
97 def btn_minus_clicked():
98     if disp.get() == '0':
99         disp.delete(0, END)
100        pos = len(disp.get())
101        disp.insert(pos, '-')
102
103
104 # button "*" :- to enter *
105 def btn_multiply_clicked():
106     if disp.get() == '0':
107         disp.delete(0, END)
108         pos = len(disp.get())
109         disp.insert(pos, '*')
110
111
112 # button "/" :- to enter /
113 def btn_divide_clicked():
114     if disp.get() == '0':
115         disp.delete(0, END)
116         pos = len(disp.get())
117         disp.insert(pos, '/')
118
119
120 # button "=" :- to enter =
121 def btn_equals_clicked():
122     if disp.get() == '0':
123         disp.delete(0, END)
124         pos = len(disp.get())
125         disp.insert(pos, '=')
126
127
128 # button "C" :- to clear the display
129 def btn_clear_clicked():
130     disp.delete(0, END)
131     pos = len(disp.get())
132     disp.insert(pos, '0')
133
134
135 # button "CE" :- to clear the entry
136 def btn_clear_entry_clicked():
137     disp.delete(0, END)
138     pos = len(disp.get())
139     disp.insert(pos, '0')
140
141
142 # button "MC" :- to clear the memory
143 def btn_memory_clear_clicked():
144     disp.delete(0, END)
145     pos = len(disp.get())
146     disp.insert(pos, '0')
147
148
149 # button "MS" :- to store the memory
150 def btn_memory_store_clicked():
151     disp.delete(0, END)
152     pos = len(disp.get())
153     disp.insert(pos, '0')
154
155
156 # button "MR" :- to retrieve the memory
157 def btn_memory_retrieve_clicked():
158     disp.delete(0, END)
159     pos = len(disp.get())
160     disp.insert(pos, '0')
161
162
163 # button "M+" :- to add the memory
164 def btn_memory_add_clicked():
165     disp.delete(0, END)
166     pos = len(disp.get())
167     disp.insert(pos, '0')
168
169
170 # button "M-" :- to subtract the memory
171 def btn_memory_subtract_clicked():
172     disp.delete(0, END)
173     pos = len(disp.get())
174     disp.insert(pos, '0')
175
176
177 # button "M*" :- to multiply the memory
178 def btn_memory_multiply_clicked():
179     disp.delete(0, END)
180     pos = len(disp.get())
181     disp.insert(pos, '0')
182
183
184 # button "M/" :- to divide the memory
185 def btn_memory_divide_clicked():
186     disp.delete(0, END)
187     pos = len(disp.get())
188     disp.insert(pos, '0')
189
190
191 # button "sqrt" :- to calculate the square root
192 def btn_sqrt_clicked():
193     disp.delete(0, END)
194     pos = len(disp.get())
195     disp.insert(pos, '0')
196
197
198 # button "sin" :- to calculate the sine
199 def btn_sin_clicked():
200     disp.delete(0, END)
201     pos = len(disp.get())
202     disp.insert(pos, '0')
203
204
205 # button "cos" :- to calculate the cosine
206 def btn_cos_clicked():
207     disp.delete(0, END)
208     pos = len(disp.get())
209     disp.insert(pos, '0')
210
211
212 # button "tan" :- to calculate the tangent
213 def btn_tan_clicked():
214     disp.delete(0, END)
215     pos = len(disp.get())
216     disp.insert(pos, '0')
217
218
219 # button "log" :- to calculate the logarithm
220 def btn_log_clicked():
221     disp.delete(0, END)
222     pos = len(disp.get())
223     disp.insert(pos, '0')
224
225
226 # button "ln" :- to calculate the natural logarithm
227 def btn_ln_clicked():
228     disp.delete(0, END)
229     pos = len(disp.get())
230     disp.insert(pos, '0')
231
232
233 # button "exp" :- to calculate the exponential
234 def btn_exp_clicked():
235     disp.delete(0, END)
236     pos = len(disp.get())
237     disp.insert(pos, '0')
238
239
240 # button "1/x" :- to calculate the reciprocal
241 def btn_reciprocal_clicked():
242     disp.delete(0, END)
243     pos = len(disp.get())
244     disp.insert(pos, '0')
245
246
247 # button "x^y" :- to calculate the power
248 def btn_power_clicked():
249     disp.delete(0, END)
250     pos = len(disp.get())
251     disp.insert(pos, '0')
252
253
254 # button "x^2" :- to calculate the square
255 def btn_square_clicked():
256     disp.delete(0, END)
257     pos = len(disp.get())
258     disp.insert(pos, '0')
259
260
261 # button "x^3" :- to calculate the cube
262 def btn_cube_clicked():
263     disp.delete(0, END)
264     pos = len(disp.get())
265     disp.insert(pos, '0')
266
267
268 # button "x^n" :- to calculate the nth power
269 def btn_nth_power_clicked():
270     disp.delete(0, END)
271     pos = len(disp.get())
272     disp.insert(pos, '0')
273
274
275 # button "x^n" :- to calculate the nth power
276 def btn_nth_power_clicked():
277     disp.delete(0, END)
278     pos = len(disp.get())
279     disp.insert(pos, '0')
280
281
282 # button "x^n" :- to calculate the nth power
283 def btn_nth_power_clicked():
284     disp.delete(0, END)
285     pos = len(disp.get())
286     disp.insert(pos, '0')
287
288
289 # button "x^n" :- to calculate the nth power
290 def btn_nth_power_clicked():
291     disp.delete(0, END)
292     pos = len(disp.get())
293     disp.insert(pos, '0')
294
295
296 # button "x^n" :- to calculate the nth power
297 def btn_nth_power_clicked():
298     disp.delete(0, END)
299     pos = len(disp.get())
300     disp.insert(pos, '0')
301
302
303 # button "x^n" :- to calculate the nth power
304 def btn_nth_power_clicked():
305     disp.delete(0, END)
306     pos = len(disp.get())
307     disp.insert(pos, '0')
308
309
310 # button "x^n" :- to calculate the nth power
311 def btn_nth_power_clicked():
312     disp.delete(0, END)
313     pos = len(disp.get())
314     disp.insert(pos, '0')
315
316
317 # button "x^n" :- to calculate the nth power
318 def btn_nth_power_clicked():
319     disp.delete(0, END)
320     pos = len(disp.get())
321     disp.insert(pos, '0')
322
323
324 # button "x^n" :- to calculate the nth power
325 def btn_nth_power_clicked():
326     disp.delete(0, END)
327     pos = len(disp.get())
328     disp.insert(pos, '0')
329
330
331 # button "x^n" :- to calculate the nth power
332 def btn_nth_power_clicked():
333     disp.delete(0, END)
334     pos = len(disp.get())
335     disp.insert(pos, '0')
336
337
338 # button "x^n" :- to calculate the nth power
339 def btn_nth_power_clicked():
340     disp.delete(0, END)
341     pos = len(disp.get())
342     disp.insert(pos, '0')
343
344
345 # button "x^n" :- to calculate the nth power
346 def btn_nth_power_clicked():
347     disp.delete(0, END)
348     pos = len(disp.get())
349     disp.insert(pos, '0')
350
351
352 # button "x^n" :- to calculate the nth power
353 def btn_nth_power_clicked():
354     disp.delete(0, END)
355     pos = len(disp.get())
356     disp.insert(pos, '0')
357
358
359 # button "x^n" :- to calculate the nth power
360 def btn_nth_power_clicked():
361     disp.delete(0, END)
362     pos = len(disp.get())
363     disp.insert(pos, '0')
364
365
366 # button "x^n" :- to calculate the nth power
367 def btn_nth_power_clicked():
368     disp.delete(0, END)
369     pos = len(disp.get())
370     disp.insert(pos, '0')
371
372
373 # button "x^n" :- to calculate the nth power
374 def btn_nth_power_clicked():
375     disp.delete(0, END)
376     pos = len(disp.get())
377     disp.insert(pos, '0')
378
379
380 # button "x^n" :- to calculate the nth power
381 def btn_nth_power_clicked():
382     disp.delete(0, END)
383     pos = len(disp.get())
384     disp.insert(pos, '0')
385
386
387 # button "x^n" :- to calculate the nth power
388 def btn_nth_power_clicked():
389     disp.delete(0, END)
390     pos = len(disp.get())
391     disp.insert(pos, '0')
392
393
394 # button "x^n" :- to calculate the nth power
395 def btn_nth_power_clicked():
396     disp.delete(0, END)
397     pos = len(disp.get())
398     disp.insert(pos, '0')
399
400
401 # button "x^n" :- to calculate the nth power
402 def btn_nth_power_clicked():
403     disp.delete(0, END)
404     pos = len(disp.get())
405     disp.insert(pos, '0')
406
407
408 # button "x^n" :- to calculate the nth power
409 def btn_nth_power_clicked():
410     disp.delete(0, END)
411     pos = len(disp.get())
412     disp.insert(pos, '0')
413
414
415 # button "x^n" :- to calculate the nth power
416 def btn_nth_power_clicked():
417     disp.delete(0, END)
418     pos = len(disp.get())
419     disp.insert(pos, '0')
420
421
422 # button "x^n" :- to calculate the nth power
423 def btn_nth_power_clicked():
424     disp.delete(0, END)
425     pos = len(disp.get())
426     disp.insert(pos, '0')
427
428
429 # button "x^n" :- to calculate the nth power

```

```

64 # button "7" :- to enter 7
65 def btn7_clicked():
66     if disp.get() == '0':
67         disp.delete(0, END)
68         pos = len(disp.get())
69         disp.insert(pos, '7')
70
71
72 # button "8" :- to enter 8
73 def btn8_clicked():
74     if disp.get() == '0':
75         disp.delete(0, END)
76         pos = len(disp.get())
77         disp.insert(pos, '8')
78
79
80 # button "9" :- to enter 9
81 def btn9_clicked():
82     if disp.get() == '0':
83         disp.delete(0, END)
84         pos = len(disp.get())
85         disp.insert(pos, '9')
86
87
88 # button "0" :- to enter 0
89 def btn0_clicked():
90     if disp.get() == '0':
91         disp.delete(0, END)
92         pos = len(disp.get())
93         disp.insert(pos, '0')
94
95
96 # function for entering the numeric characters and point(.)
97 def key_event(*args):
98     if disp.get() == '0':
99         key_event()
100
101
102 # function for entering the numeric characters and point(.)
103 def key_event(*args):
104     if disp.get() == '0':
105         disp.delete(0, END)
106
107
108 # button "+" :- to perform add operation
109 def btnp_clicked():
110     pos = len(disp.get())
111     disp.insert(pos, '+')
112
113
114 # button "-" :- to perform subtraction operation
115 def btmm_clicked():
116     pos = len(disp.get())
117     disp.insert(pos, '-')
118
119
120 # button "*" :- to perform multiplication operation
121 def btmm1_clicked():
122     pos = len(disp.get())
123     disp.insert(pos, '*')
124
125
126 # button "/" :- to perform division operation
127 def btnd_clicked():
128     pos = len(disp.get())
129     disp.insert(pos, '/')
130
131
132 # button "C" :- to clean the screen
133 def btnc_clicked(*args):
134     disp.delete(0, END)
135
136
137 btnc_clicked()

```

```

125 # button "C" :- to clean the screen
126 def btn_clicked(angles):
127     disp.delete(0, END)
128     disp.insert(0, '0')
129
130
131 # button "sin" :- to perform sin() function on the entered number
132 def sin_clicked():
133     try:
134         ans = float(disp.get())
135         if switch is True:
136             ans = math.sin(math.radians(ans))
137         else:
138             ans = math.sin(ans)
139         disp.delete(0, END)
140         disp.insert(0, str(ans))
141     except Exception:
142         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
143
144
145 # button "cos" :- to perform cos() function on the entered number
146 def cos_clicked():
147     try:
148         ans = float(disp.get())
149         if switch is True:
150             ans = math.cos(math.radians(ans))
151         else:
152             ans = math.cos(ans)
153         disp.delete(0, END)
154         disp.insert(0, str(ans))
155     except Exception:
156         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
157
158
159 cos_clicked() except Exception
160
161 # button "tan" :- to perform tan() function on the entered number
162 def tan_clicked():
163     try:
164         ans = float(disp.get())
165         if switch is True:
166             ans = math.tan(math.radians(ans))
167         else:
168             ans = math.tan(ans)
169         disp.delete(0, END)
170         disp.insert(0, str(ans))
171     except Exception:
172         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
173
174
175 # button "sin-1" :- to perform asin() function on the entered number
176 def arcsin_clicked():
177     try:
178         ans = float(disp.get())
179         if switch is True:
180             ans = math.degrees(math.asin(ans))
181         else:
182             ans = math.asin(ans)
183         disp.delete(0, END)
184         disp.insert(0, str(ans))
185     except Exception:
186         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
187
188
189 # button "cos-1" :- to perform acos() function on the entered number
190 def arccos_clicked():
191     try:
192         ans = float(disp.get())
193         if switch is True:

```

```

187 # button "cos-1" :- to perform acos() function on the entered number
188 def arccos_clicked():
189     try:
190         ans = float(dispatch.get())
191         if switch is True:
192             ans = math.degrees(math.acos(ans))
193         else:
194             ans = math.acos(ans)
195         disp.delete(0, END)
196         disp.insert(0, str(ans))
197     except Exception:
198         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
199
200
201 # button "tan-1" :- to perform atan() function on the entered number
202 def arctan_clicked():
203     try:
204         ans = float(dispatch.get())
205         if switch is True:
206             ans = math.degrees(math.atan(ans))
207         else:
208             ans = math.atan(ans)
209         disp.delete(0, END)
210         disp.insert(0, str(ans))
211     except Exception:
212         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
213
214
215 # button "x^y" :- to perform y times x operation on the entered number x(say)
216 def pow_clicked():
217     pos = len(dispatch.get())
218     disp.insert(pos, '**')
219
220 pow_clicked()
221
222 # button "round" :- to roundoff the entered number
223 def round_clicked():
224     try:
225         ans = float(dispatch.get())
226         ans = round(ans)
227         disp.delete(0, END)
228         disp.insert(0, str(ans))
229     except Exception:
230         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
231
232
233 # button "log" :- to perform log() function on the entered number
234 def logarithm_clicked():
235     try:
236         ans = float(dispatch.get())
237         ans = math.log10(ans)
238         disp.delete(0, END)
239         disp.insert(0, str(ans))
240     except Exception:
241         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
242
243
244 # button "x!" :- to perform factorial operation on the entered number x(say)
245 def fact_clicked():
246     try:
247         ans = float(dispatch.get())
248         ans = math.factorial(ans)
249         disp.delete(0, END)
250         disp.insert(0, str(ans))
251     except Exception:
252         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
253
254
255 # button "√x" :- to perform sgare root operation on the entered number x(say)

```

```

253 # button "√x" :- to perform sqare root operation on the entered number x(say)
254 def sqr_clicked():
255     try:
256         ans = float(disg.get())
257         ans = math.sqrt(ans)
258         disg.delete(0, END)
259         disg.insert(0, str(ans))
260     except Exception:
261         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
262
263
264 # button "dot(.)" :- to enter point(.)
265 def dot_clicked():
266     pos = len(disg.get())
267     disg.insert(pos, '.')
268
269
270 # button "π(pi)" :- to enter pi(π)
271 def pi_clicked():
272     if disg.get() == '0':
273         disg.delete(0, END)
274     pos = len(disg.get())
275     disg.insert(pos, str(math.pi))
276
277
278 # button "e" :- to enter the value of "e=2.718281828459045"
279 def e_clicked():
280     if disg.get() == '0':
281         disg.delete(0, END)
282     pos = len(disg.get())
283     disg.insert(pos, str(math.e))
284
285
286 # button "(" :- to enter "("
287
288 # button "(" :- to enter "("
289 def bl_clicked():
290     pos = len(disg.get())
291     disg.insert(pos, '(')
292
293
294 # button ")" :- to enter ")"
295 def br_clicked():
296     pos = len(disg.get())
297     disg.insert(pos, ')')
298
299
300 # button "⌫" :- to delete the last entered character
301 def del_clicked():
302     pos = len(disg.get())
303     display = str(disg.get())
304     if display == '':
305         disg.insert(0, '0')
306     elif display == ' ':
307         disg.insert(0, '0')
308     elif display == '0':
309         pass
310     else:
311         disg.delete(0, END)
312         disg.insert(0, display[0:pos-1])
313
314
315 # button "rad/deg" :- to convert from radians to degree and vice-versa
316 def conv_clicked():
317     global switch
318     if switch is None:
319         switch = True
320         conv_btn['text'] = "Deg"
321         ans = float(disg.get())
322         conv_clicked()
323     if switch is None:

```

```

311
312
313 # button "rad/deg" :- to convert from radians to degree and vice-versa
314 def conv_clicked():
315     global switch
316     if switch is None:
317         switch = True
318         conv_btn['text'] = "Deg"
319         ans = float(dispatch.get())
320         ans = math.radians(ans)
321         disp.delete(0, END)
322         disp.insert(0, str(ans))
323
324     else:
325         switch = None
326         conv_btn['text'] = "Rad"
327         ans = float(dispatch.get())
328         ans = math.degrees(ans)
329         disp.delete(0, END)
330         disp.insert(0, str(ans))
331
332
333 # button "ln" :- to perform ln operation on the entered number
334 def ln_clicked():
335     try:
336         ans = float(dispatch.get())
337         ans = math.log(ans)
338         disp.delete(0, END)
339         disp.insert(0, str(ans))
340     except Exception:
341         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
342
343
344 # button "%" :- to get the remainder (x%y)
345 ln_clicked()
346 try
347
348 # button "%" :- to get the remainder (x%y)
349 def mod_clicked():
350     pos = len(dispatch.get())
351     disp.insert(pos, '%')
352
353
354 # button "=" :- to get the output of the performed operation
355 def btneq_clicked(*args):
356     try:
357         ans = dispatch.get()
358         ans = eval(ans)
359         disp.delete(0, END)
360         disp.insert(0, ans)
361
362     except:
363         tkinter.messagebox.showerror("Value Error", "Check your values and operators")
364
365
366 # Label
367 data = StringVar()
368
369 disp = Entry(root, font="Times 20", fg="#43FFF6", bg="black", bd=0, justify=RIGHT, insertbackground="#abbab1", cursor="arrow")
370 disp.bind("<Return>", btneq_clicked)
371 disp.bind("<Escape>", btnc_clicked)
372 disp.bind("<Key-1>", key_event)
373 disp.bind("<Key-2>", key_event)
374 disp.bind("<Key-3>", key_event)
375 disp.bind("<Key-4>", key_event)
376 disp.bind("<Key-5>", key_event)
377 disp.bind("<Key-6>", key_event)
378 disp.bind("<Key-7>", key_event)
379 disp.bind("<Key-8>", key_event)
380 disp.bind("<Key-9>", key_event)

```



```

377 disp.bind("<Key-9", key_event)
378 disp.bind("<Key-0", key_event)
379 disp.bind("<Key-.", key_event)
380 disp.insert(0, '0')
381 disp.focus_set()
382 disp.pack(expand=TRUE, fill=BOTH)
383
384 # Row 1 Buttons
385
386 btnrow1 = Frame(root, bg="#000000")
387 btnrow1.pack(expand=TRUE, fill=BOTH)
388
389 pi_btn = Button(btnrow1, text="π", font="Times 18", relief=GROOVE, bd=0, command=pi_clicked, fg="#43FFF6", bg="black")
390 pi_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
391
392 fact_btn = Button(btnrow1, text="x!", font="Times 18", relief=GROOVE, bd=0, command=fact_clicked, fg="#43FFF6", bg="black")
393 fact_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
394
395 sin_btn = Button(btnrow1, text="sin", font="Times 18", relief=GROOVE, bd=0, command=sin_clicked, fg="#43FFF6", bg="black")
396 sin_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
397
398 cos_btn = Button(btnrow1, text="cos", font="Times 18", relief=GROOVE, bd=0, command=cos_clicked, fg="#43FFF6", bg="black")
399 cos_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
400
401 tan_btn = Button(btnrow1, text="tan", font="Times 18", relief=GROOVE, bd=0, command=tan_clicked, fg="#43FFF6", bg="black")
402 tan_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
403
404 btn1 = Button(btnrow1, text="1", font="Times 23", relief=GROOVE, bd=0, command=btn1_clicked, fg="#43FFF6", bg="black")
405 btn1.pack(side=LEFT, expand=TRUE, fill=BOTH)
406
407 btn2 = Button(btnrow1, text="2", font="Times 23", relief=GROOVE, bd=0, command=btn2_clicked, fg="#43FFF6", bg="black")
408 btn2.pack(side=LEFT, expand=TRUE, fill=BOTH)
409
410 btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
411
412 btn3 = Button(btnrow1, text="3", font="Times 23", relief=GROOVE, bd=0, command=btn3_clicked, fg="#43FFF6", bg="black")
413 btn3.pack(side=LEFT, expand=TRUE, fill=BOTH)
414
415 btnp = Button(btnrow1, text="+", font="Times 23", relief=GROOVE, bd=0, command=btnp_clicked, fg="#43FFF6", bg="black")
416 btnp.pack(side=LEFT, expand=TRUE, fill=BOTH)
417
418 # Row 2 Buttons
419
420 btnrow2 = Frame(root)
421 btnrow2.pack(expand=TRUE, fill=BOTH)
422
423 e_btn = Button(btnrow2, text="e", font="Times 18", relief=GROOVE, bd=0, command=e_clicked, fg="#43FFF6", bg="black")
424 e_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
425
426 sqr_btn = Button(btnrow2, text="√x", font="Times 18", relief=GROOVE, bd=0, command=sqr_clicked, fg="#43FFF6", bg="black")
427 sqr_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
428
429 sinh_btn = Button(btnrow2, text="sin-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arcsin_clicked, fg="#43FFF6", bg="black")
430 sinh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
431
432 cosh_btn = Button(btnrow2, text="cos-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arccos_clicked, fg="#43FFF6", bg="black")
433 cosh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
434
435 tanh_btn = Button(btnrow2, text="tan-1", font="Times 11 bold", relief=GROOVE, bd=0, command=arctan_clicked, fg="#43FFF6", bg="black")
436 tanh_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
437
438 btn4 = Button(btnrow2, text="4", font="Times 23", relief=GROOVE, bd=0, command=btn4_clicked, fg="#43FFF6", bg="black")
439 btn4.pack(side=LEFT, expand=TRUE, fill=BOTH)
440
441 btn5 = Button(btnrow2, text="5", font="Times 23", relief=GROOVE, bd=0, command=btn5_clicked, fg="#43FFF6", bg="black")
442 btn5.pack(side=LEFT, expand=TRUE, fill=BOTH)
443
444 btn6 = Button(btnrow2, text="6", font="Times 23", relief=GROOVE, bd=0, command=btn6_clicked, fg="#43FFF6", bg="black")
445 btn6.pack(side=LEFT, expand=TRUE, fill=BOTH)

```

```

544 btnm = Button(btnrow2, text="", font="Times 23", relief=GROOVE, bd=0, command=btnm_clicked, fg="#43FFF6", bg="black")
545 btnm.pack(side=LEFT, expand=TRUE, fill=BOTH)
546
547 # Row 3 Buttons
548
549 btnrow3 = Frame(root)
550 btnrow3.pack(expand=TRUE, fill=BOTH)
551
552 conv_btn = Button(btnrow3, text="Rad", font="Times 12 bold", relief=GROOVE, bd=0, command=conv_clicked, fg="#43FFF6", bg="black")
553 conv_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
554
555 round_btn = Button(btnrow3, text="round", font="Times 10 bold", relief=GROOVE, bd=0, command=round_clicked, fg="#43FFF6", bg="black")
556 round_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
557
558 ln_btn = Button(btnrow3, text="ln", font="Times 18", relief=GROOVE, bd=0, command=ln_clicked, fg="#43FFF6", bg="black")
559 ln_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
560
561 logarithm_btn = Button(btnrow3, text="log", font="Times 17", relief=GROOVE, bd=0, command=logarithm_clicked, fg="#43FFF6", bg="black")
562 logarithm_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
563
564 pow_btn = Button(btnrow3, text="x^y", font="Times 17", relief=GROOVE, bd=0, command=pow_clicked, fg="#43FFF6", bg="black")
565 pow_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
566
567 btn7 = Button(btnrow3, text="7", font="Times 23", relief=GROOVE, bd=0, command=btn7_clicked, fg="#43FFF6", bg="black")
568 btn7.pack(side=LEFT, expand=TRUE, fill=BOTH)
569
570 btn8 = Button(btnrow3, text="8", font="Times 23", relief=GROOVE, bd=0, command=btn8_clicked, fg="#43FFF6", bg="black")
571 btn8.pack(side=LEFT, expand=TRUE, fill=BOTH)
572
573 btn9 = Button(btnrow3, text="9", font="Times 23", relief=GROOVE, bd=0, command=btn9_clicked, fg="#43FFF6", bg="black")
574 btn9.pack(side=LEFT, expand=TRUE, fill=BOTH)
575
576 btnml = Button(btnrow3, text="", font="Times 23", relief=GROOVE, bd=0, command=btnml_clicked, fg="#43FFF6", bg="black")
577
578 btnml = Button(btnrow3, text="", font="Times 23", relief=GROOVE, bd=0, command=btnml_clicked, fg="#43FFF6", bg="black")
579 btnml.pack(side=LEFT, expand=TRUE, fill=BOTH)
580
581 # Row 4 Buttons
582
583 btnrow4 = Frame(root)
584 btnrow4.pack(expand=TRUE, fill=BOTH)
585
586 mod_btn = Button(btnrow4, text="K", font="Times 21", relief=GROOVE, bd=0, command=mod_clicked, fg="#43FFF6", bg="black")
587 mod_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
588
589 bl_btn = Button(btnrow4, text=" ( ", font="Times 21", relief=GROOVE, bd=0, command=bl_clicked, fg="#43FFF6", bg="black")
590 bl_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
591
592 br_btn = Button(btnrow4, text=" ) ", font="Times 21", relief=GROOVE, bd=0, command=br_clicked, fg="#43FFF6", bg="black")
593 br_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
594
595 dot_btn = Button(btnrow4, text=" * ", font="Times 21", relief=GROOVE, bd=0, command=dot_clicked, fg="#43FFF6", bg="black")
596 dot_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
597
598 btnc = Button(btnrow4, text="C", font="Times 23", relief=GROOVE, bd=0, command=btnc_clicked, fg="#43FFF6", bg="black")
599 btnc.pack(side=LEFT, expand=TRUE, fill=BOTH)
600
601 del_btn = Button(btnrow4, text="⌫", font="Times 20", relief=GROOVE, bd=0, command=del_clicked, fg="#43FFF6", bg="black")
602 del_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
603
604 btn0 = Button(btnrow4, text="0", font="Times 23", relief=GROOVE, bd=0, command=btn0_clicked, fg="#43FFF6", bg="black")
605 btn0.pack(side=LEFT, expand=TRUE, fill=BOTH)
606
607 btneq = Button(btnrow4, text="=", font="Times 23", relief=GROOVE, bd=0, command=btneq_clicked, fg="#43FFF6", bg="black")
608 btneq.pack(side=LEFT, expand=TRUE, fill=BOTH)
609
610 bnd = Button(btnrow4, text="/", font="Times 23", relief=GROOVE, bd=0, command=btnd_clicked, fg="#43FFF6", bg="black")
611 btnd.pack(side=LEFT, expand=TRUE, fill=BOTH)

```

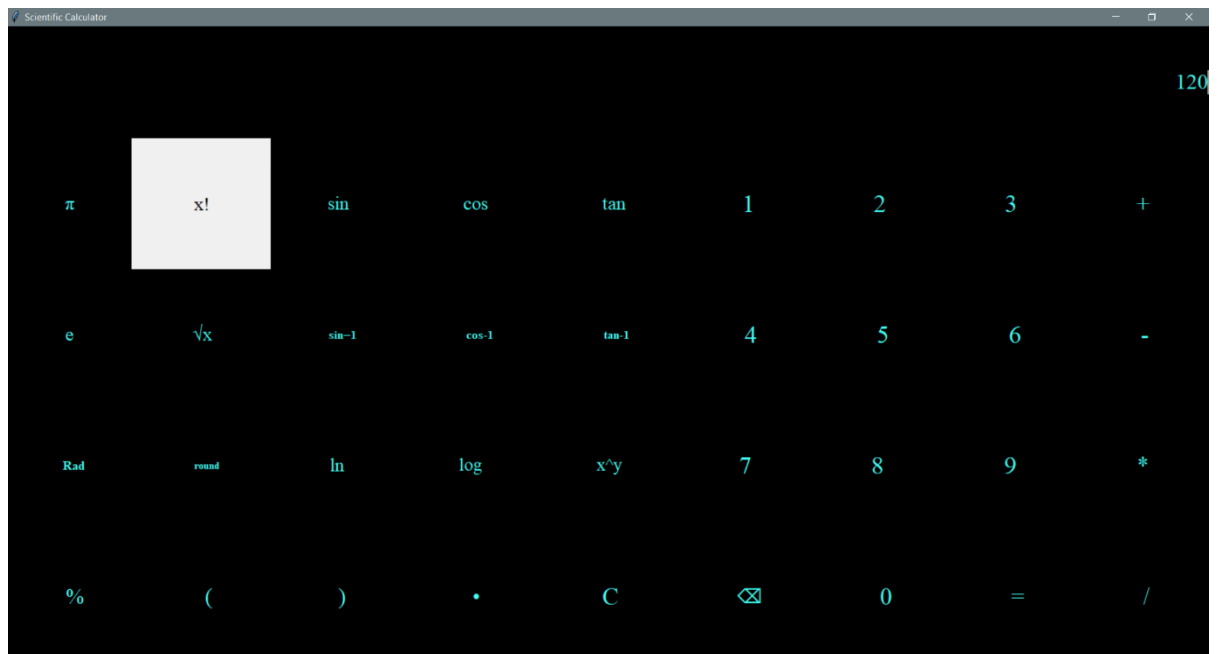
```

505 mod_btn = Button(btnrow4, text="%", font="Times 21", relief=GROOVE, bd=0, command=mod_clicked, fg="#43FFF6", bg="black")
506 mod_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
507
508 bl_btn = Button(btnrow4, text="(", font="Times 21", relief=GROOVE, bd=0, command=bl_clicked, fg="#43FFF6", bg="black")
509 bl_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
510
511 br_btn = Button(btnrow4, text=")", font="Times 21", relief=GROOVE, bd=0, command=br_clicked, fg="#43FFF6", bg="black")
512 br_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
513
514 dot_btn = Button(btnrow4, text=".", font="Times 21", relief=GROOVE, bd=0, command=dot_clicked, fg="#43FFF6", bg="black")
515 dot_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
516
517 btnc = Button(btnrow4, text="C", font="Times 23", relief=GROOVE, bd=0, command=btnc_clicked, fg="#43FFF6", bg="black")
518 btnc.pack(side=LEFT, expand=TRUE, fill=BOTH)
519
520 del_btn = Button(btnrow4, text="⌫", font="Times 20", relief=GROOVE, bd=0, command=del_clicked, fg="#43FFF6", bg="black")
521 del_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)
522
523 btn0 = Button(btnrow4, text="0", font="Times 23", relief=GROOVE, bd=0, command=btn0_clicked, fg="#43FFF6", bg="black")
524 btn0.pack(side=LEFT, expand=TRUE, fill=BOTH)
525
526 btneq = Button(btnrow4, text="=", font="Times 23", relief=GROOVE, bd=0, command=btneq_clicked, fg="#43FFF6", bg="black")
527 btneq.pack(side=LEFT, expand=TRUE, fill=BOTH)
528
529 btnd = Button(btnrow4, text="/", font="Times 23", relief=GROOVE, bd=0, command=btnd_clicked, fg="#43FFF6", bg="black")
530 btnd.pack(side=LEFT, expand=TRUE, fill=BOTH)
531
532
533 root.mainloop()

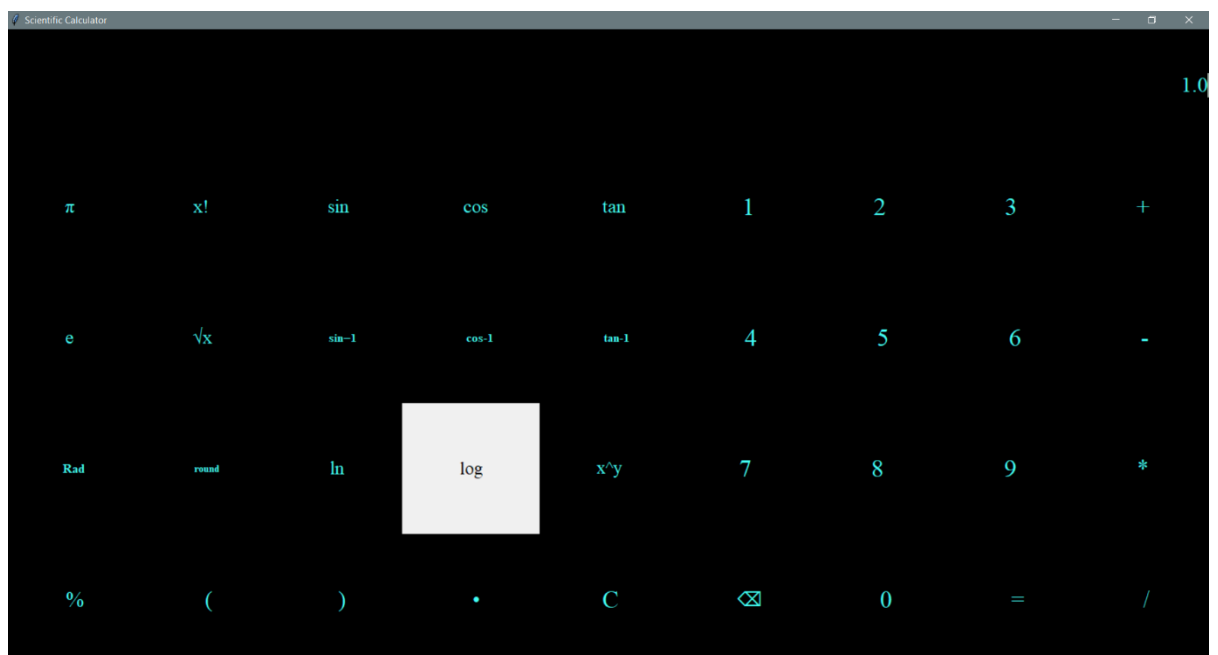
```

## 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.