

Course Design A Report

Title of the Project: Calculator



Major :	Software Engineering
Class :	2019
Name :	HOSEN ARAFAT
ID No :	199076003

School of Computer Science and Technology

December 14, 2021

CONTENTS

I . Environment configuration

1. Download and install Pycharm (The whole process needs network) ...	5
1.1. Open the official website of Pycharm, Click DOWNLOAD to download. (https://www.jetbrains.com/pycharm/).....	5
1.2. Click the download button under community to download.....	5
1.3. run the installation package (in administrator mode).....	6
1.4. always click 'next' to install.....	7
1.5. click all of the chooses.....	8
1.6. Download successful.....	8

2. Create a project

2.1. Click 'i confrim...' to enter the pycharm.....	9
2.2. Click new project to create a project.....	10
2.3. Choose the environment. The default is OK. Click create to create it.....	11
2.4. After entering, delete all codes in main.py first.....	11

II .Development design

1.Basic configuration

1.1. The pyinstaller library needs to be installed in this experiment. We need to import it first. First, click file - > setting to open the configuration interface and select Python interpreter under project.....	12
1.2. Click the configure button to select Add to add compiler.....	14
1.3. Select the python folder location in the base interpreter.....	15
1.4. After selecting the compiler, click '+' to add the library.....	16
1.5. Select the pyinstaller library to download.....	18
1.6. download succeeded.....	20

2.Interface writing

2.1. Add the Tkinter library that comes with the python compiler.....	21
2.2. Add a form, set its title, length, width, and display.....	21
2.3. Click Run - > run... And select the main.py file to compile.....	22
2.4. Add input text box (display box).....	24
2.5. Add buttons for 1-9 and other operations.....	24
2.6. Add bigger length, width, and display.....	25
2.7. Add bigger buttons for 1-9 and other operations.....	27
2.8. Add buttons border for 1-9 and other operations.....	28
2.9. Add buttons background color for 1-9 and other operations.....	30

3.Logic design

3.1 Write the number enter function of the input box, and add the change function to the 1-9 button. When the button inputs a number, it will be updated in the input box in real time.....	32
3.2 Write the sum of total function.....	34
3.3. Write the Clear Entry and All Clear Entry function.....	36
3.4 Compiling calculation function add, subtract, multiply and divide, and equal function, and write code, click the button to call the corresponding function.....	38
3.5. Write the mathPM function.....	40
3.6. Write the Squared function.....	42
3.7. Run the program for calculator related operations.....	46
1) "add" 10+4=14.....	46
2) "sub" 50 – 6 = 44.....	48
3) "multi" 16 * 2 = 32.....	49
4) "divide" 25 / 5 = 5.....	50
5) "mathPM" 69 ±.....	52
6) "Squared" 9√.....	53

III. Packaged applications

1. Click terminal to open the terminal.....	54
2. Enter the code "pyinstaller –onefile main.py " package	54
3. Package succeeded.....	55
4.Open the main.exe file in the dist folder.....	55

IV.Codes..... 56

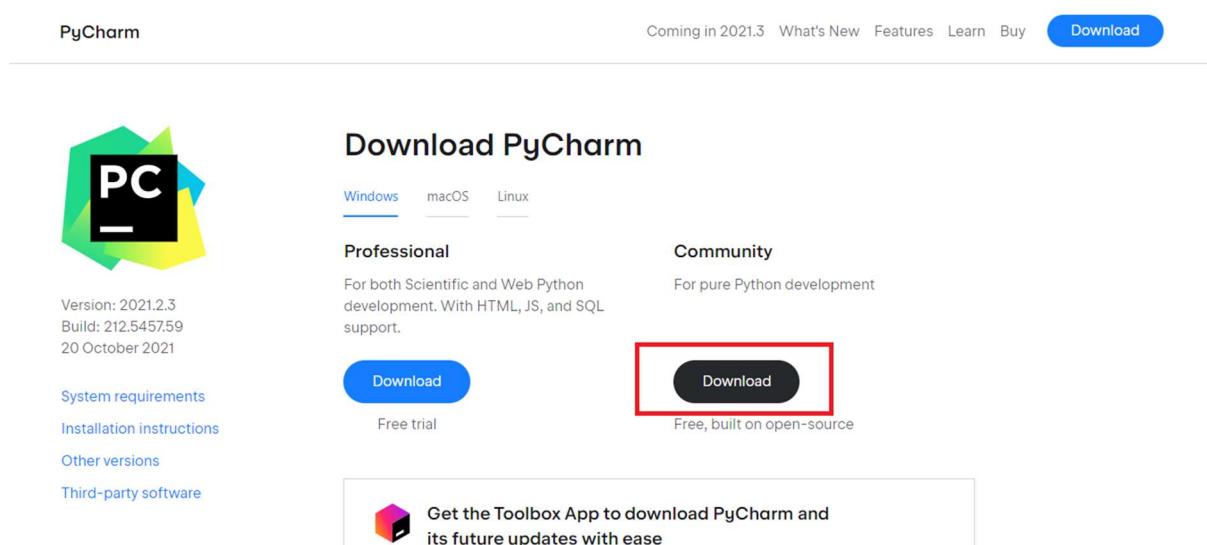
I . Environment configuration

1. Download and install Pycharm (The whole process needs network)

1.1. Open the official website of Pycharm, Click DOWNLOAD to download. (<https://www.jetbrains.com/pycharm/>)



1.2. Click the download button under community to download



S

Thank you for downloading PyCharm!

Your download should have started. If it hasn't, please use the [direct link](#).

Download and verify the file [SHA-256 checksum](#).



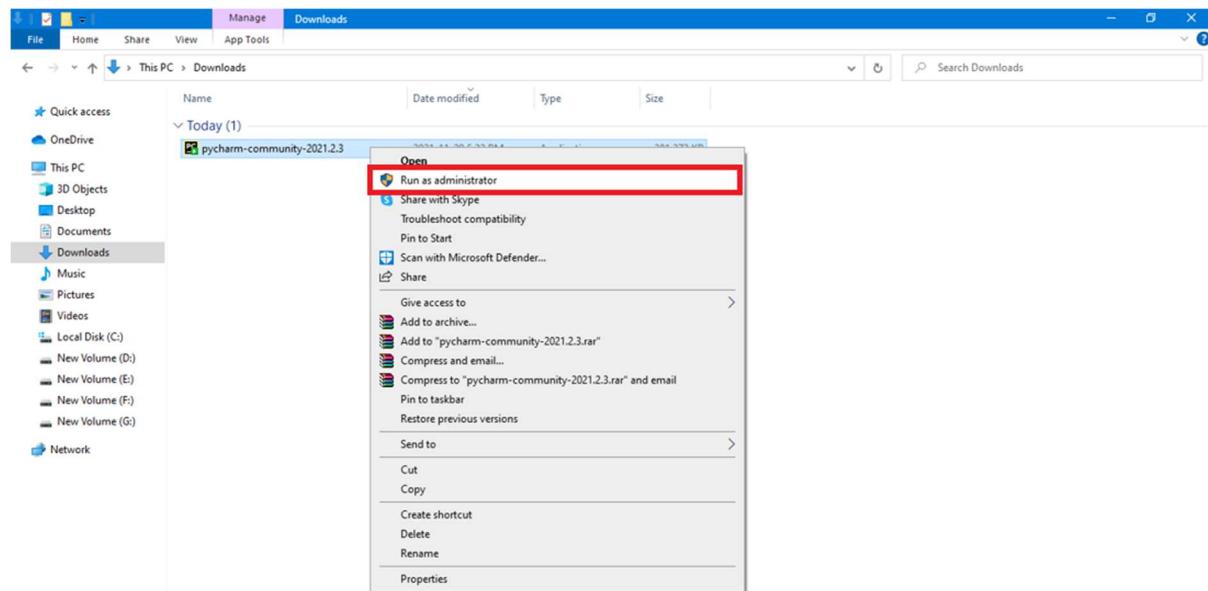
Getting Started

Send me helpful educational materials during my trial.

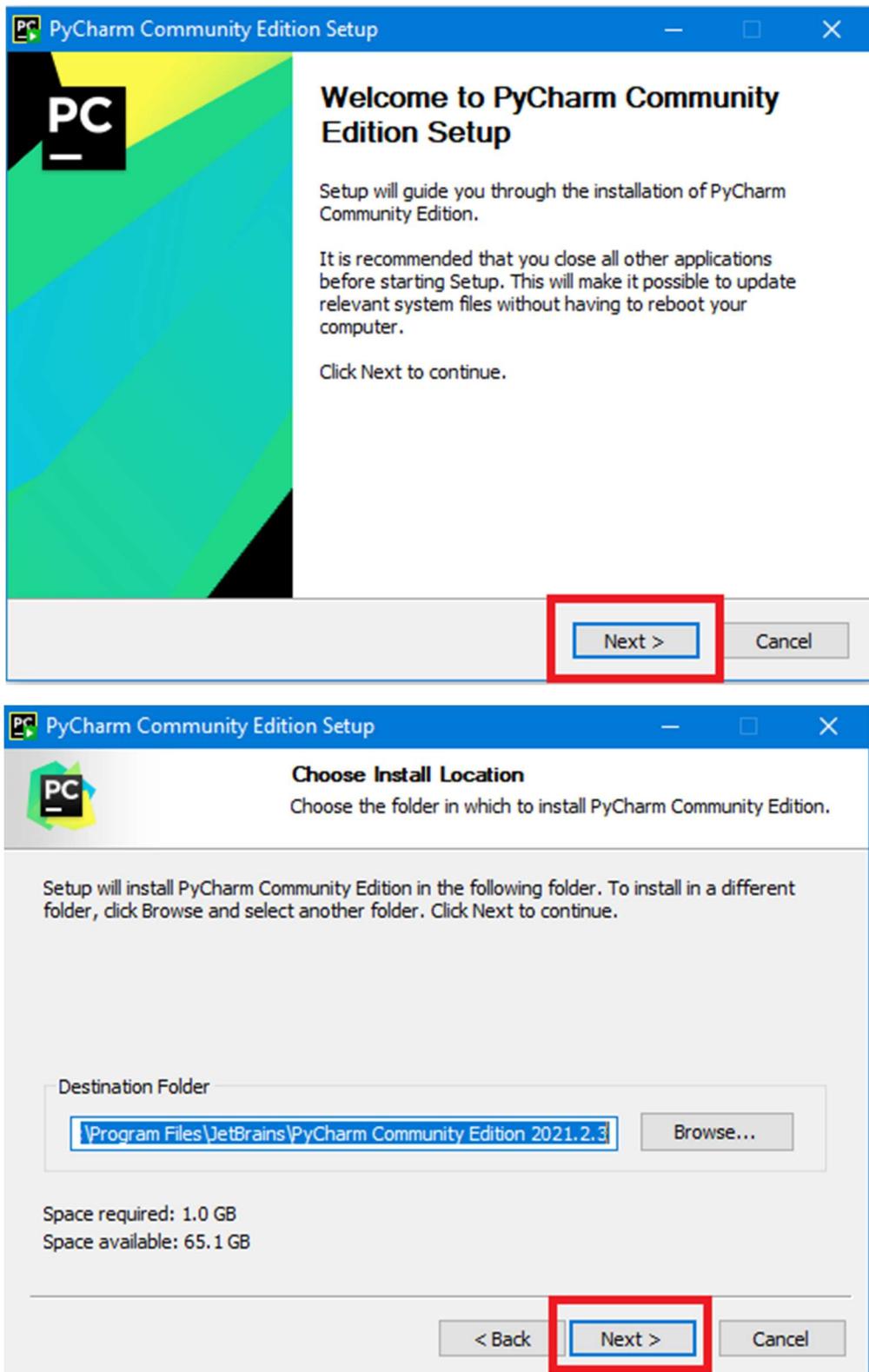
New to PyCharm?

[Follow us on Twitter](#)

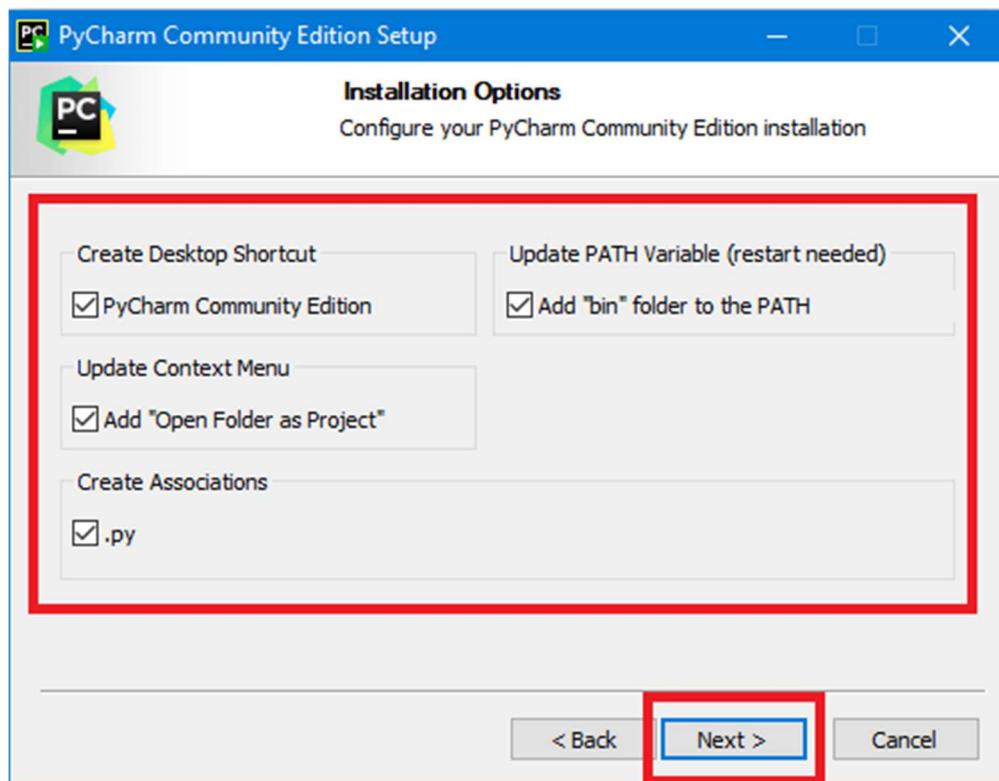
1.3. Run the installation package (in administrator mode).



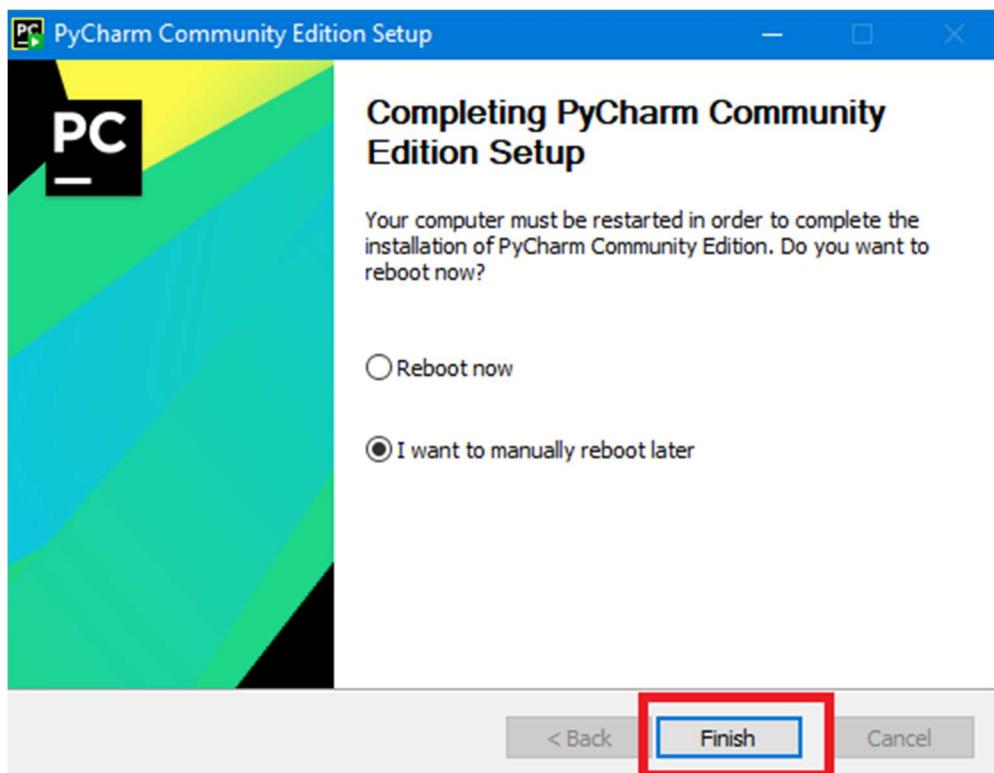
1.4. always click 'next' to install.



1.5. click all of the chooses

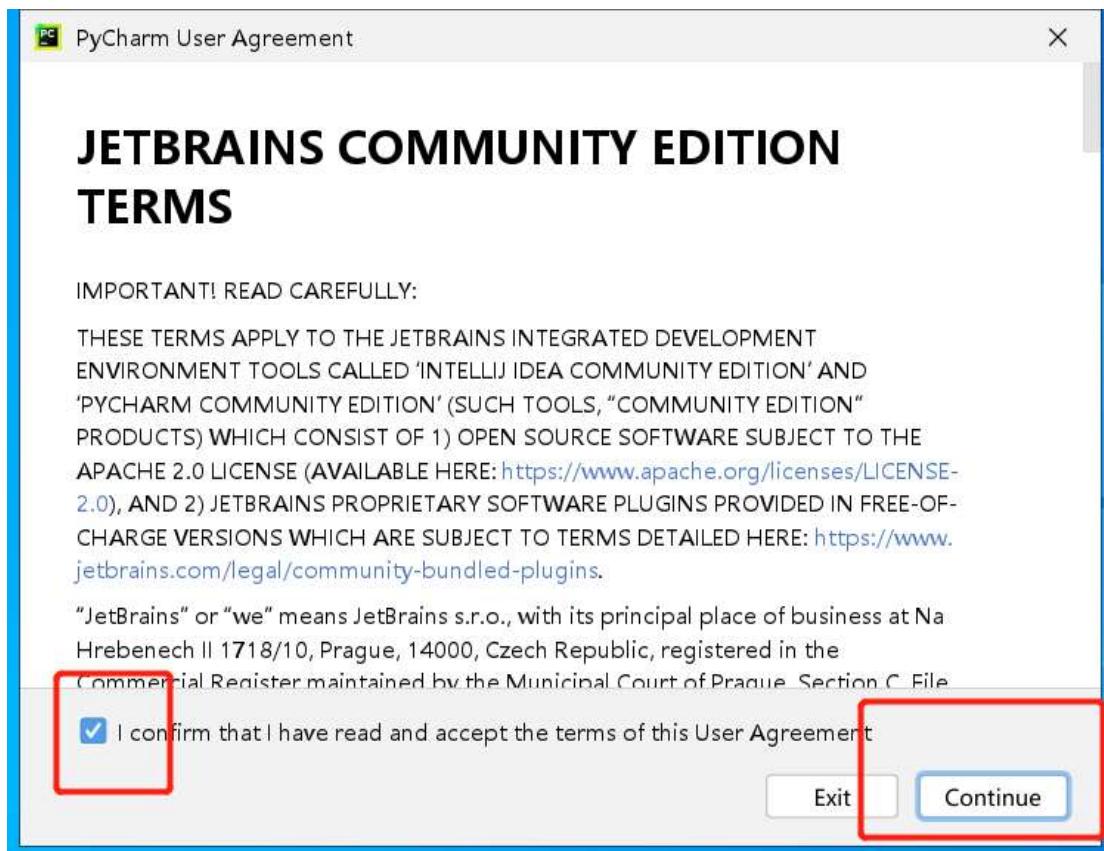


1.6. Download successful

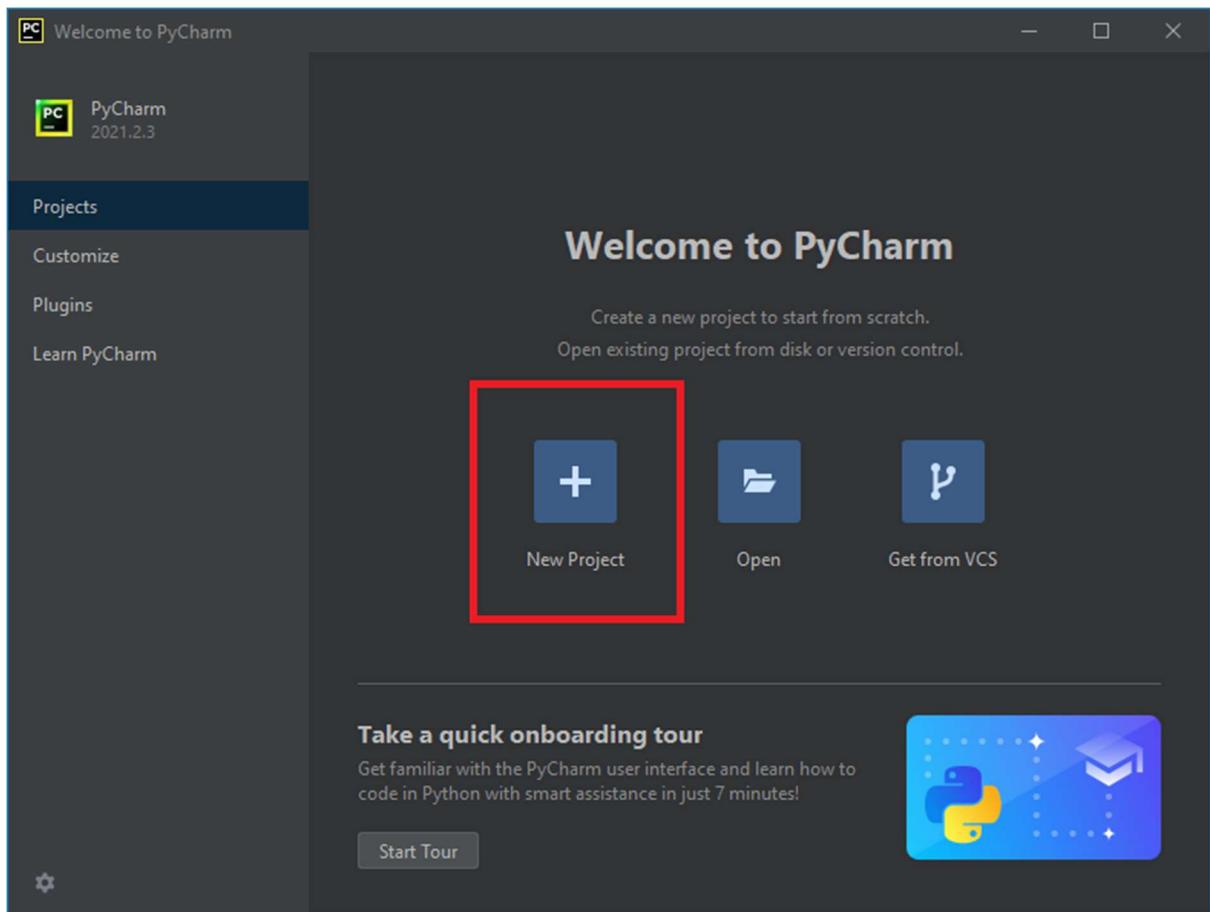


2. Create a project

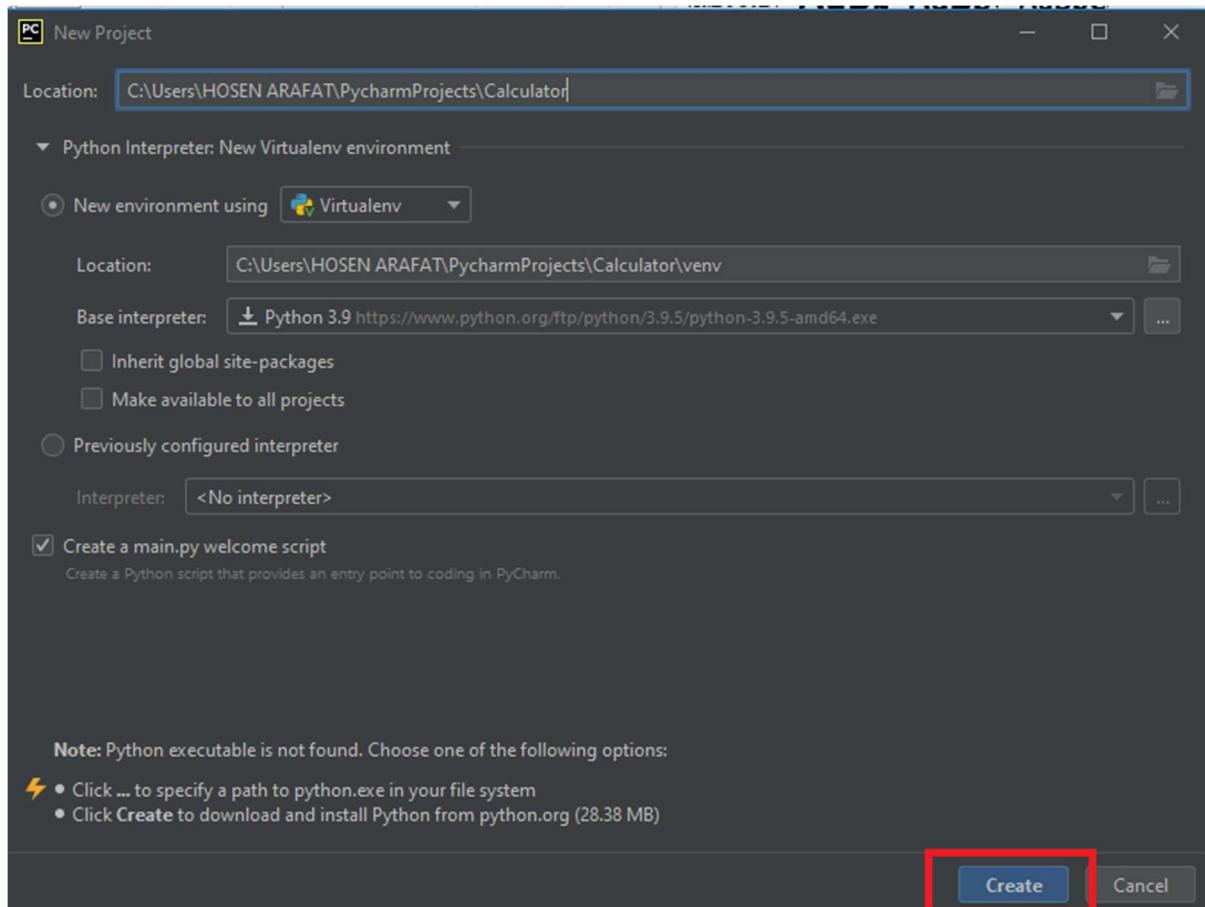
2.1. Click 'i confrim...' to enter the pycharm



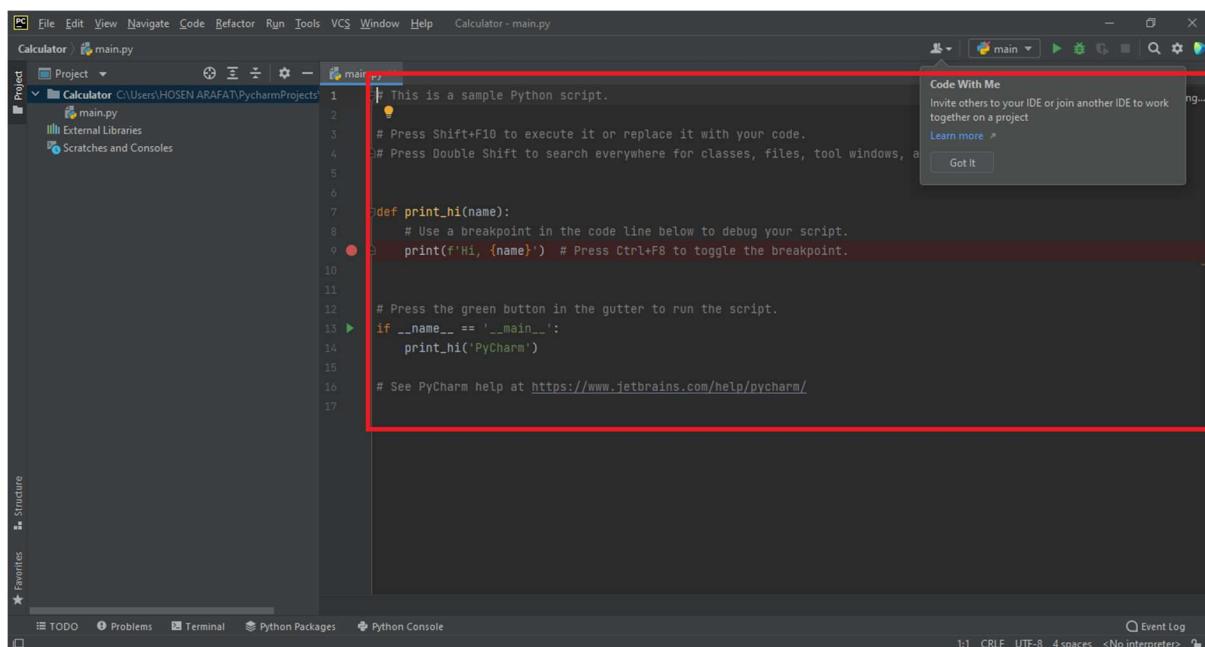
2.2. Click new project to create a project

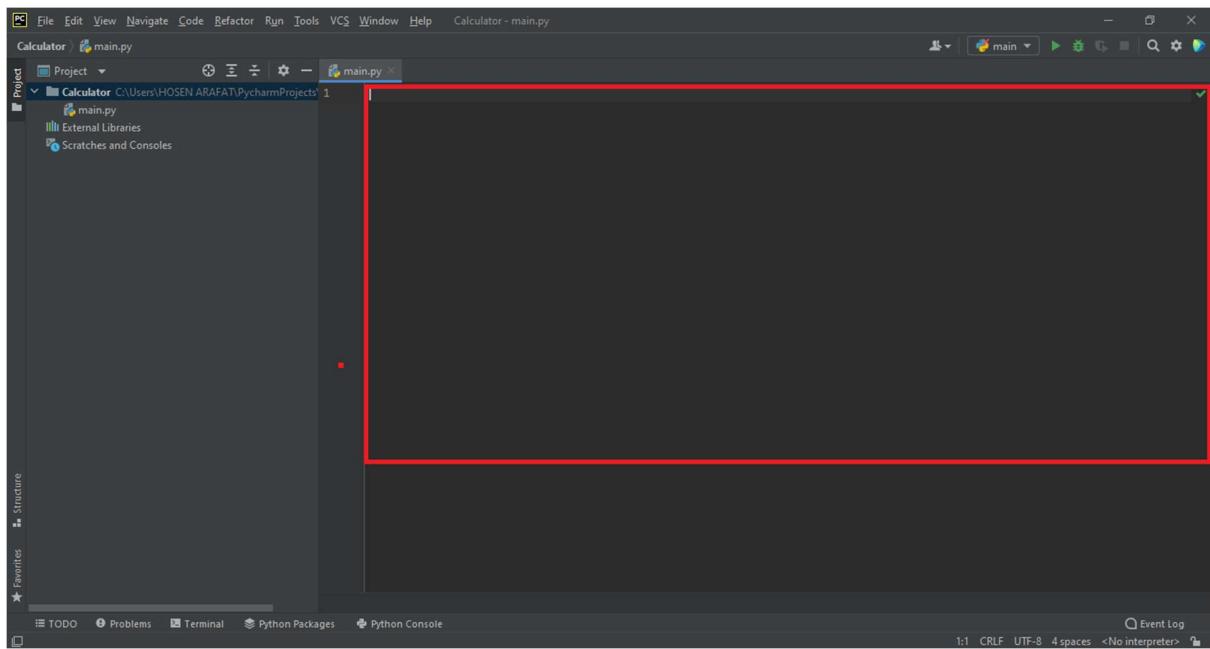


2.3. Choose the environment. The default is OK. Click create to create it



2.4. After entering, delete all codes in main.py first

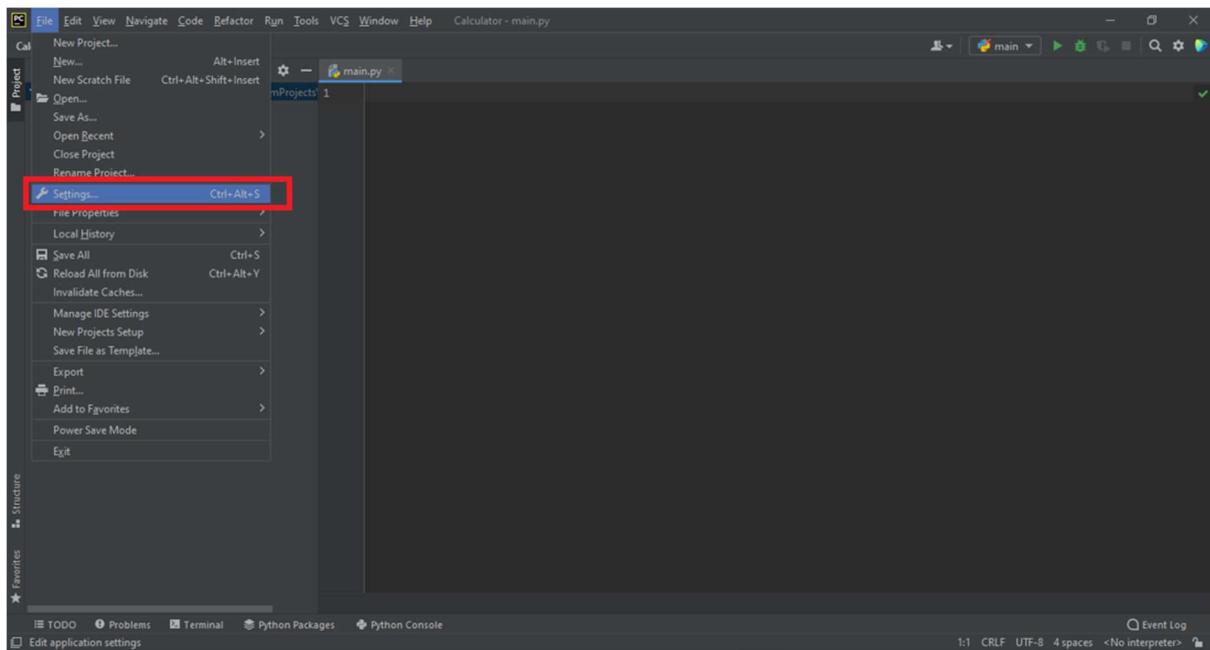


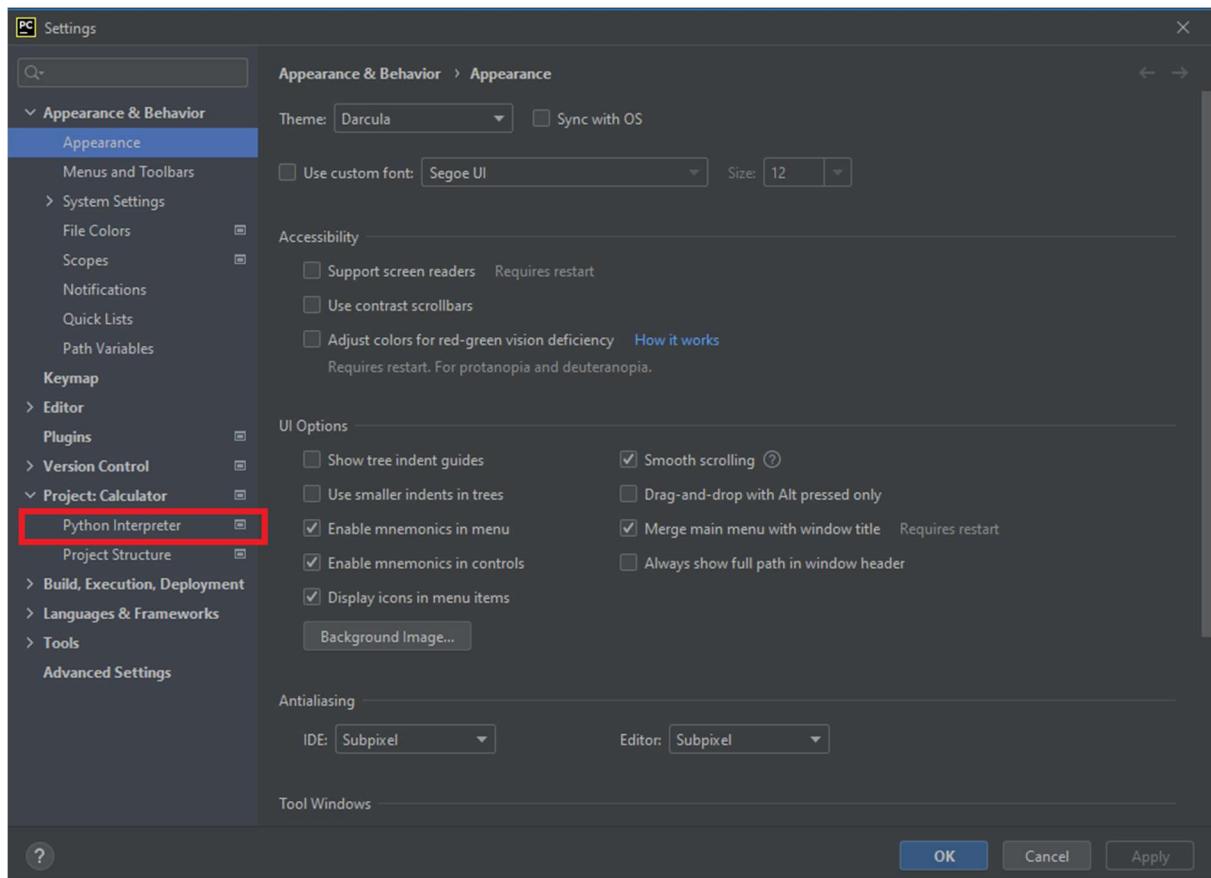


II .Development design

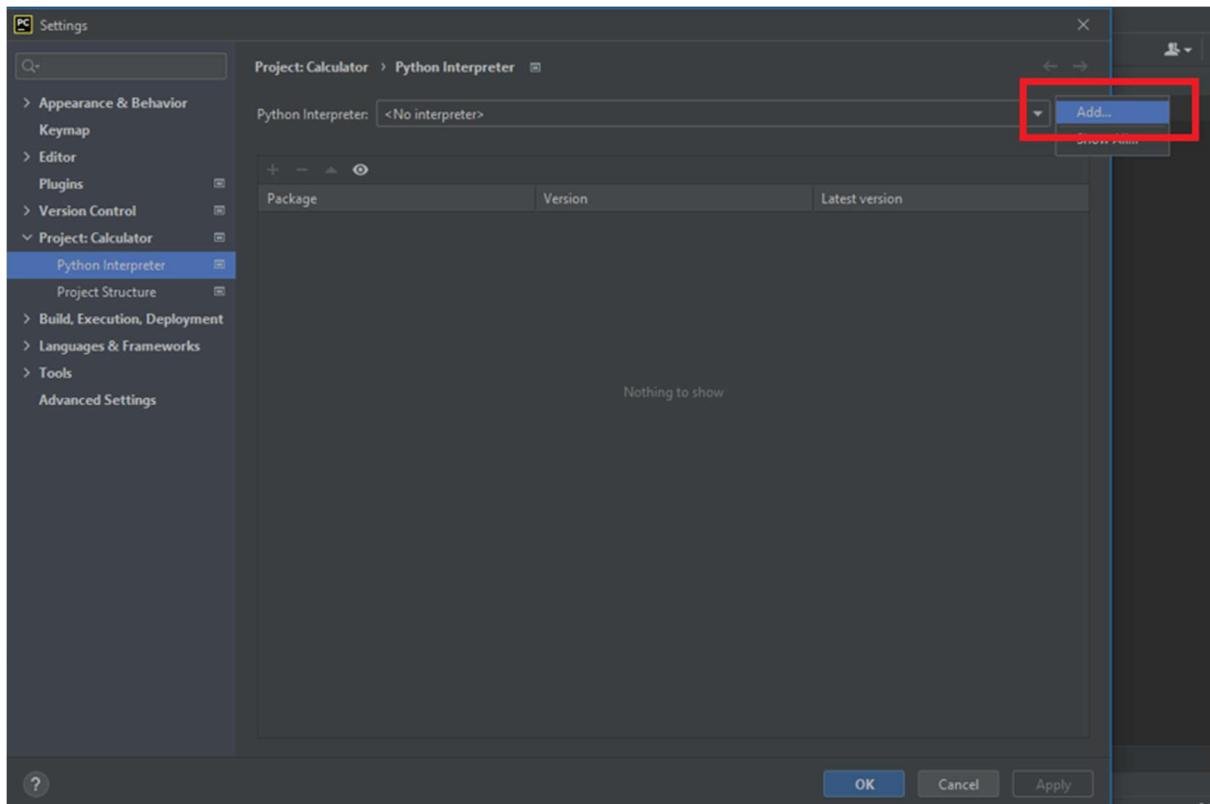
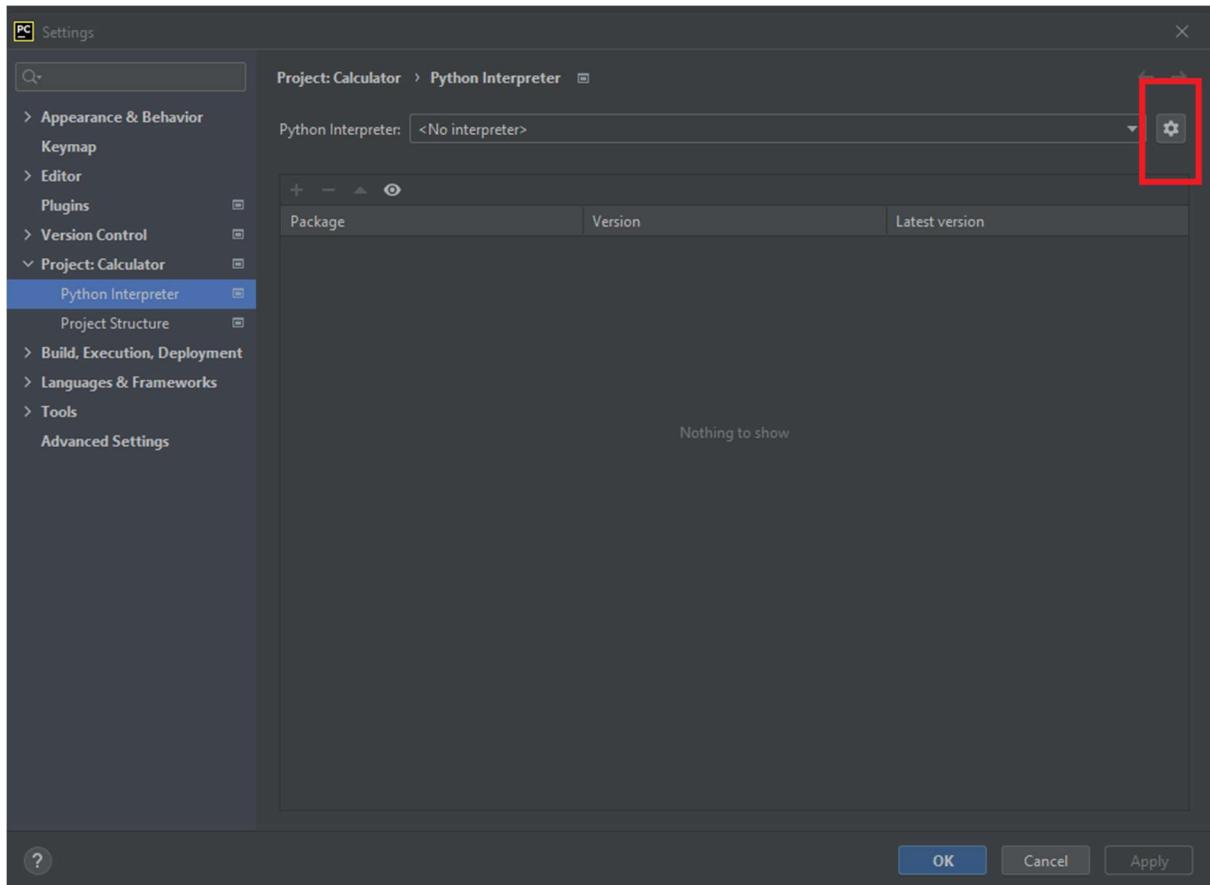
1.Basic configuration

1.1. The pyinstaller library needs to be installed in this experiment. We need to import it first. First, click file -> setting to open the configuration interface and select Python interpreter under project

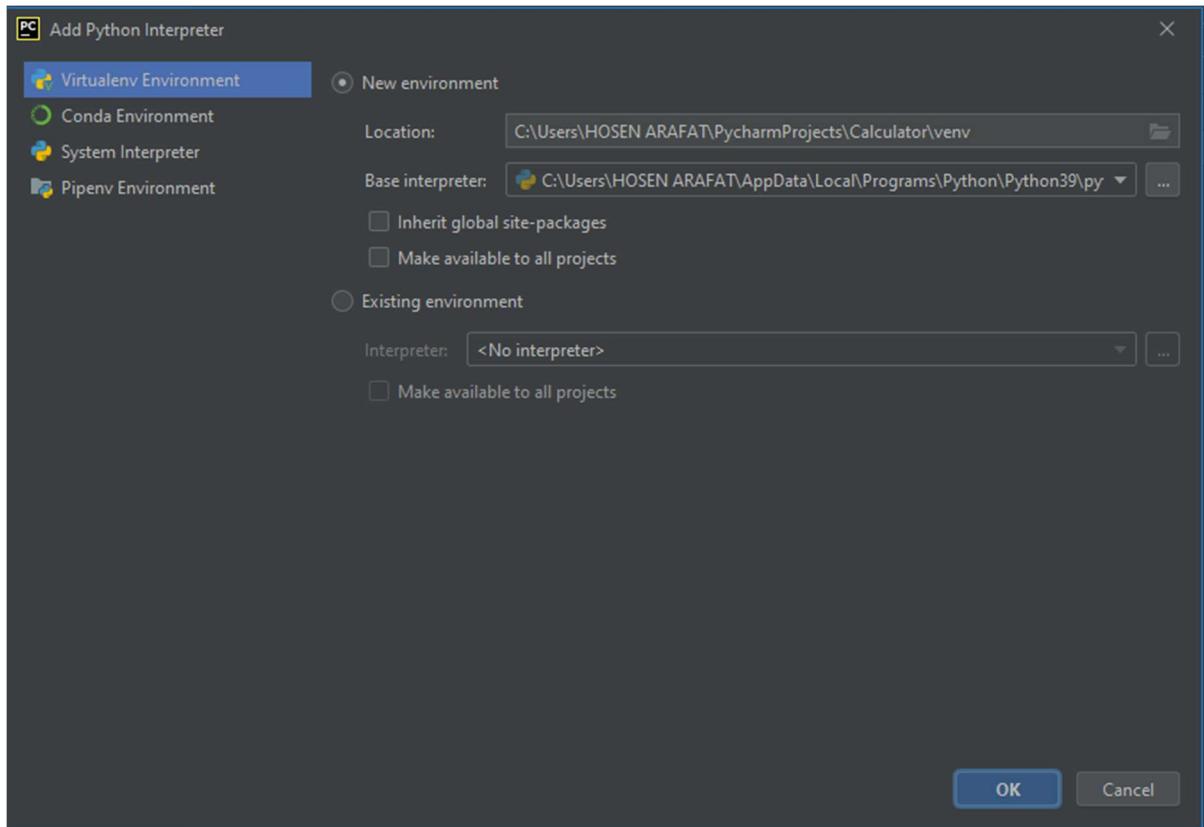




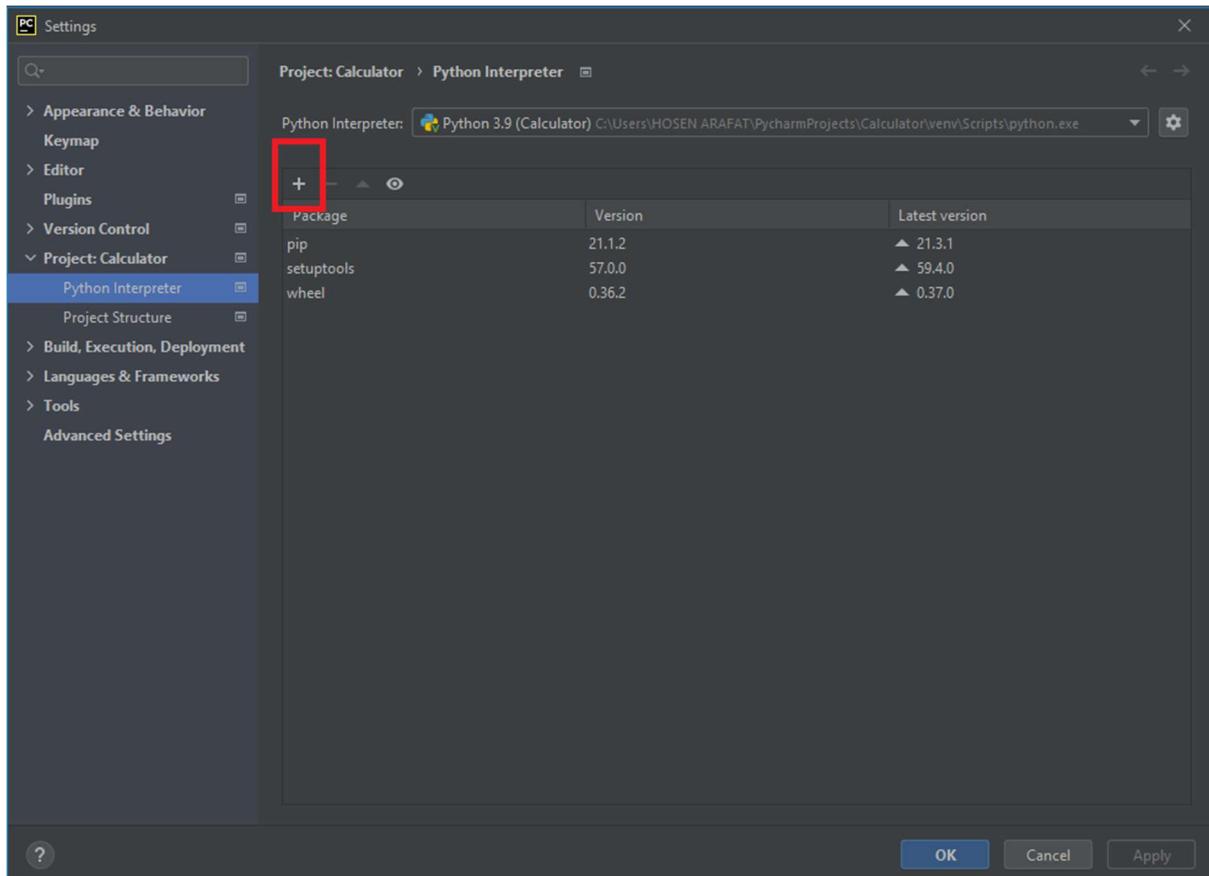
1.2. Click the configure button to select Add to add compiler.

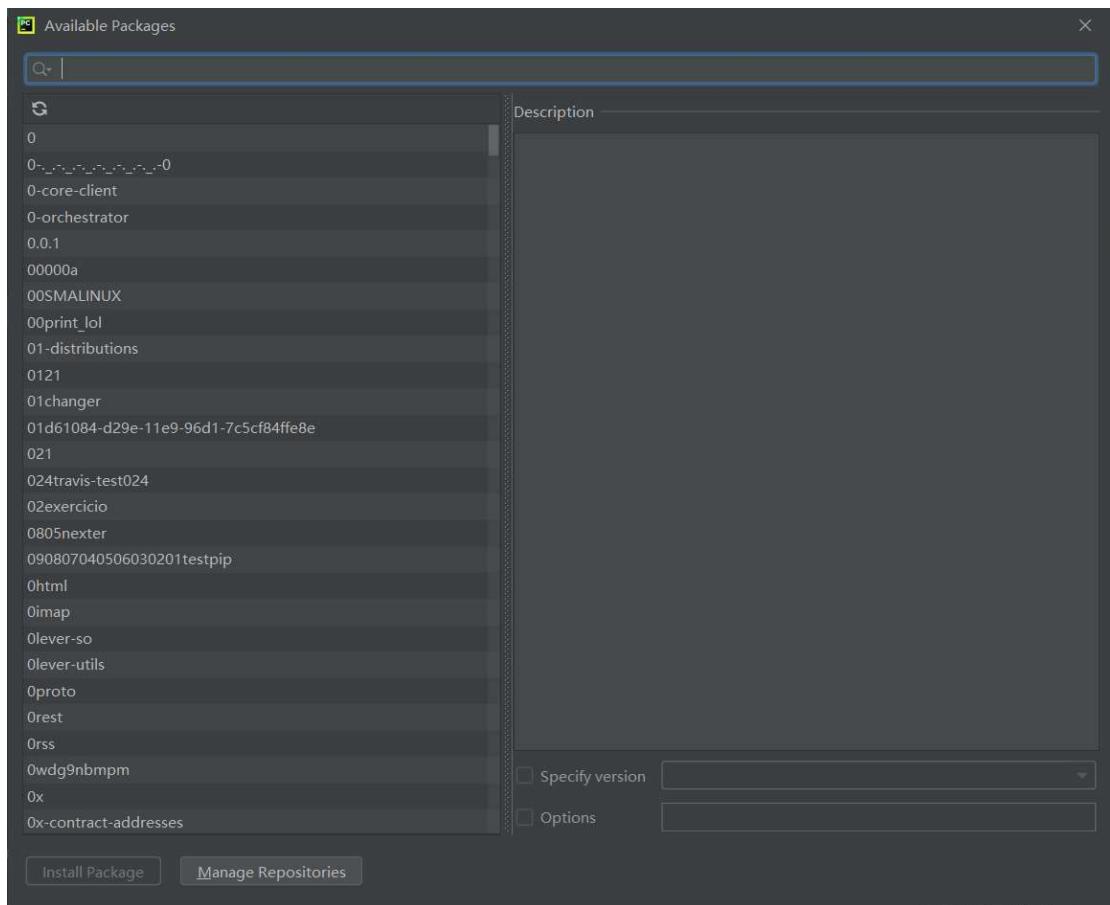


1.3. Select the python folder location in the base interpreter

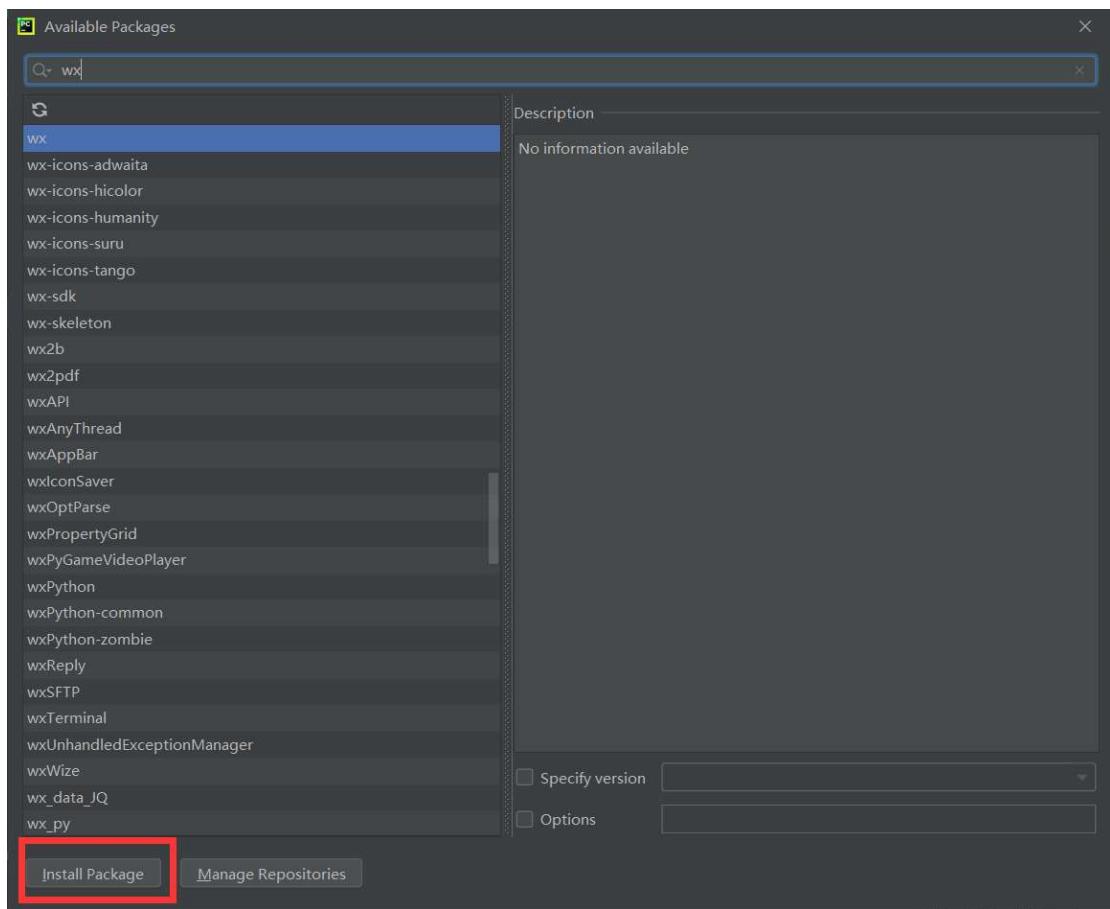


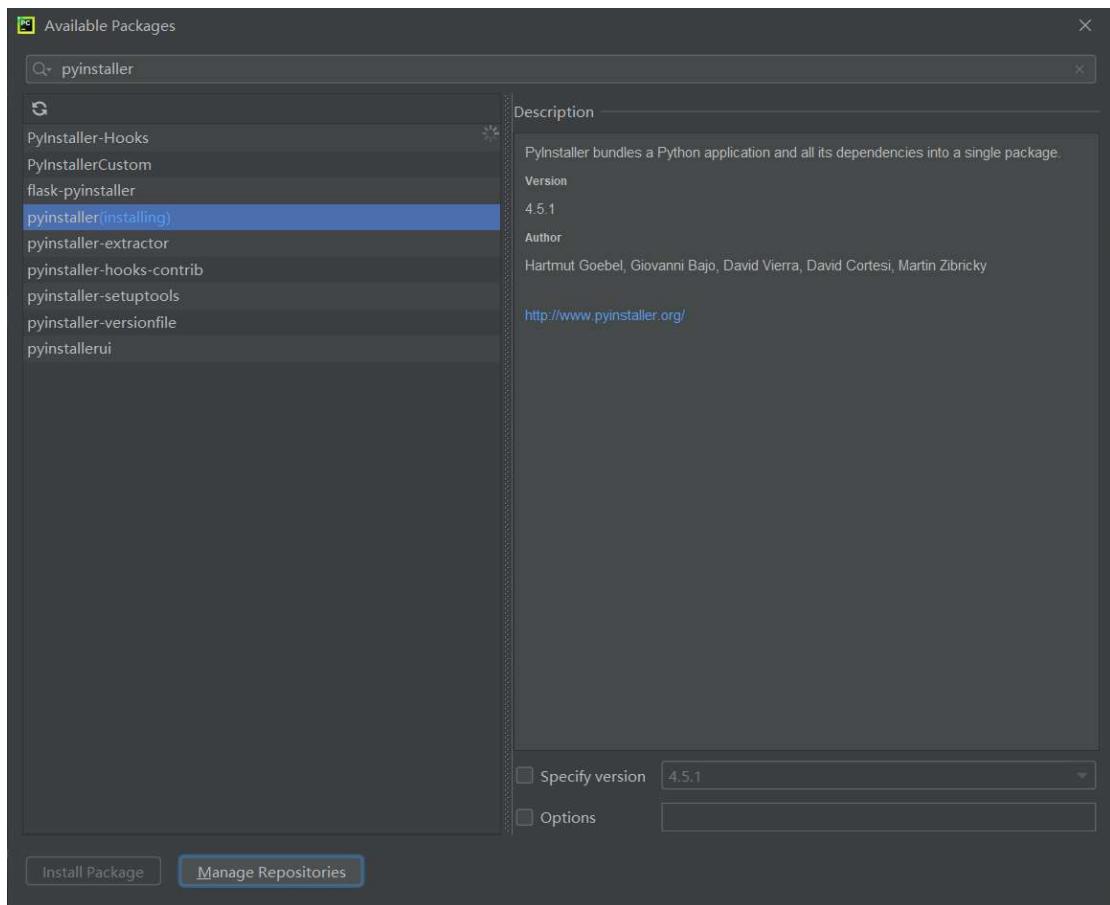
1.4. After selecting the compiler, click '+' to add the library



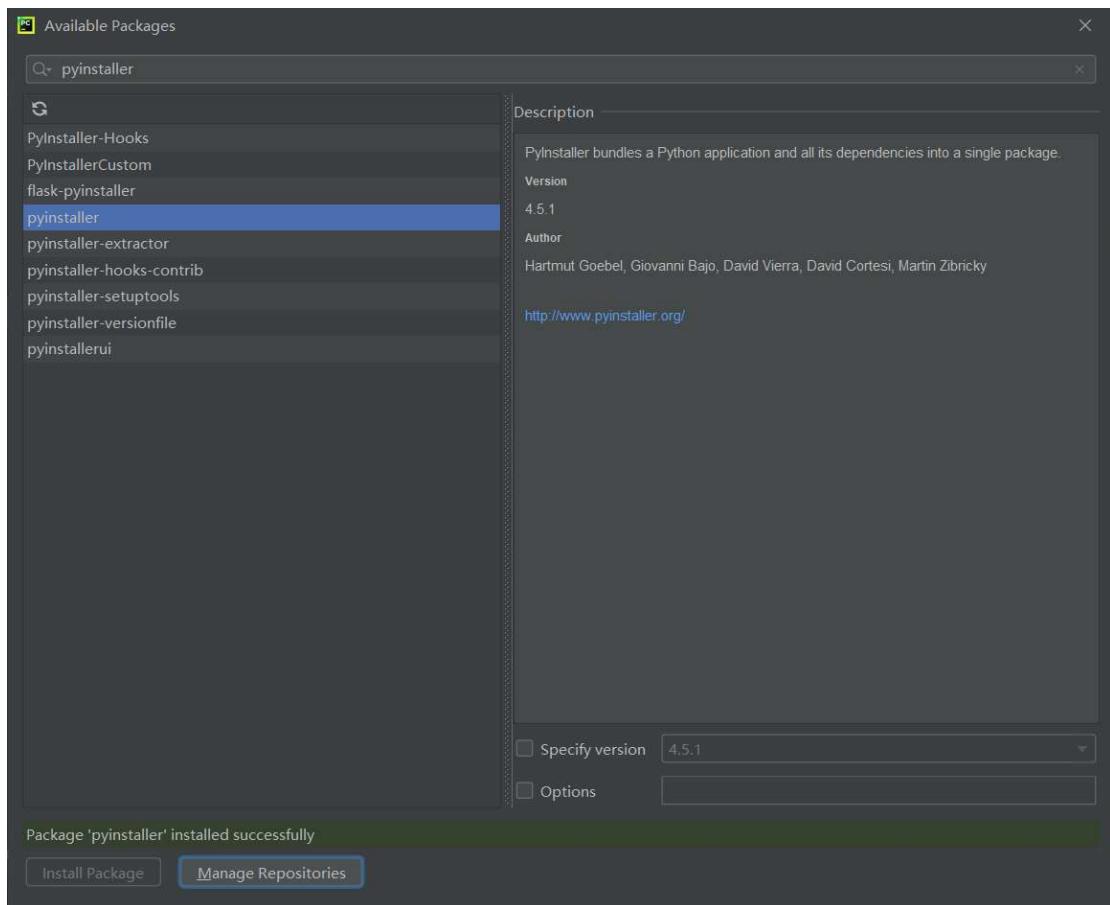


1.5. Select the pyinstaller library to download



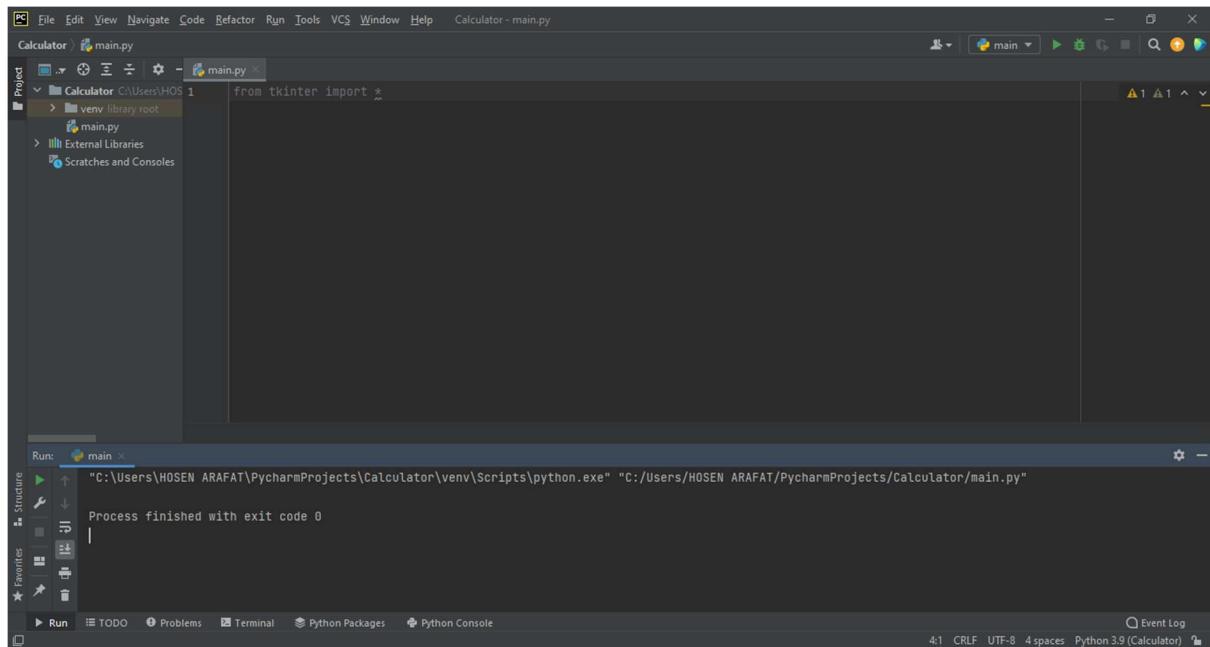


1.6. Download succeeded



2.Interface writing

1.1. Add the Tkinter library that comes with the python compiler



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator > main.py
Project venv library root main.py External Libraries Scratches and Consoles
from tkinter import *
```

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar says "Calculator - main.py". The left sidebar has a "Project" view showing a folder named "Calculator" containing "venv library root" and "main.py". The main code editor window contains the following Python code:

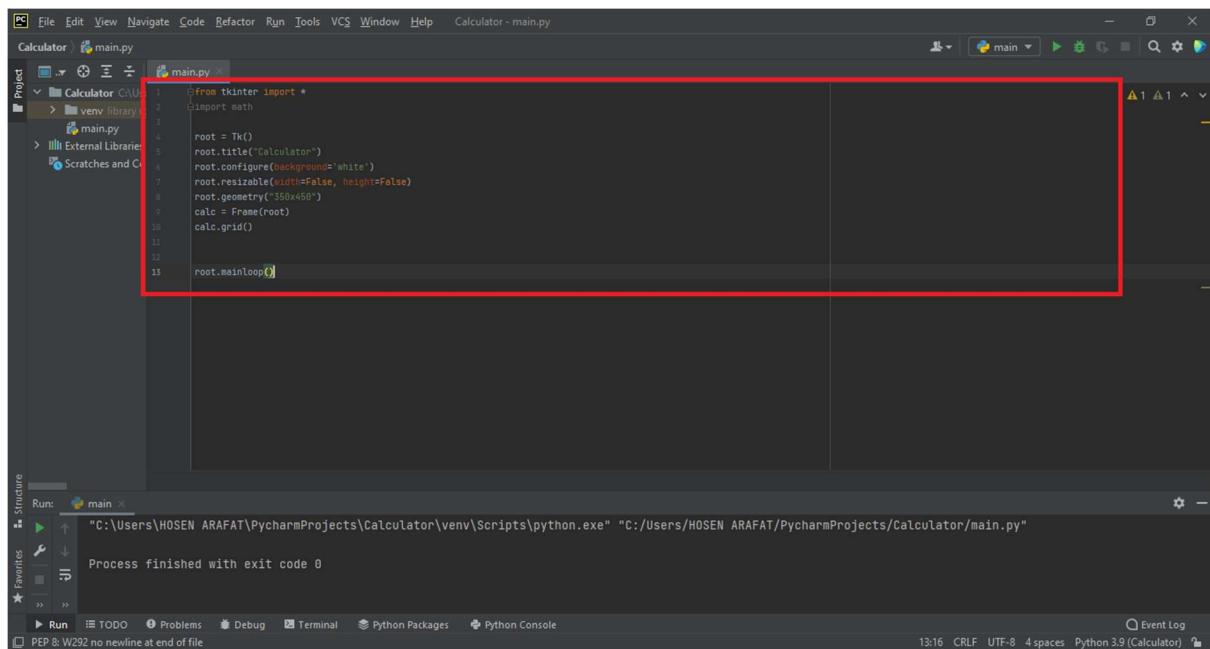
```
from tkinter import *
```

The code runs successfully, as indicated by the output in the "Run" tab:

```
"C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
Process finished with exit code 0
```

The bottom status bar shows the file path as "C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" and the Python version as "Python 3.9 (Calculator)".

1.2. Add a form, set its title, length, width, and display



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator > main.py
Project venv library root main.py External Libraries Scratches and Consoles
from tkinter import *
import math
root = Tk()
root.title("Calculator")
root.configure(background='white')
root.resizable(width=False, height=False)
root.geometry("350x450")
calc = Frame(root)
calc.grid()
root.mainloop()
```

The screenshot shows the PyCharm IDE interface, similar to the previous one. The code editor window now contains the following Python code, with the entire window configuration section highlighted by a red box:

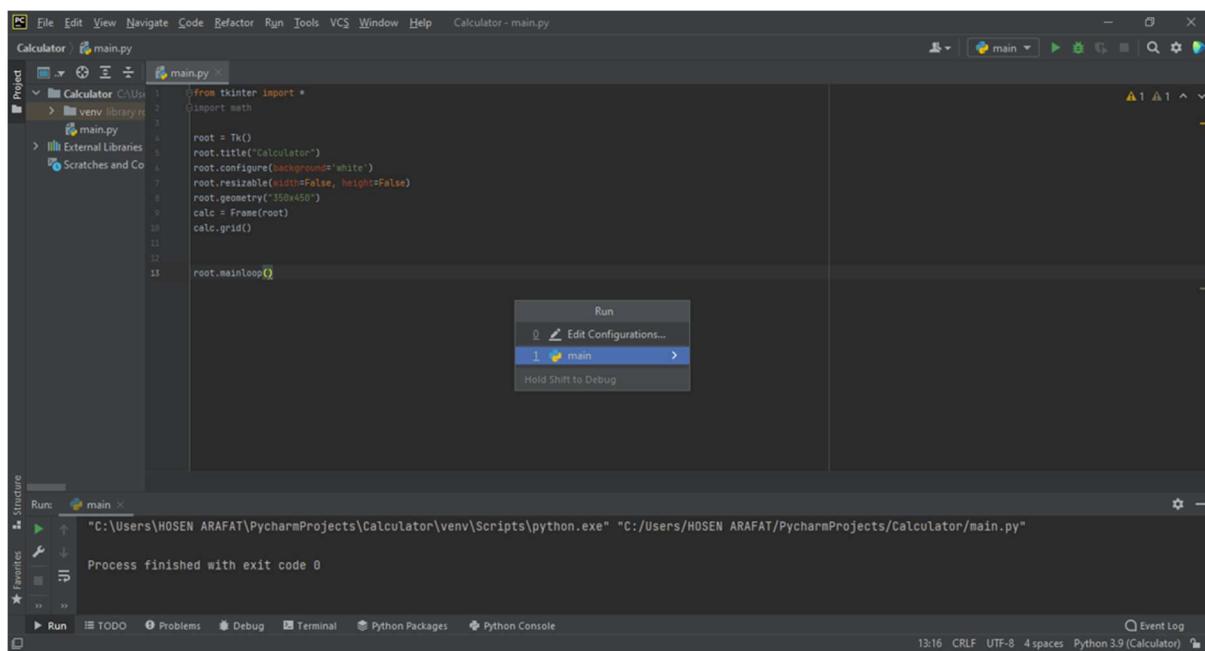
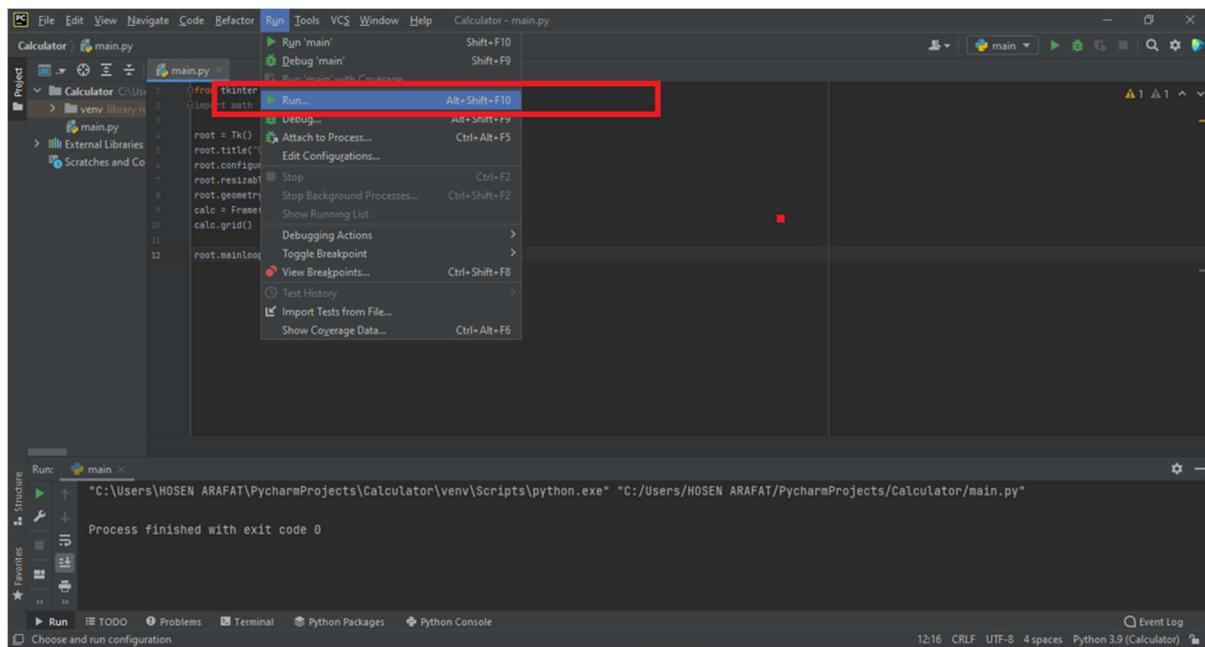
```
from tkinter import *
import math
root = Tk()
root.title("Calculator")
root.configure(background='white')
root.resizable(width=False, height=False)
root.geometry("350x450")
calc = Frame(root)
calc.grid()
root.mainloop()
```

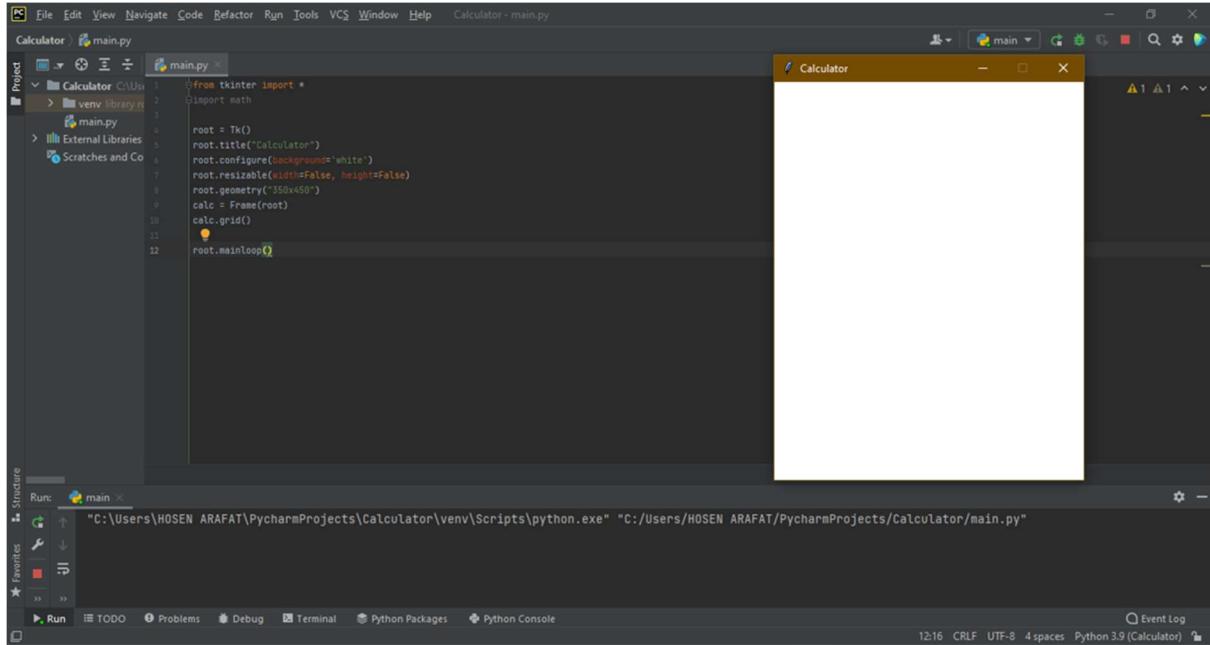
The code runs successfully, as indicated by the output in the "Run" tab:

```
"C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
Process finished with exit code 0
```

The bottom status bar shows the file path as "C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" and the Python version as "Python 3.9 (Calculator)".

1.3. Click Run -> run... And select the main.py file to compile





A screenshot of the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar says "Calculator - main.py". The left sidebar shows a project structure with a "Calculator" folder containing "main.py" and "venv library.r" files, along with "External Libraries" and "Scratches and Co". The main code editor window displays the following Python script:

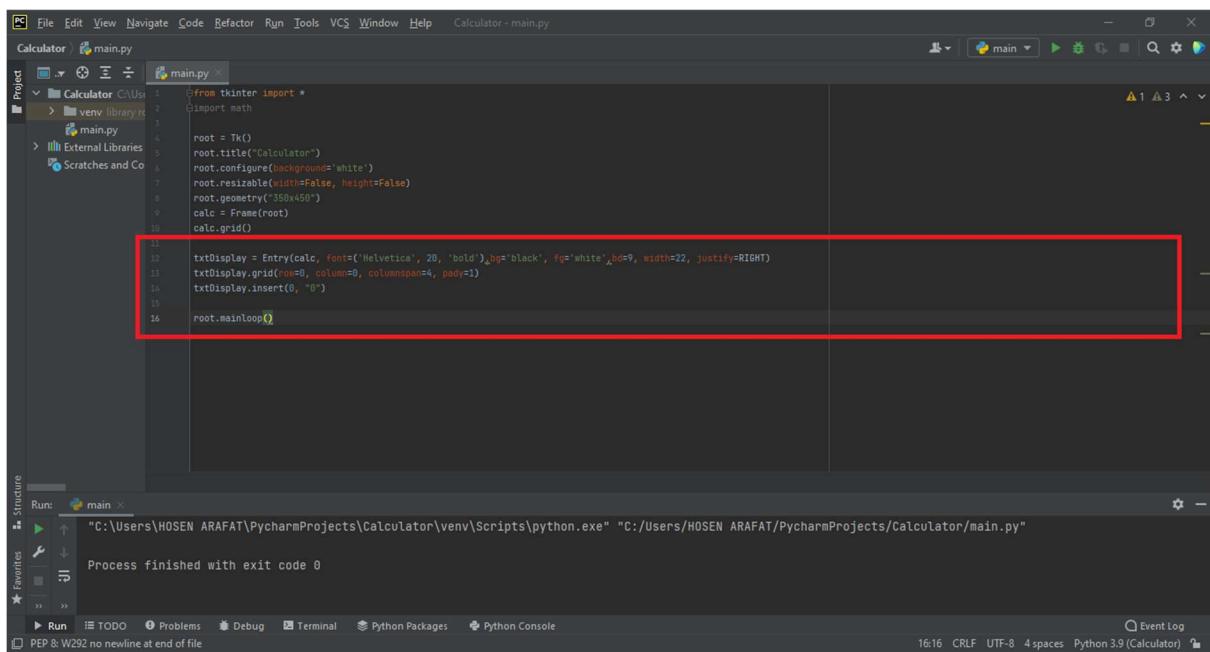
```
from tkinter import *
import math

root = Tk()
root.title("Calculator")
root.configure(background='white')
root.resizable(width=False, height=False)
root.geometry("350x450")
calc = Frame(root)
calc.grid()

root.mainloop()
```

The right side of the interface shows a preview window titled "Calculator" which is currently empty. Below the code editor is the "Run" tool bar with a green play button icon, a dropdown menu, and other run-related buttons. The status bar at the bottom shows the command "C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py" and the time 12:16.

1.4. Add input text box (display box)



A screenshot of the PyCharm IDE interface, similar to the previous one but with a red rectangular box highlighting the code from line 11 to 16. This highlights the addition of a text entry box (Entry) to the calculator application. The code in the editor is:

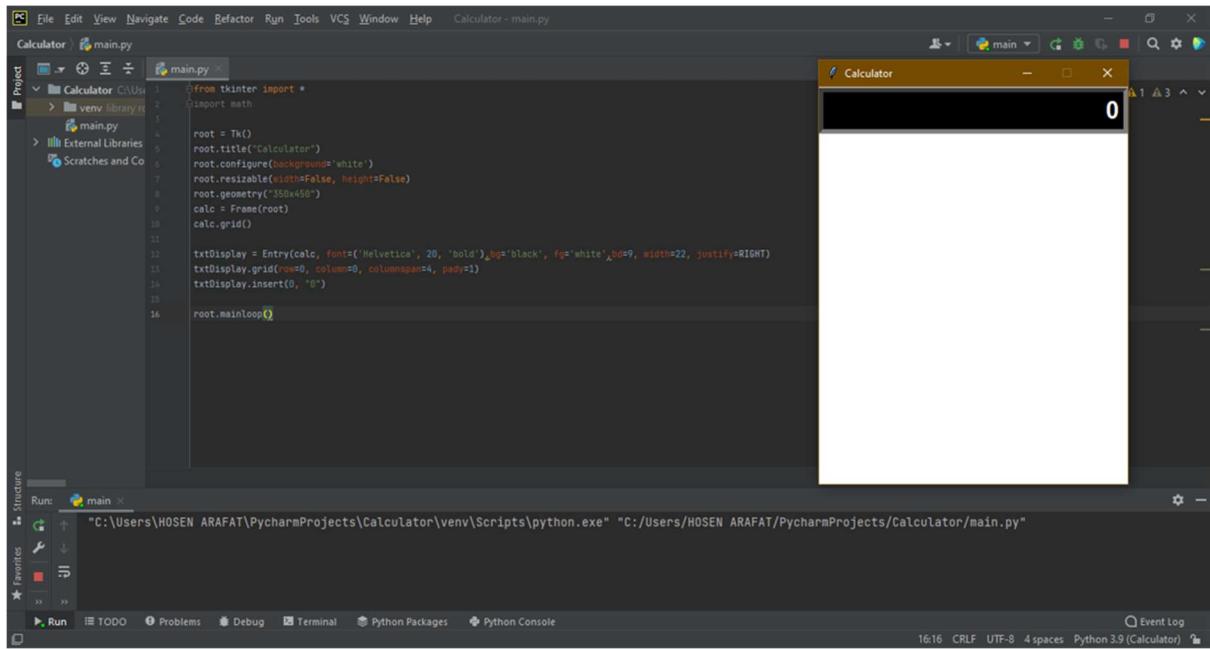
```
from tkinter import *
import math

root = Tk()
root.title("Calculator")
root.configure(background='white')
root.resizable(width=False, height=False)
root.geometry("350x450")
calc = Frame(root)
calc.grid()

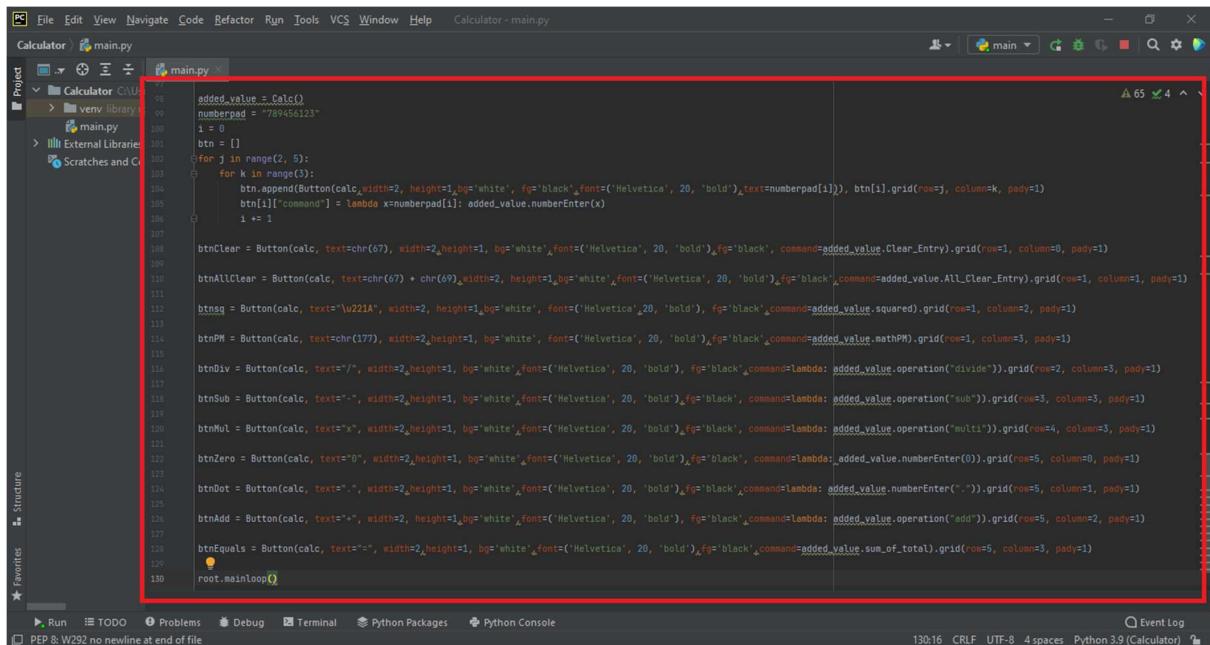
txtDisplay = Entry(calc, font='Helvetica', 20, 'bold')  
txtDisplay.grid(row=0, column=0, columnspan=4, pady=1)  
txtDisplay.insert(0, "0")

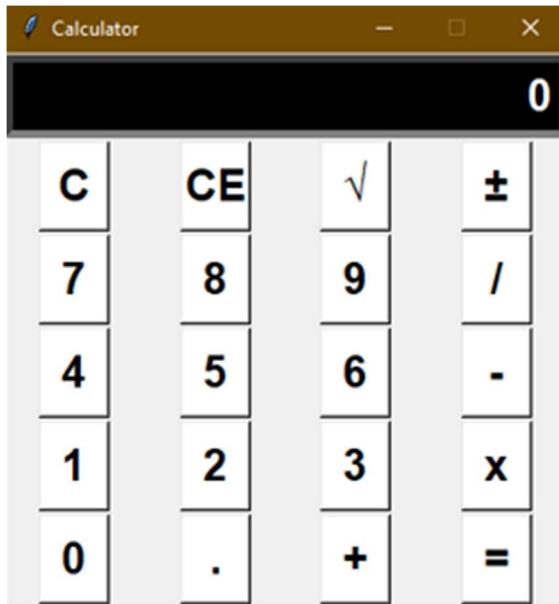
root.mainloop()
```

The rest of the interface is identical to the first screenshot, including the preview window, run toolbar, and status bar.



1.5. Add buttons for 1-9 and other operations





1.6. Add bigger length, width, and display

Screenshot of PyCharm showing the code for the calculator application. The code has several sections highlighted with red boxes:

```

from tkinter import *
import math

root = Tk()
root.title("Calculator")
root.configure(background='white')
root.resizable(width=False, height=False)
root.geometry("480x568+450+90")

calc = Frame(root)
calc.grid()

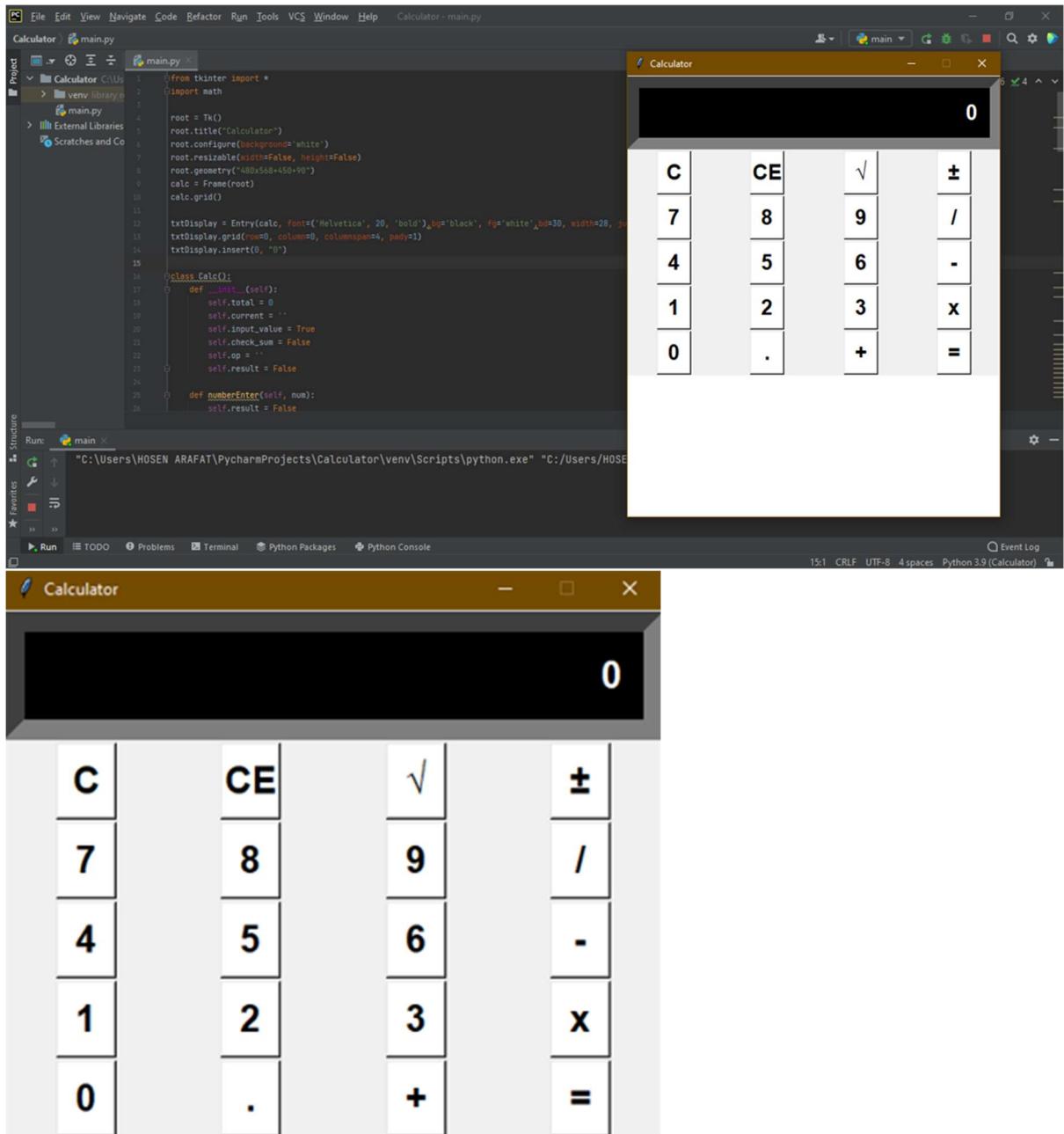
txtDisplay = Entry(calc, font=("", 20, "bold"), bg="black", fg="white", width=30, justify=RIGHT)
txtDisplay.grid(row=0, column=0, columnspan=4, padx=10)
txtDisplay.insert(0, "0")

class Calc():
    def __init__(self):
        self.total = 0
        self.current = ''
        self.input_value = True
        self.check_sum = False
        self.op = ''
        self.result = False

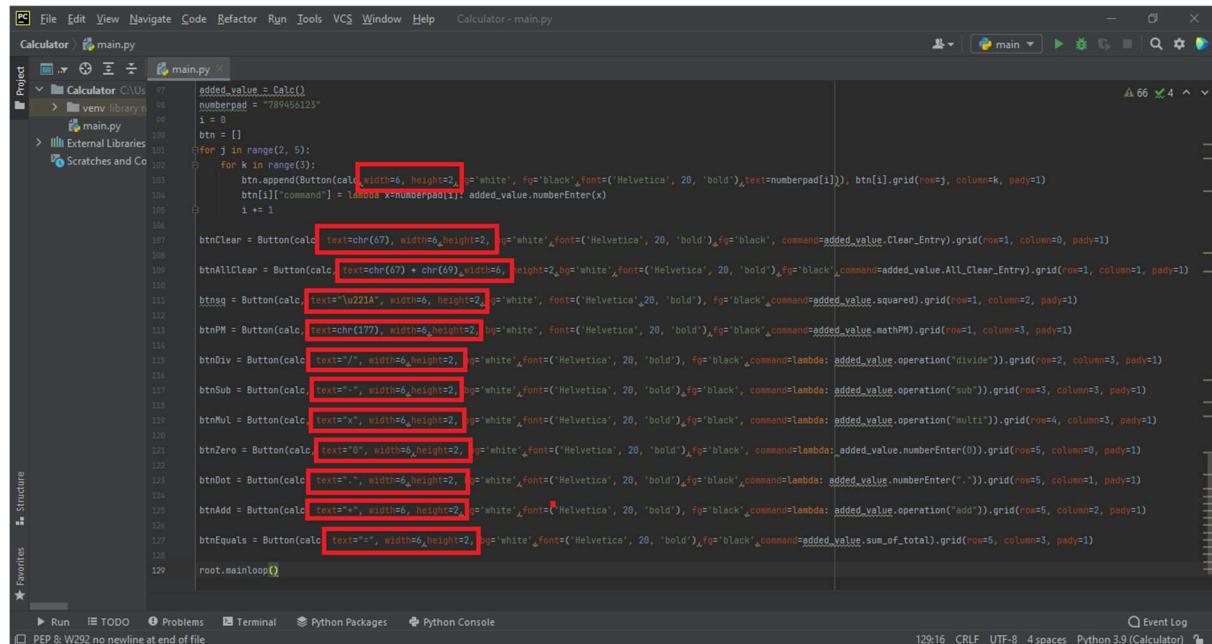
    def numberEnter(self, num):
        self.result = False
        firstnum = txtDisplay.get()
        secondnum = str(num)
        if self.input_value:
            self.current = secondnum
            self.input_value = False
        else:
            if secondnum == '.':
                if secondnum in firstnum:
                    pass
            else:
                self.current = firstnum + secondnum
        txtDisplay.delete(0, END)
        txtDisplay.insert(0, self.current)

    def sum(self, op):
        self.result = False
        if self.check_sum:
            self.current = float(self.current)
            if self.op == '+':
                self.total += self.current
            elif self.op == '-':
                self.total -= self.current
            elif self.op == '*':
                self.total *= self.current
            elif self.op == '/':
                self.total /= self.current
            self.input_value = True
            self.check_sum = False
            self.current = ''
            self.op = ''
            txtDisplay.delete(0, END)
            txtDisplay.insert(0, self.total)
        else:
            self.total = float(self.current)
            self.input_value = True
            self.check_sum = True
            self.op = op
            self.current = ''
    ...

```



1.7. Add bigger buttons for 1-9 and other operations



```
PC File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator > main.py
Project v Calculator核算
main.py
97     added_value = Calc()
98     numberpad = "789456123"
99
100    i = 0
101
102    btn = []
103
104    for j in range(2, 5):
105        for k in range(3):
106            btn.append(Button(calc, width=6, height=2, bg='white', fg='black', font=('Helvetica', 20, 'bold'), text=numberpad[i]), btn[i].grid(row=j, column=k, pady=1))
107            btn[i][k] = lambda x=numberpad[i]: added_value.numberEnter(x)
108            i += 1
109
110    btnClear = Button(calc, text=chr(67), width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=added_value.Clear_Entry).grid(row=1, column=0, pady=1)
111    btnAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=added_value.All_Clear_Entry).grid(row=1, column=1, pady=1)
112
113    btnSqrt = Button(calc, text="\u221a", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=added_value.squared).grid(row=1, column=2, pady=1)
114
115    btnPM = Button(calc, text="(\u0333)", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=added_value.mathPM).grid(row=1, column=3, pady=1)
116
117    btnDiv = Button(calc, text="/", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)
118
119    btnSub = Button(calc, text="-", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.operation("sub")).grid(row=3, column=3, pady=1)
120
121    btnMul = Button(calc, text="\u00d7", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.operation("multi")).grid(row=4, column=3, pady=1)
122
123    btnZero = Button(calc, text="0", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.numberEnter(0)).grid(row=5, column=0, pady=1)
124
125    btnDot = Button(calc, text=". ", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.numberEnter(". ")).grid(row=5, column=1, pady=1)
126
127    btnAdd = Button(calc, text="+", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=lambda: added_value.operation("add")).grid(row=5, column=2, pady=1)
128
129    btnEquals = Button(calc, text="=", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', command=added_value.sum_of_total).grid(row=5, column=3, pady=1)
130
131    root.mainloop()
```



1.8. Add buttons border for 1-9 and other operations

The screenshot shows the PyCharm IDE interface with the main.py file open. The code defines a calculator application with a 4x4 grid of buttons. The buttons are styled with a black border and white background. The application window displays a numeric keypad and basic arithmetic operators.

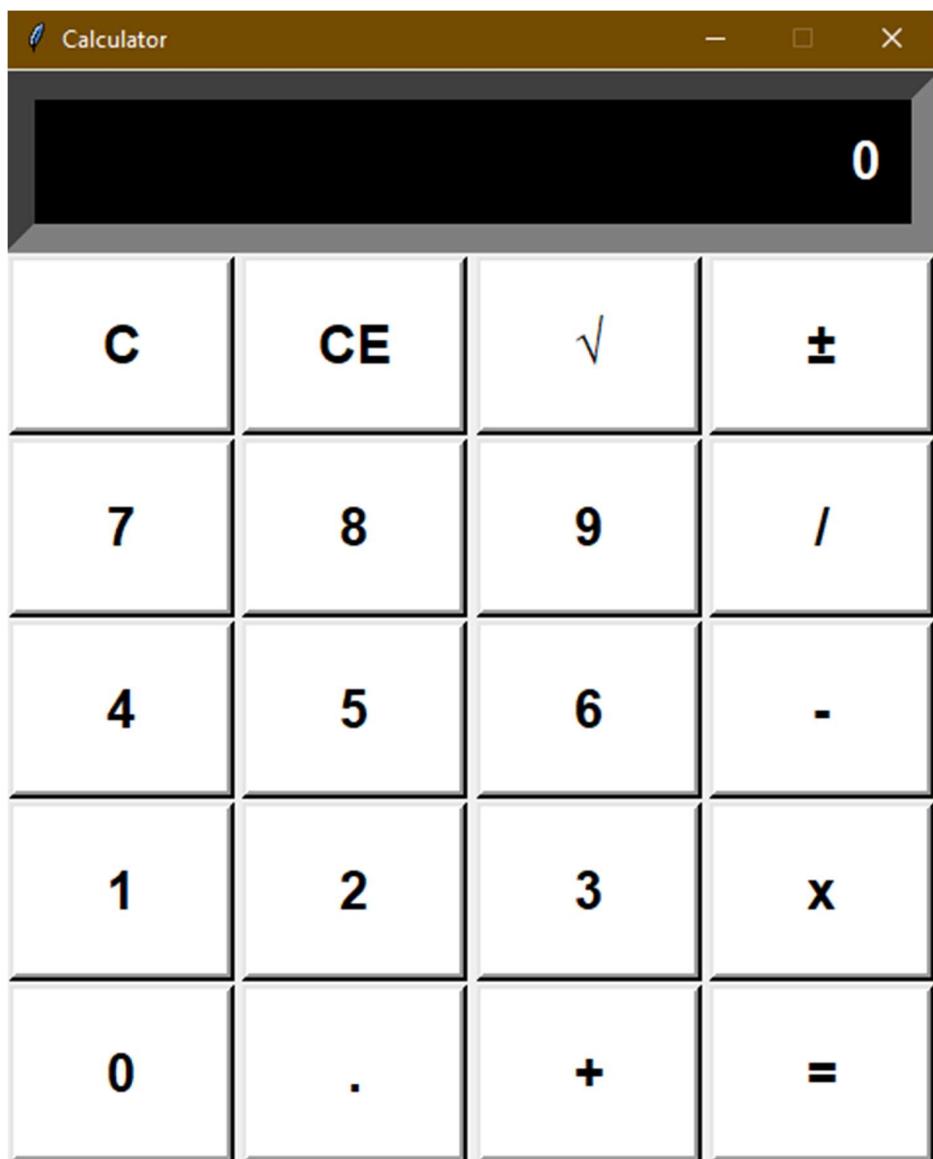
```

added_value = Calc()
numberpad = "789456123"
i = 0
btn = []
for j in range(2, 6):
    for k in range(3):
        btn.append(Button(calc, width=6, height=2, bg='white', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda x=numberpad[i]: added_value.numberEnter(x)))
        btn[i]["command"] = lambda x=numberpad[i]: added_value.numberEnter(x)
        i += 1

btnClear = Button(calc, text=chr(67), width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.Clear_Entry).grid(row=1, column=0, pady=1)
btnAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.All_Clear_Entry).grid(row=1, column=1, pady=1)
btnsq = Button(calc, text="\u00b2", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.mathSqd).grid(row=1, column=2, pady=1)
btnPM = Button(calc, text=chr(177), width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.mathPM).grid(row=1, column=3, pady=1)
btnDiv = Button(calc, text="/", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)
btnSub = Button(calc, text="-", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("sub")).grid(row=3, column=3, pady=1)
btnMul = Button(calc, text="*", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("mult")).grid(row=4, column=3, pady=1)
btnZero = Button(calc, text="0", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.numberEnter(0)).grid(row=5, column=0, pady=1)
btnDot = Button(calc, text=". ", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.numberEnter(".")).grid(row=5, column=1, pady=1)
btnAdd = Button(calc, text="+", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("add")).grid(row=5, column=2, pady=1)
btnEquals = Button(calc, text="=", width=6, height=2, bg='white', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.sum_of_total).grid(row=5, column=3, pady=1)

root.mainloop()

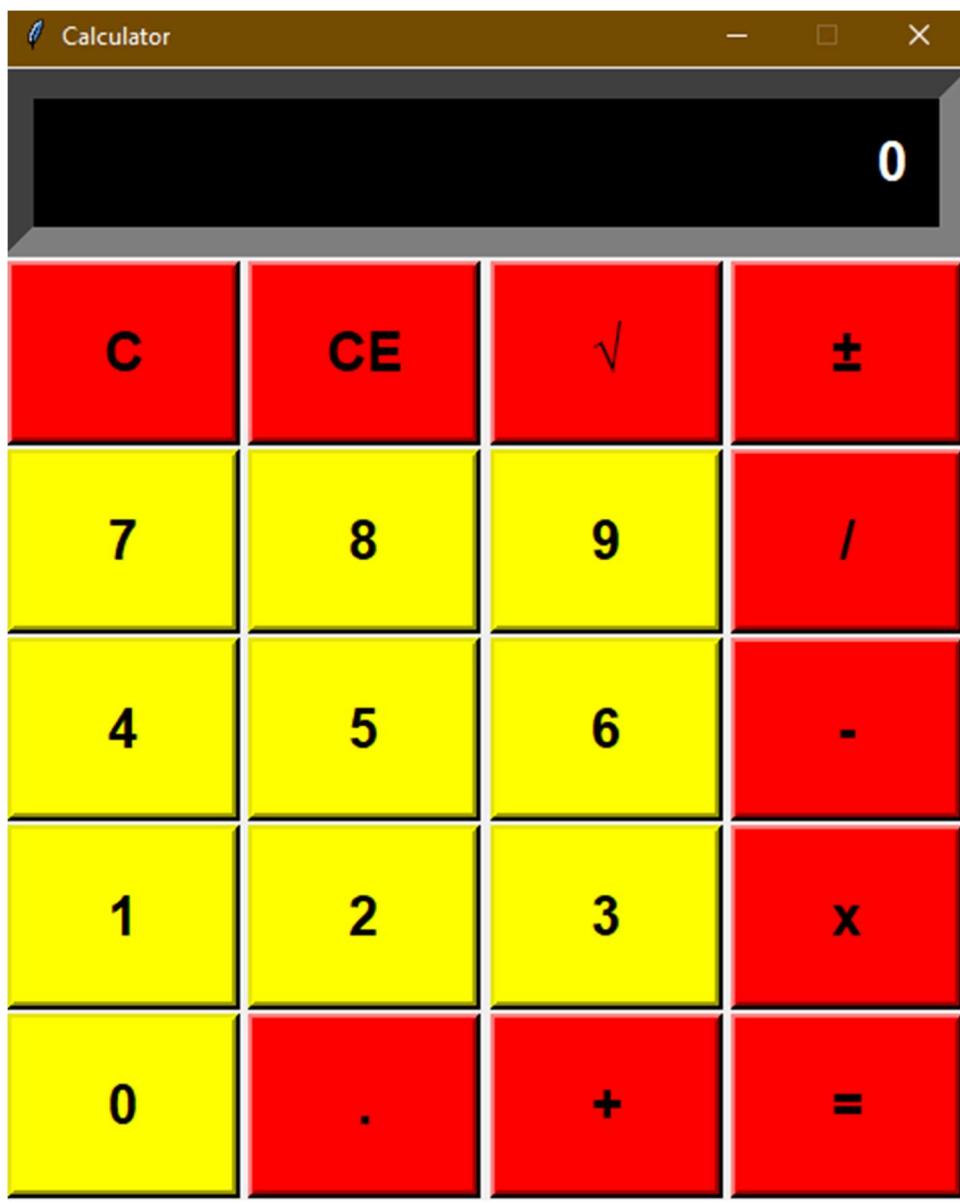
```



1.9. Add buttons background color for 1-9 and other operations

```
PC File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Project: Calculator C:\Users\HOSEN\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN/PycharmProjects/Calculator/main.py"
main.py
added_value = Calc()
numberpad = "789456123"
1 = 0
btn = []
for j in range(2, 5):
    for k in range(3):
        btn.append(Button(calc, width=6, height=2, bg='yellow', fg='black', font=('Helvetica', 20, 'bold'), bd=4, text=numberpad[i]), btn[i].grid(row=j, column=k, padx=1))
        i += 1
        btnClear = Button(calc, text=chr(67), width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.Clear_Entry).grid(row=1, column=0, pady=1)
        btnAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.All_Clear_Entry).grid(row=1, column=1, pady=1)
        btnsq = Button(calc, text="\u221a", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.squared).grid(row=1, column=2, padx=1)
        btnPM = Button(calc, text=chr(177), width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.mathPM).grid(row=1, column=3, pady=1)
        btnDiv = Button(calc, text="/", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)
        btnSub = Button(calc, text="-", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("sub")).grid(row=3, column=3, pady=1)
        btnMul = Button(calc, text="*", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("multi")).grid(row=4, column=3, pady=1)
        btnZero = Button(calc, text="0", width=6, height=2, bg='yellow', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.numberEnter(0)).grid(row=5, column=0, pady=1)
        btnDot = Button(calc, text=". ", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.numberEnter(". ")).grid(row=5, column=1, pady=1)
        btnAdd = Button(calc, text="+", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("add")).grid(row=5, column=2, pady=1)
        btnEquals = Button(calc, text="=", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.sum_of_total).grid(row=5, column=3, pady=1)
root.mainloop()
```

```
PC File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Project: Calculator C:\Users\HOSEN\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN/PycharmProjects/Calculator/main.py"
main.py
btncAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.All_Clear_Entry).grid(row=1, column=0, pady=1)
btncsq = Button(calc, text="\u221a", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.squared).grid(row=1, column=1, padx=1)
btncPM = Button(calc, text=chr(177), width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.mathPM).grid(row=1, column=2, padx=1)
btncDiv = Button(calc, text="/", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)
btncSub = Button(calc, text="-", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("sub")).grid(row=3, column=3, pady=1)
btncMul = Button(calc, text="*", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("multi")).grid(row=4, column=3, pady=1)
btncZero = Button(calc, text="0", width=6, height=2, bg='yellow', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.numberEnter(0)).grid(row=5, column=0, pady=1)
btncDot = Button(calc, text=". ", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.numberEnter(". ")).grid(row=5, column=1, pady=1)
btncAdd = Button(calc, text="+", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda: added_value.operation("add")).grid(row=5, column=2, pady=1)
btncEquals = Button(calc, text="=", width=6, height=2, bg='red', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=added_value.sum_of_total).grid(row=5, column=3, pady=1)
root.mainloop()
```



3.logic design

3.1 Write the number enter function of the input box, and add the change function to the 1-9 button. When the button inputs a number, it will be updated in the input box in real time.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator main.py
Project
Calculator C:\US\venv\library\main.py
External Libraries
Scratches and Consoles
10     self.total = 0
11
12     self.current = ''
13
14     self.input_value = True
15
16     self.check_sum = False
17
18     self.op = ''
19
20     self.result = False
21
22
23
24
25     def numberEnter(self, num):
26         self.result = False
27         firstnum = txtDisplay.get()
28         secondnum = str(num)
29         if self.input_value:
30             self.current = secondnum
31             self.input_value = False
32         else:
33             if secondnum == '.':
34                 if secondnum in firstnum:
35                     return
36             self.current = firstnum + secondnum
37         self.display(self.current)
38
39
40     def sum_of_total(self):
41         self.result = True
42         self.current = float(self.current)
43         if self.check_sum == True:
44             self.valid_function()
45         else:
```

The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Calculator - main.py:** The current file being edited.
- Project Tree:** Shows the project structure with files like main.py, venv, and external libraries.
- Code Editor:** The main.py code is displayed, containing methods for clearing entry, performing math operations, and handling button clicks for a calculator. A red box highlights the button creation loop from lines 97 to 116.
- Toolbars and Status Bar:** Standard PyCharm toolbars and status bar at the bottom.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py

Calculator main.py
Project
Calculator C:\Users\...
> venv\lib\site-packages
External Libraries
Scratches and Co

main.py

83     def All_Clear_Entry(self):
84         self.Clear_Entry()
85         self.total = 0
86
87     def mathPM(self):
88         self.result = False
89         self.current = -(float(txtDisplay.get()))
90         self.display(self.current)
91
92     def squared(self):
93         self.result = False
94         self.current = math.sqrt(float(txtDisplay.get()))
95         self.display(self.current)
96
97     added_value = Calc()
98     numberpad = "789456123"
99     i = 0
100    btns = []
101
102    for j in range(2, 5):
103        for k in range(3):
104            btn = Button(calc, width=6, height=2, bg='yellow', fg='black', font=('Helvetica', 20, 'bold'), bd=4, command=lambda x=numberpad[i]: added_value.numberEnter(x))
105            btn.grid(row=j, column=k, pady=1)
106            i += 1
107
108    btnClear = Button(calc, text=chr(67), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.Clear_Entry).grid(row=1, column=0, pady=1)
109
110    btnAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.All_Clear_Entry).grid(row=1, column=1, pady=1)
111
112    btnsq = Button(calc, text="\u00b2", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.squared).grid(row=1, column=2, pady=1)
113
114    btnPM = Button(calc, text=chr(177), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.mathPM).grid(row=1, column=3, pady=1)
115
116    btnDiv = Button(calc, text="/", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)

Calc
Run TODO Problems Terminal Python Packages Python Console
83:1 CRLF UTF-8 4 spaces Python 3.9 (Calculator)
Event Log
```

The screenshot shows the PyCharm IDE interface. On the left, the Project tool window displays a 'Calculator' project with files like 'main.py'. The main editor window shows the following Python code:

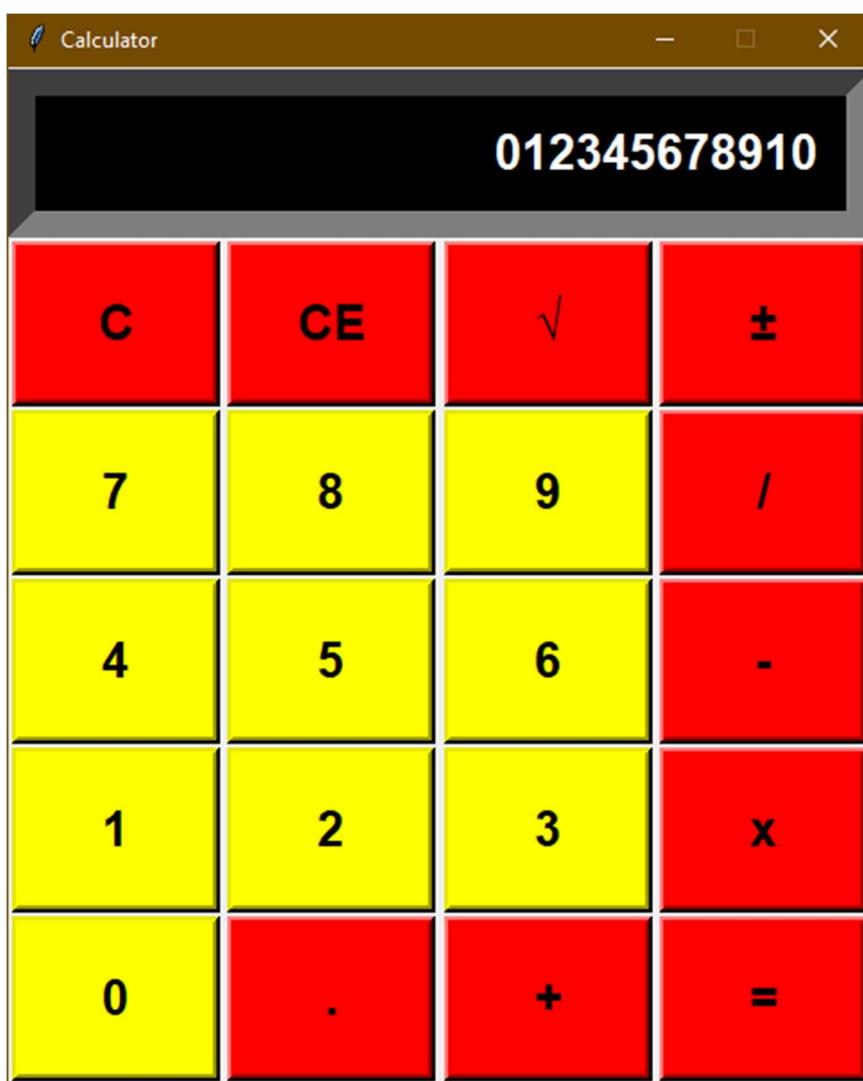
```
class Calc:
    def __init__(self):
        self.op = ''
        self.result = False

    def numberEnter(self, num):
        self.result = False
        firstnum = txtDisplay.get()
        secondnum = str(num)
        if self.input_value:
            self.current = secondnum
            self.input_value = False
        else:
            if secondnum == '.':
                if secondnum in firstnum:
                    return
            self.current = firstnum + secondnum
            self.display(self.current)

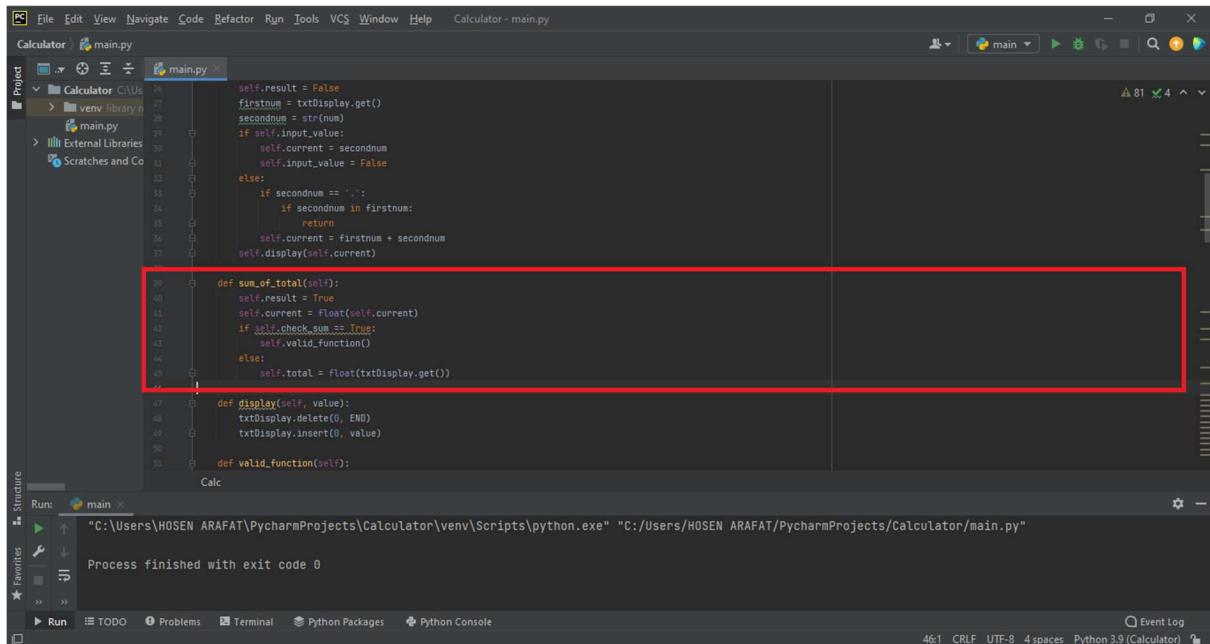
    def sum_of_total(self):
        self.result = True
        self.current = float(self.current)
```

On the right, a window titled 'Calculator' is displayed, showing the number '012345678910' on its digital display. The calculator has a 4x4 grid of buttons. The layout is as follows:

C	CE	√	±
7	8	9	/
4	5	6	-
1	2	3	x
0	.	+	=



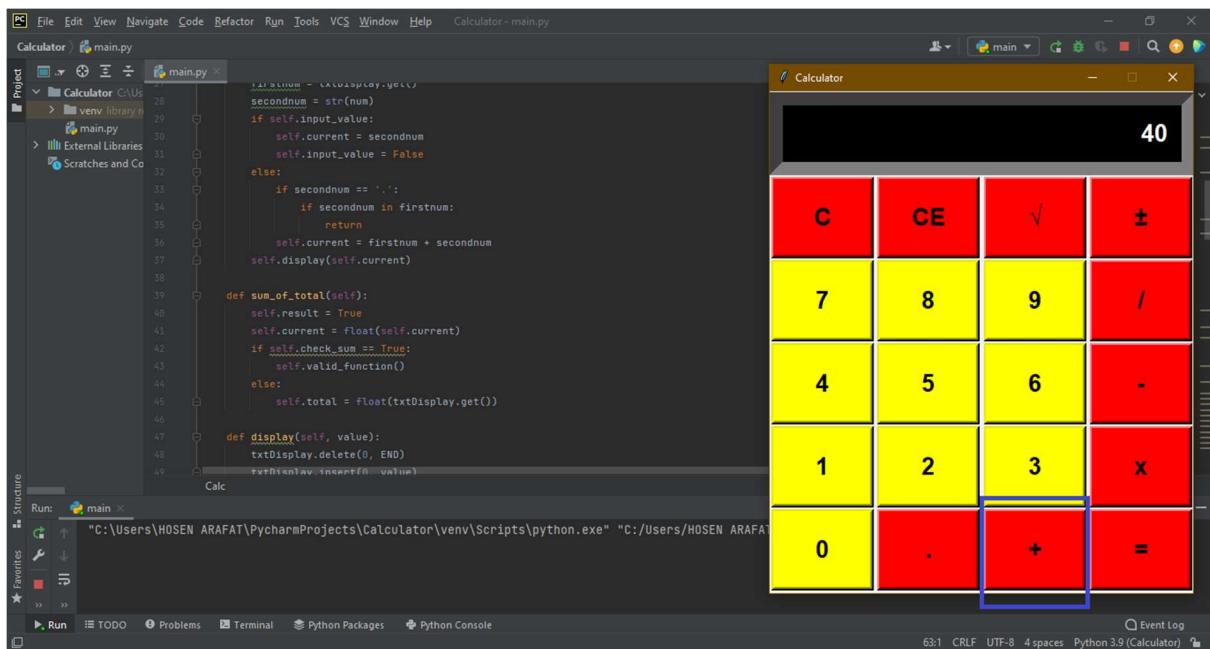
3.2 Write the sum of total function.



```

Calculator > main.py
Project: Calculator C:\Us
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator(main.py)
main.py
External Libraries
Scratches and Co
26     self.result = False
27     firstnum = txtDisplay.get()
28     secondnum = str(num)
29     if self.input_value:
30         self.current = secondnum
31         self.input_value = False
32     else:
33         if secondnum == '.':
34             if secondnum in firstnum:
35                 return
36         self.current = firstnum + secondnum
37         self.display(self.current)
38
39     def sum_of_total(self):
40         self.result = True
41         self.current = float(self.current)
42         if self.check_sum == True:
43             self.valid_function()
44         else:
45             self.total = float(txtDisplay.get())
46
47     def display(self, value):
48         txtDisplay.delete(0, END)
49         txtDisplay.insert(0, value)
50
51     def valid_function(self):
52
Calc
Run: main
C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
Process finished with exit code 0
Event Log
461 CRLF UTF-8 4 spaces Python 3.9 (Calculator)

```



```

Calculator > main.py
Project: Calculator C:\Us
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator(main.py)
main.py
External Libraries
Scratches and Co
26     self.result = False
27     firstnum = txtDisplay.get()
28     secondnum = str(num)
29     if self.input_value:
30         self.current = secondnum
31         self.input_value = False
32     else:
33         if secondnum == '.':
34             if secondnum in firstnum:
35                 return
36         self.current = firstnum + secondnum
37         self.display(self.current)
38
39     def sum_of_total(self):
40         self.result = True
41         self.current = float(self.current)
42         if self.check_sum == True:
43             self.valid_function()
44         else:
45             self.total = float(txtDisplay.get())
46
47     def display(self, value):
48         txtDisplay.delete(0, END)
49         txtDisplay.insert(0, value)
50
51     def valid_function(self):
52
Calc
Run: main
C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
Process finished with exit code 0
Event Log
631 CRLF UTF-8 4 spaces Python 3.9 (Calculator)

```

Calculator > main.py

```

27         self.current = txtDisplay.get()
28     secondnum = str(num)
29     if self.input_value:
30         self.current = secondnum
31         self.input_value = False
32     else:
33         if secondnum == '.':
34             if secondnum in firstnum:
35                 return
36             self.current = firstnum + secondnum
37             self.display(self.current)
38
39     def sum_of_total(self):
40         self.result = True
41         self.current = float(self.current)
42         if self.check_sum == True:
43             self.valid_function()
44         else:
45             self.total = float(txtDisplay.get())
46
47     def display(self, value):
48         txtDisplay.delete(0, END)
49         txtDisplay.insert(0, value)
50     Calc > sum_of_total() > if self.check_sum == True

```

Calculator

The calculator application shows the digit '5' in the top right corner of the display.

Calculator > main.py

```

26         self.result = False
27     firstnum = txtDisplay.get()
28     secondnum = str(num)
29     if self.input_value:
30         self.current = secondnum
31         self.input_value = False
32     else:
33         if secondnum == '.':
34             if secondnum in firstnum:
35                 return
36             self.current = firstnum + secondnum
37             self.display(self.current)
38
39     def sum_of_total(self):
40         self.result = True
41         self.current = float(self.current)
42         if self.check_sum == True:
43             self.valid_function()
44         else:
45             self.total = float(txtDisplay.get())
46
47     def display(self, value):
48         txtDisplay.delete(0, END)
49         txtDisplay.insert(0, value)
50     def valid_function(self):
51
52     Calc

```

Calculator

The calculator application shows the result '90.0' in the top right corner of the display.

3.3. Write the Clear Entry and All Clear Entry function.

The screenshot shows the PyCharm IDE interface with the code editor open to a file named 'main.py'. A red box highlights the following code block:

```
def Clear_Entry(self):
    self.result = False
    self.current = "0"
    self.display(0)
    self.input_value = True

def All_Clear_Entry(self):
    self.Clear_Entry()
    self.total = 0
```

The code defines two methods: 'Clear_Entry' which sets the result to False, current to '0', displays '0', and sets input_value to True; and 'All_Clear_Entry' which calls 'Clear_Entry' and sets total to 0. The code editor shows syntax highlighting and a code completion dropdown. The bottom status bar indicates the run configuration and Python version.

The screenshot shows the PyCharm IDE interface with the code editor open to a file named 'main.py'. A red box highlights the following code block:

```
btnClear = Button(calc, text=chr(67) + chr(67), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.Clear_Entry().grid(row=1, column=0, pady=1))
btnAllClear = Button(calc, text=chr(67) + chr(69), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.All_Clear_Entry().grid(row=1, column=1, pady=1))

btNSQ = Button(calc, text=u'\u00d7', width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.squared).grid(row=1, column=2, pady=1)
btPM = Button(calc, text=chr(177), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.mathPM).grid(row=1, column=3, pady=1)

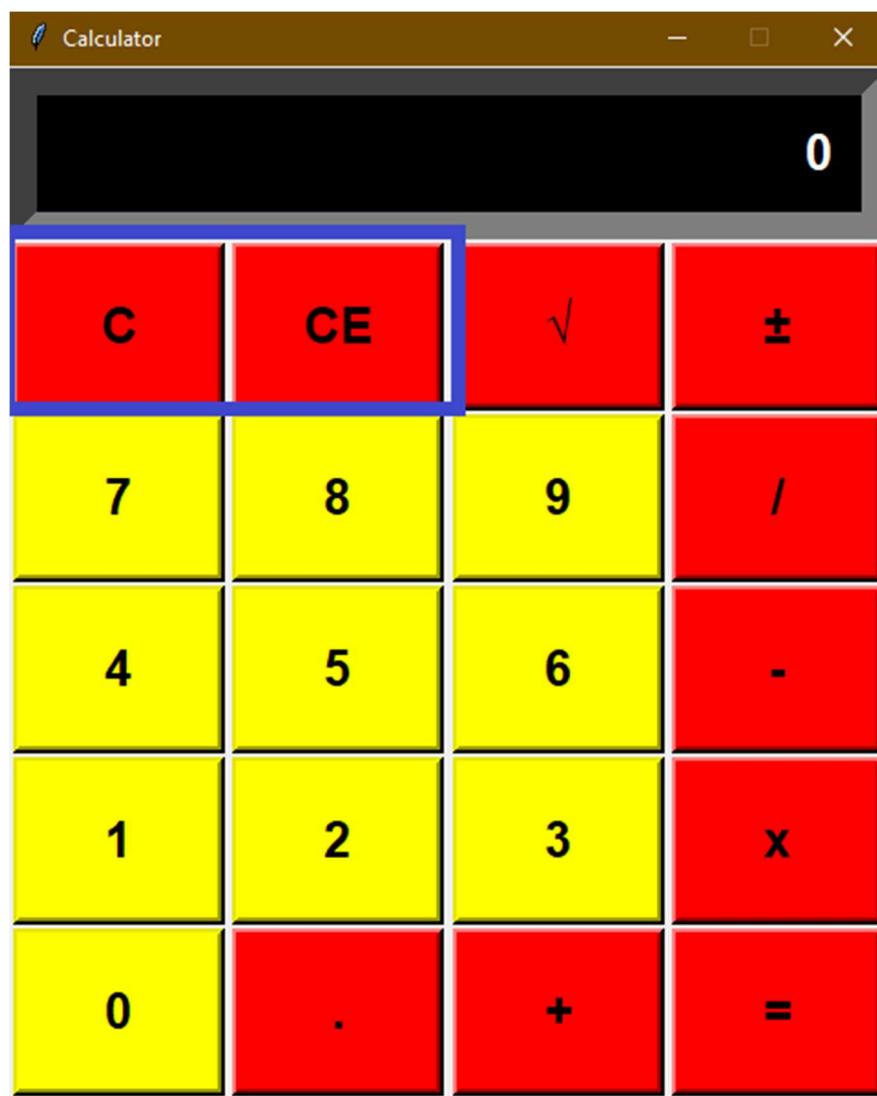
btDiv = Button(calc, text="/", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, pady=1)
btSub = Button(calc, text="-", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("sub")).grid(row=3, column=3, pady=1)
btMul = Button(calc, text="x", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("multi")).grid(row=4, column=3, pady=1)
btZero = Button(calc, text="0", width=6, height=2, bg='yellow', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.numberEnter(0)).grid(row=5, column=0, pady=1)
btDot = Button(calc, text=". ", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.numberEnter(". ")).grid(row=5, column=1, pady=1)
btAdd = Button(calc, text="+", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("add")).grid(row=5, column=2, pady=1)
btEquals = Button(calc, text="=", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.sum_of_total).grid(row=5, column=3, pady=1)

root.mainloop()
```

The code creates various buttons for a calculator application, each with specific text, dimensions, colors, and fonts. The buttons are arranged in a grid layout across five rows and four columns. The bottom status bar indicates the run configuration and Python version.

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** Calculator
- Code Editor:** The main.py file contains Python code for a calculator application. The code defines various buttons (C, CE, sqrt, ±, 7, 8, 9, /, 4, 5, 6, -), numeric buttons (1, 2, 3, x), zero, decimal point, addition, subtraction, multiplication, division, and equals. It also includes a command button and a clear button.
- Run Tab:** The run configuration is set to "main". The terminal shows the command: "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/venv/Scripts/python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py".
- Calculator Application:** A separate window titled "Calculator" is open, displaying a 4x4 grid of buttons. The buttons are colored as follows:
 - Row 1: C (red), CE (red), √ (yellow), ± (red)
 - Row 2: 7 (yellow), 8 (yellow), 9 (yellow), / (red)
 - Row 3: 4 (yellow), 5 (yellow), 6 (yellow), - (red)
 - Row 4: 1 (yellow), 2 (yellow), 3 (yellow), x (red)
 - Row 5: 0 (yellow), . (red), + (red), = (red)A blue box highlights the first row of buttons (C, CE, √, ±).



3.4 Compiling calculation function add, subtract, multiply and divide, and equal function, and write code, click the button to call the corresponding function

The screenshot shows the PyCharm IDE interface with the following details:

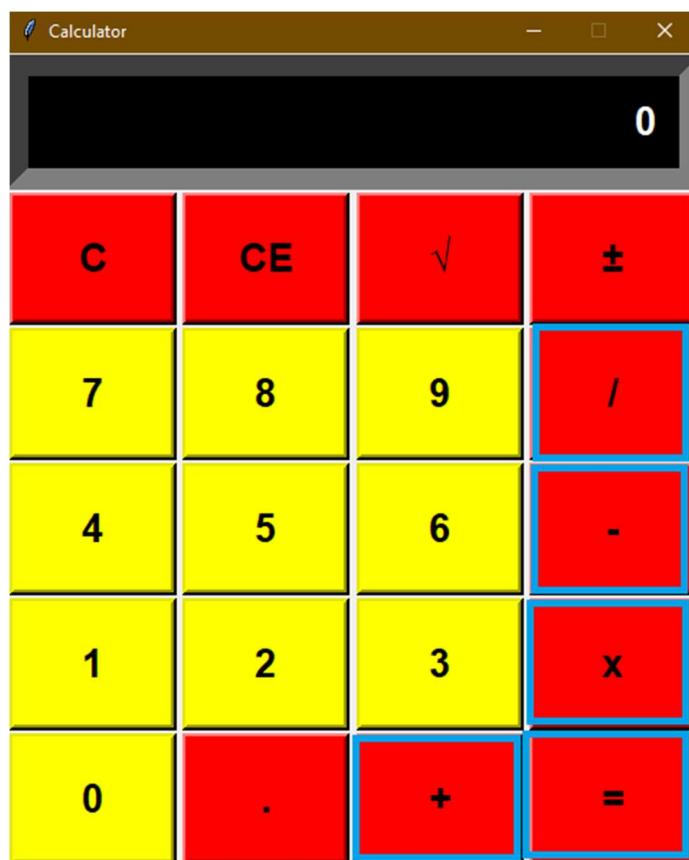
- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Calculator - main.py:** The current file being edited.
- Project Tree:** Shows the project structure with files like `Calculator`, `main.py`, and `venv/lib/python3.9/site-packages`.
- Code Editor:** The `main.py` code is displayed, containing button definitions for a calculator. Several lines of code are highlighted with red boxes:
 - Line 116: `btnDiv = Button(calc, text=chr(67), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("divide")).grid(row=2, column=3, padx=1)`
 - Line 117: `btnSub = Button(calc, text="-", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("sub")).grid(row=3, column=3, padx=1)`
 - Line 118: `btnMul = Button(calc, text="*", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("multi")).grid(row=4, column=3, padx=1)`
 - Line 119: `btnAdd = Button(calc, text="+", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.operation("add")).grid(row=5, column=2, padx=1)`
 - Line 120: `btnEquals = Button(calc, text="=", width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=added_value.sum_of_total).grid(row=5, column=3, padx=1)`
- Run Tab:** Shows the run configuration: "main" with the command "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/venv/Scripts/python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py".
- Output Tab:** Shows the message "Process finished with exit code 0".
- Bottom Status Bar:** Displays the Python version as 3.9 (Calculator).

The screenshot shows the PyCharm IDE interface with a project named 'Calculator' open. The main.py file is displayed in the editor, containing Python code for a calculator class. The code includes methods for validating functions and performing operations like addition, subtraction, multiplication, and division. A floating terminal window shows the output of the application, which is a graphical calculator with a red and yellow color scheme. The calculator has buttons for C, CE, square root, plus-minus, 7, 8, 9, division, 4, 5, 6, minus, 1, 2, 3, multiplication, 0, decimal point, plus, and equals.

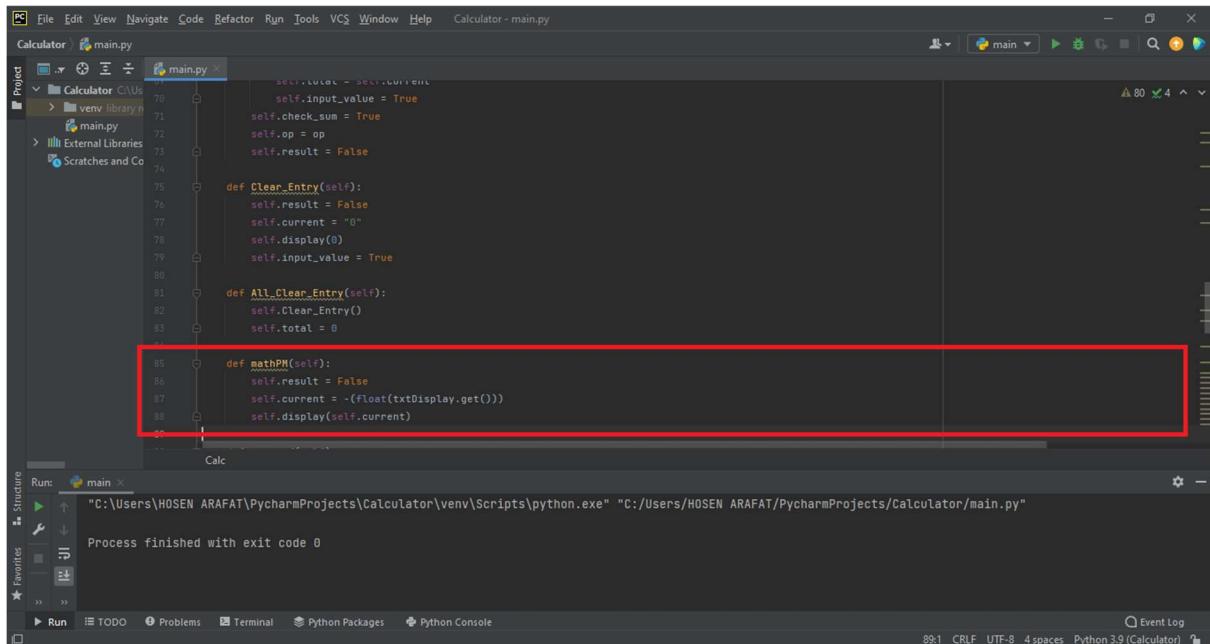
```
calculator.py, line 49, in calculate
    txtDisplay.insert(0, value)

calculator.py, line 51, in valid_function
    def valid_function(self):
        if self.op == "add":
            self.total += self.current
        if self.op == "sub":
            self.total -= self.current
        if self.op == "multi":
            self.total *= self.current
        if self.op == "divide":
            self.total /= self.current
        self.input_value = True
        self.check_sum = False
        self.display(self.total)
    
```

```
calculator.py, line 64, in operation
    def operation(self, op):
        self.current = float(self.current)
        if self.check_sum:
            self.txtDisplay.delete(0, END)
            self.txtDisplay.insert(0, value)
```



3.5. Write the mathPM function.



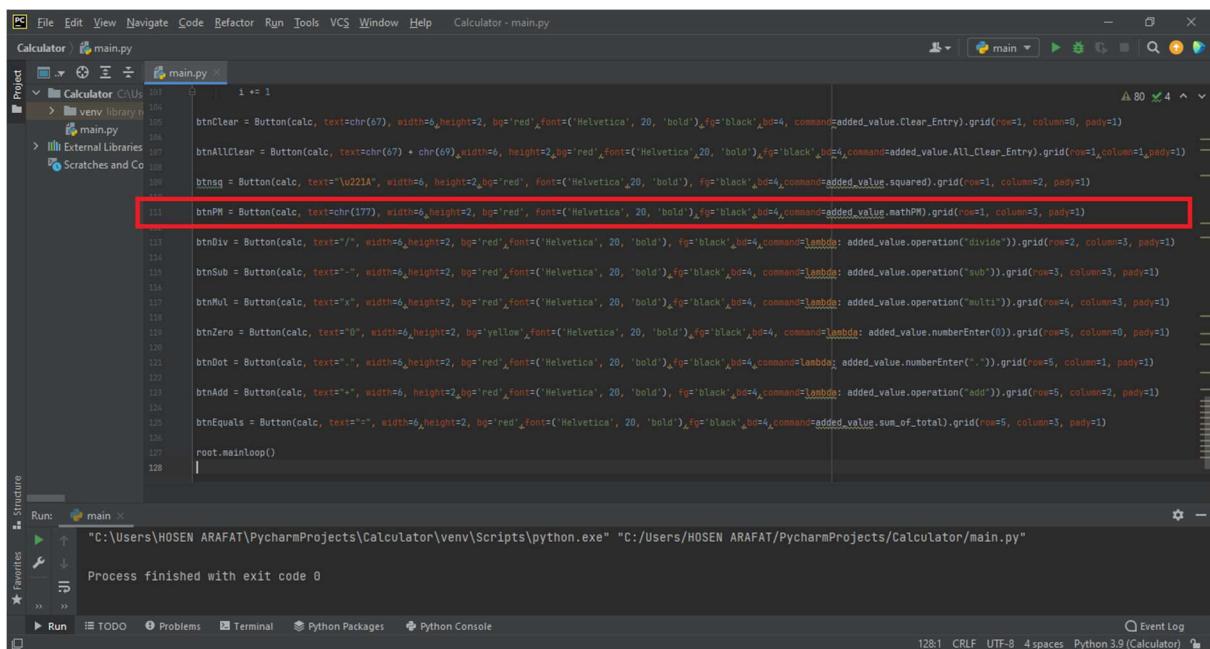
```
Project: Calculator C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator > main.py
Project: Calculator C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
def __init__(self):
    self.current = "0"
    self.input_value = True
    self.check_sum = True
    self.op = op
    self.result = False

    def Clear_Entry(self):
        self.result = False
        self.current = "0"
        self.display(0)
        self.input_value = True

    def All_Clear_Entry(self):
        self.Clear_Entry()
        self.total = 0

    def mathPM(self):
        self.result = False
        self.current = -(float(txtDisplay.get()))
        self.display(self.current)

Calc
Run: main
Process finished with exit code 0
Event Log
```

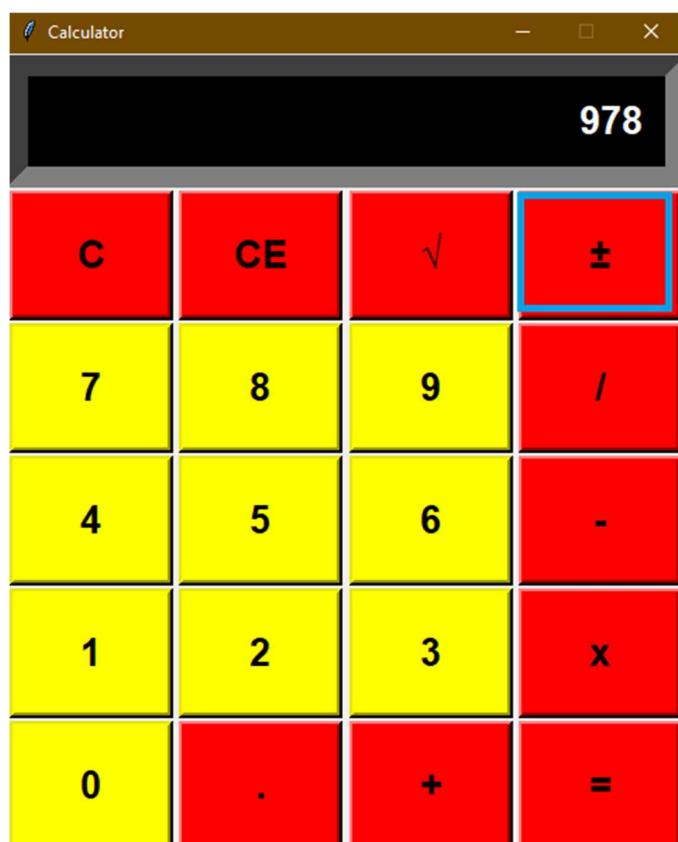


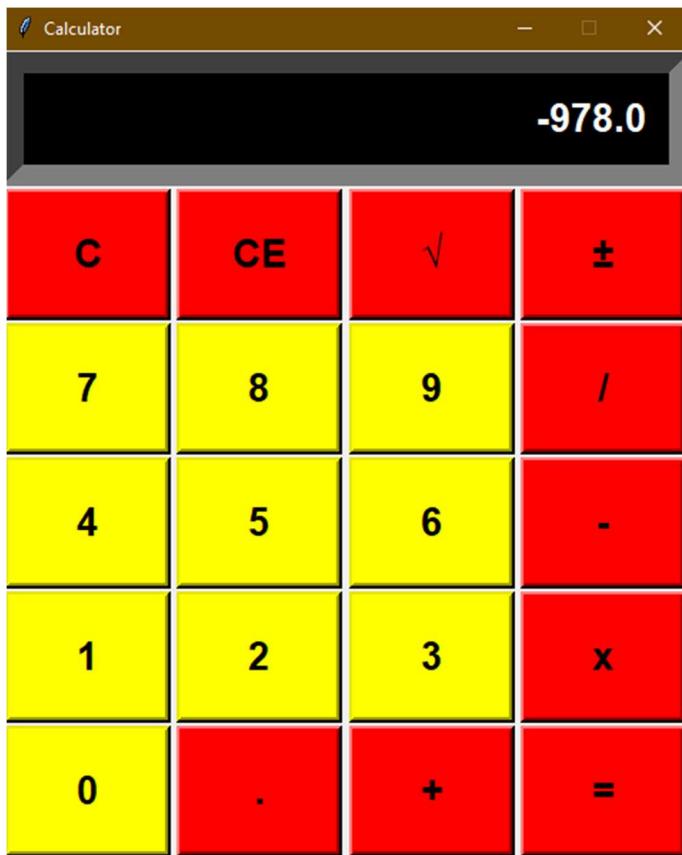
```
Project: Calculator C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator > main.py
Project: Calculator C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
111 btnPM = Button(calc, text=chr(177), width=6, height=2, bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4, command=lambda: added_value.mathPM).grid(row=1, column=3, pady=1)

Calc
Run: main
Process finished with exit code 0
Event Log
```

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** Calculator
- Code Editor:** The file `main.py` is open, displaying Python code for a calculator application. The code defines several buttons for clearing the screen, performing arithmetic operations, and handling user input.
- Calculator Application:** A window titled "Calculator" is running in the background. It features a 4x4 grid of buttons. The buttons are colored as follows:
 - Red buttons: C, CE, √, ±, ., +, =.
 - Yellow buttons: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, /, x.
- Status Bar:** Shows the path "C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\venv\Scripts\python.exe" and the file "C:/Users/HOSEN ARAFAT".
- Bottom Navigation:** Run, TODO, Problems, Terminal, Python Packages, Python Console.
- Event Log:** Shows "128:1 CRLF UTF-8 4 spaces Python 3.9 (Calculator)"





3.6. Write the Squared function.

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help Calculator - main.py
Calculator / main.py
Project
Calculator / venv library
main.py
External Libraries
Scratches and Co
Run: main.py
Structure
Run: main.py
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Event Log
94:1 CRLF UTF-8 4 spaces Python 3.9 (Calculator)

```

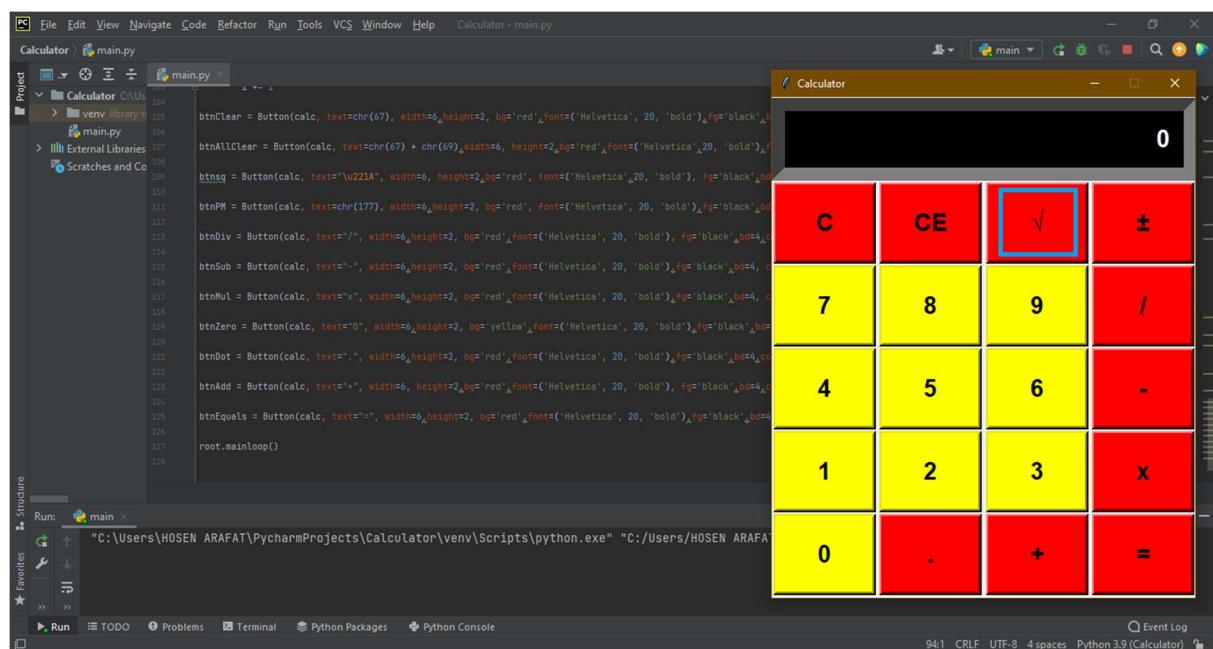
```

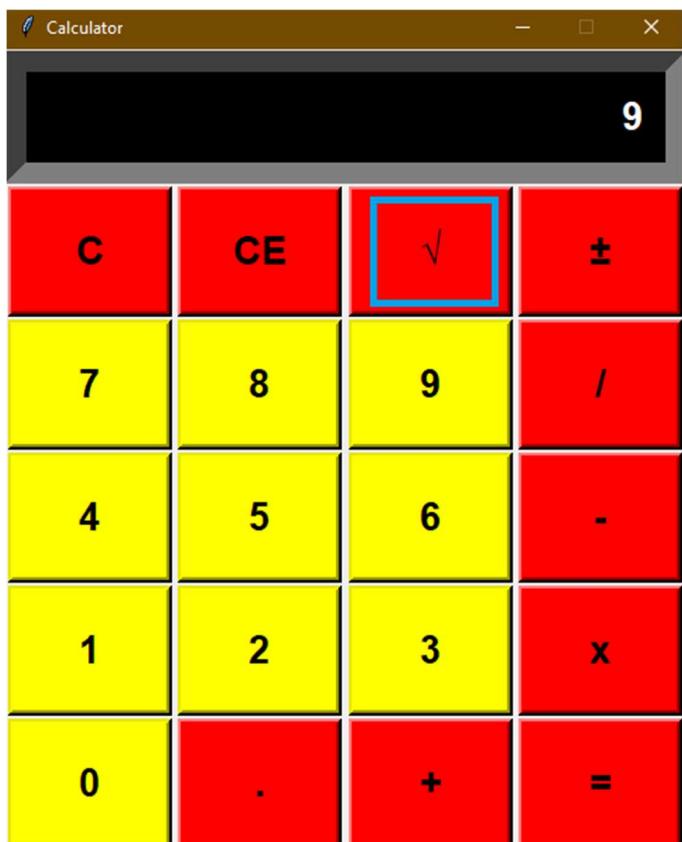
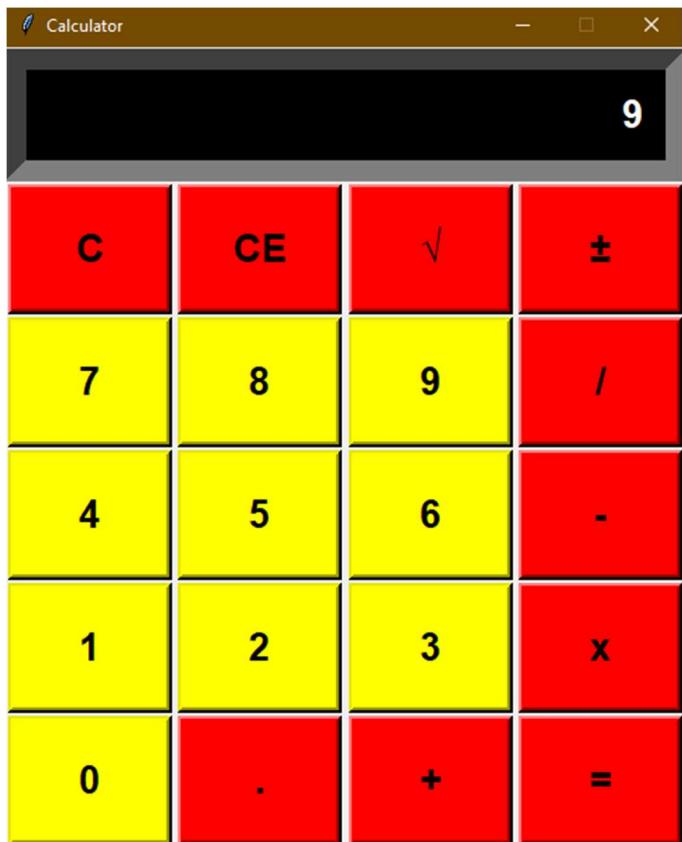
81     def ALL_Clear_Entry(self):
82         self.Clear_Entry()
83         self.total = 0
84
85     def mathPM(self):
86         self.result = False
87         self.current = -(float(txtDisplay.get()))
88         self.display(self.current)
89
90     def squared(self):
91         self.result = False
92         self.current = math.sqrt(float(txtDisplay.get()))
93         self.display(self.current)
94
95     added_value = Calc()
96     numberpad = "789456123"
97     i = 0

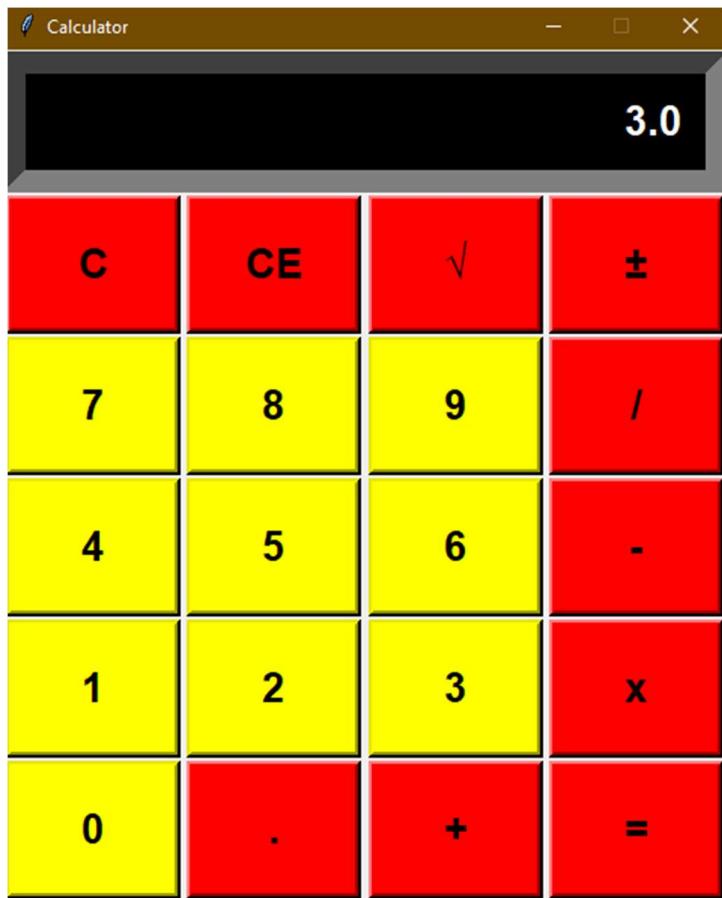
```

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** Calculator
- File:** main.py
- Code Content:** The code defines a Tkinter-based calculator application. It includes buttons for clear, all clear, square, percentage, division, subtraction, multiplication, addition, and decimal point operations. It also includes buttons for zero through nine and an equals button. The root window has a main loop.
- Run Tab:** Shows the command "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/venv/Scripts/python.exe" "C:/Users/HOSEN ARAFAT/PycharmProjects/Calculator/main.py"
- Status Bar:** Shows "Process finished with exit code 0" and "94:1 CRLF UTF-8 4 spaces Python 3.9 (Calculator)"

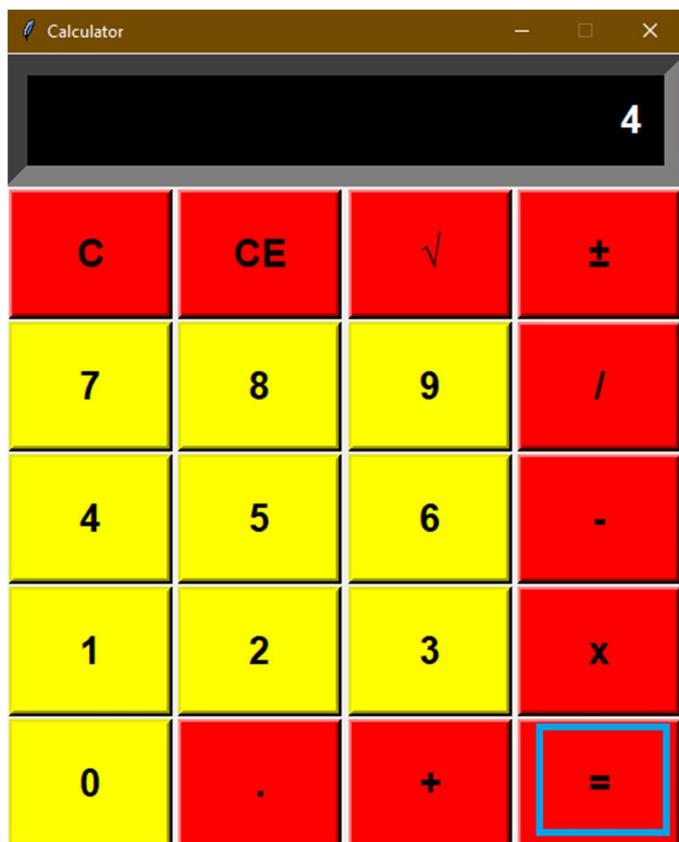
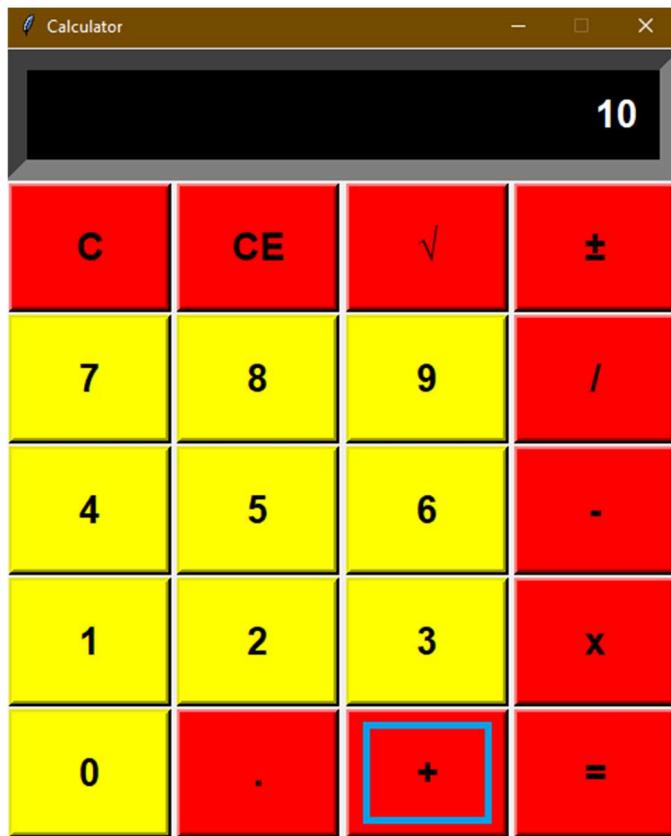


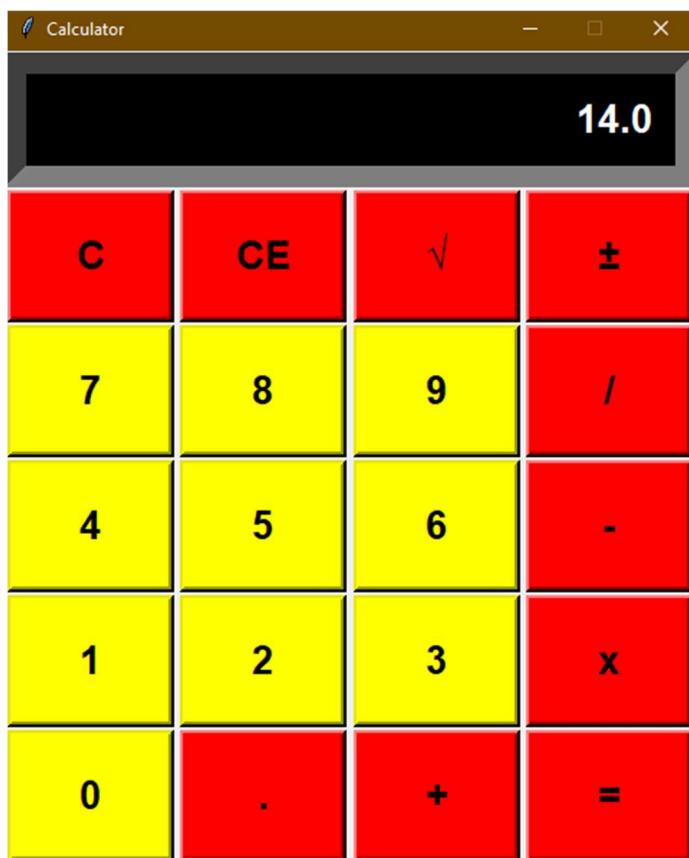




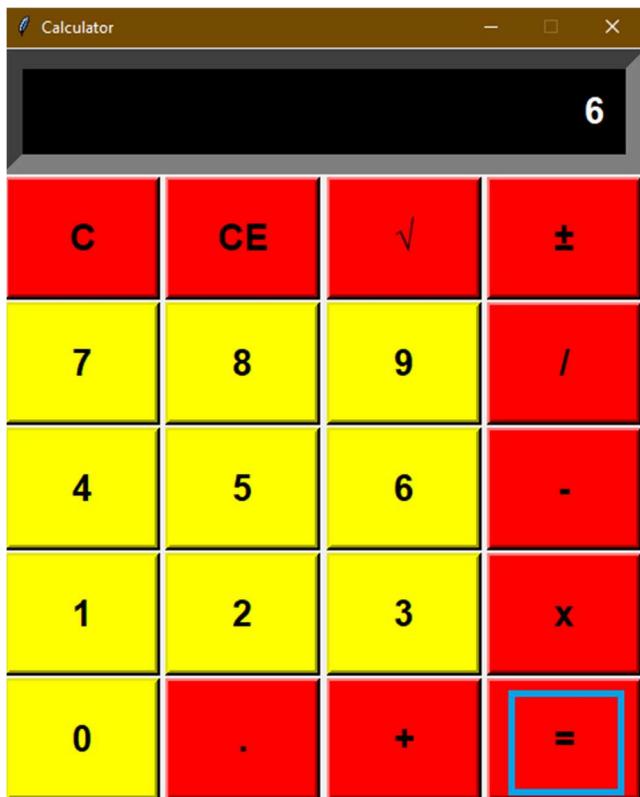
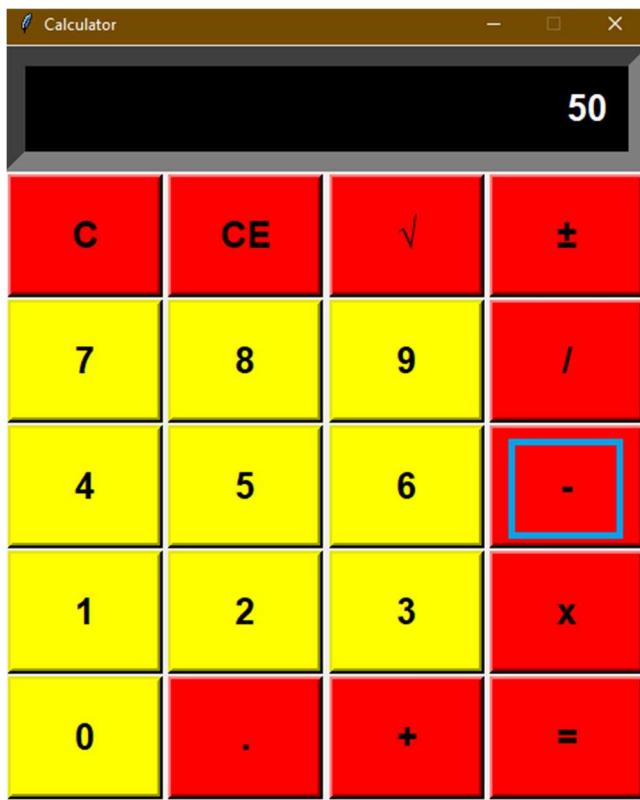
3.7 Run the program for calculator related operations

1) “add” $10+4=14$

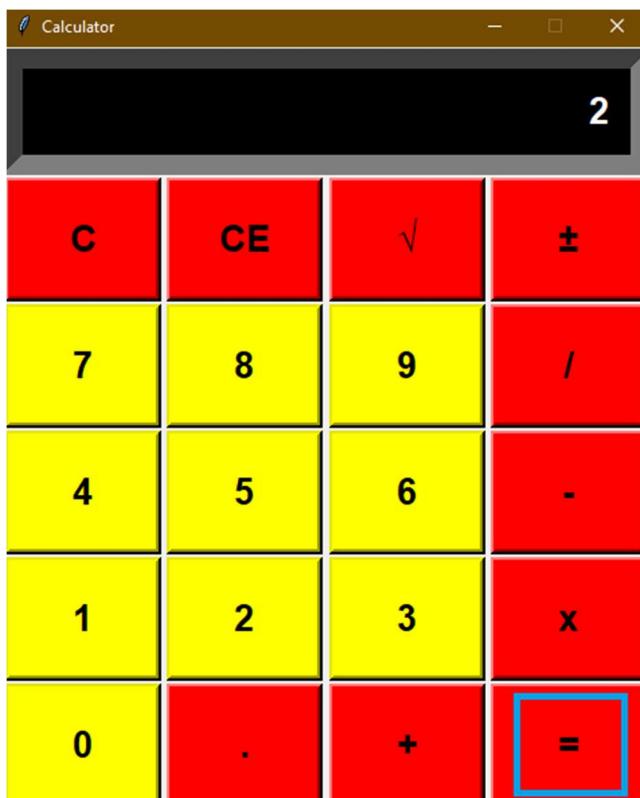
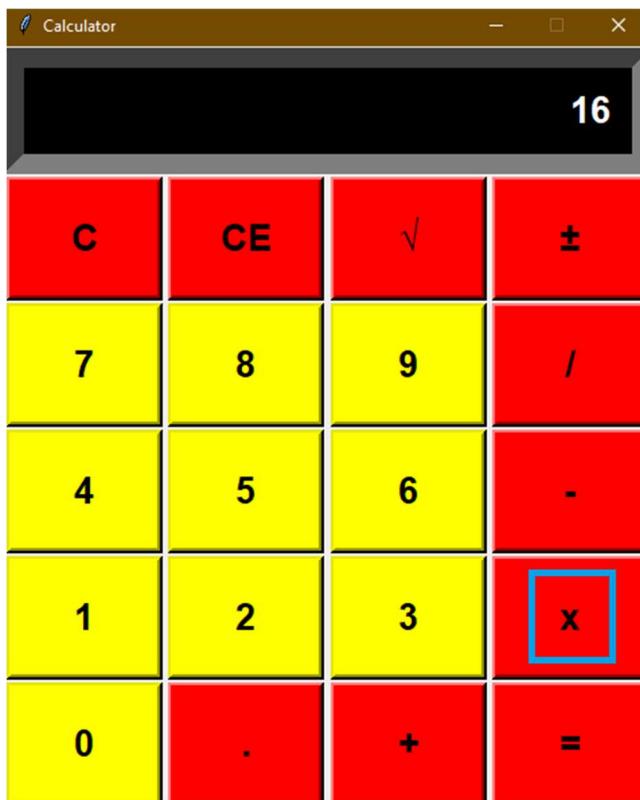


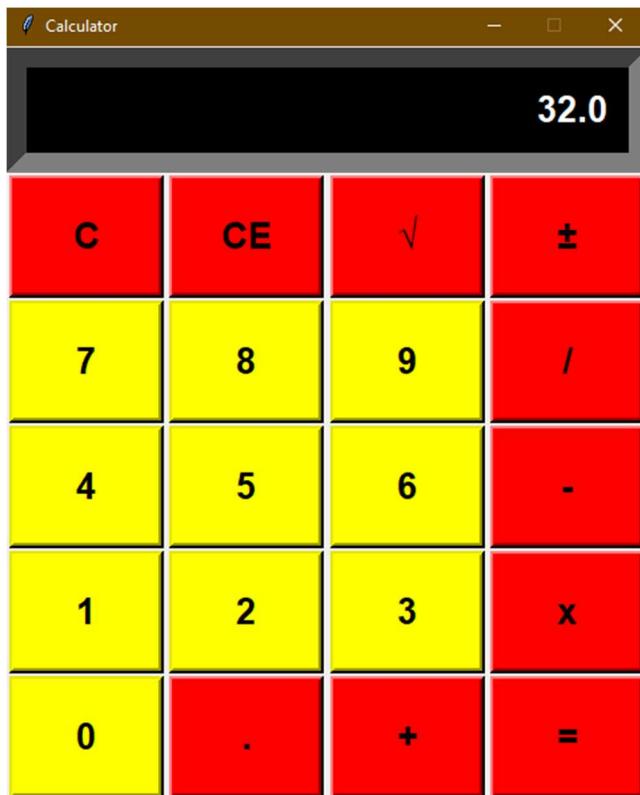


2) "sub" $50 - 6 = 44$

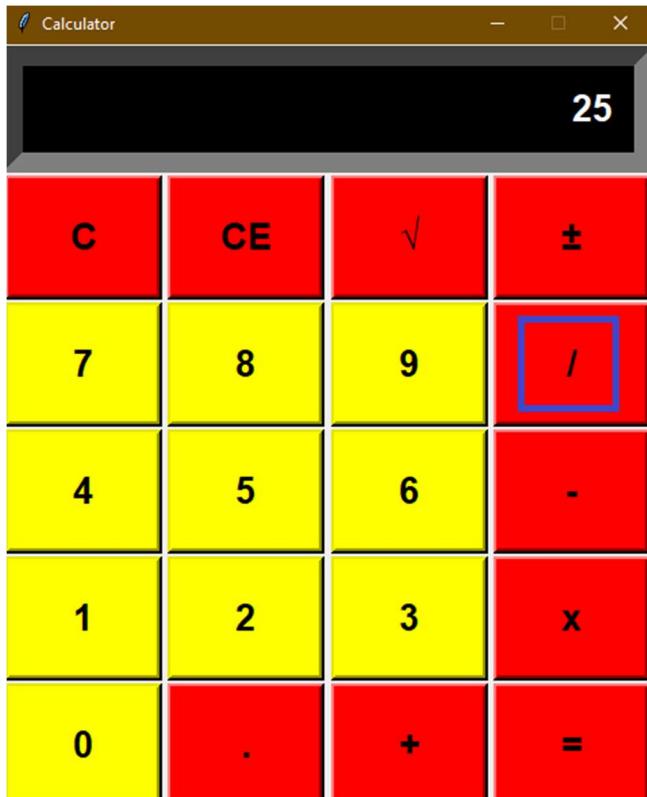


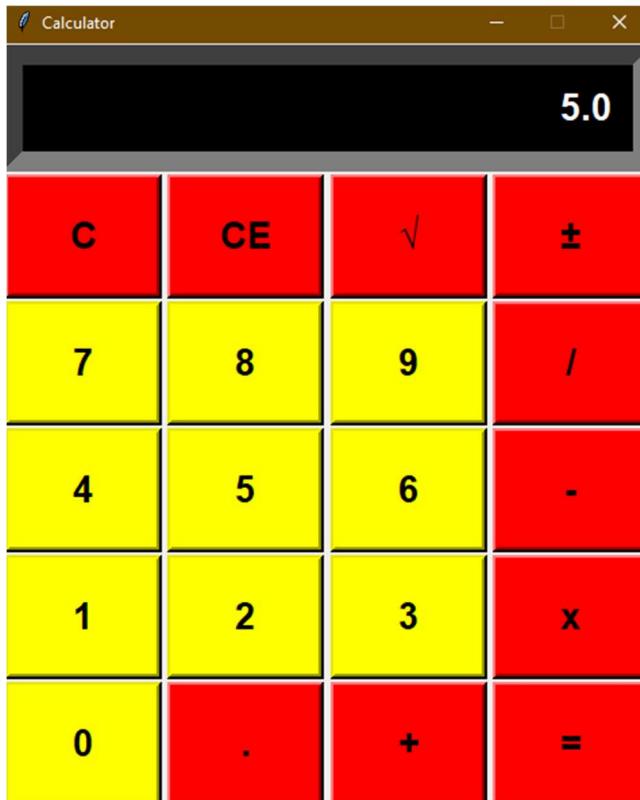
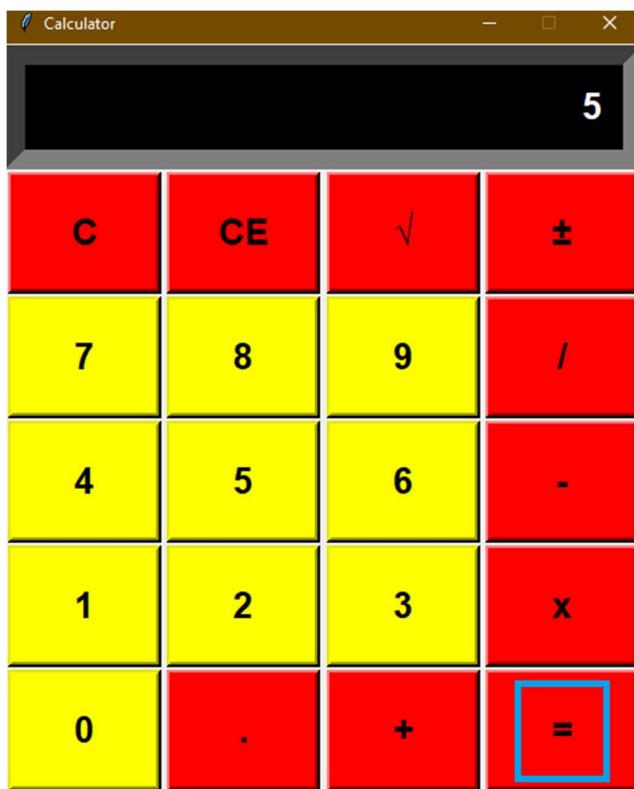
3) "multi" $16 * 2 = 32$



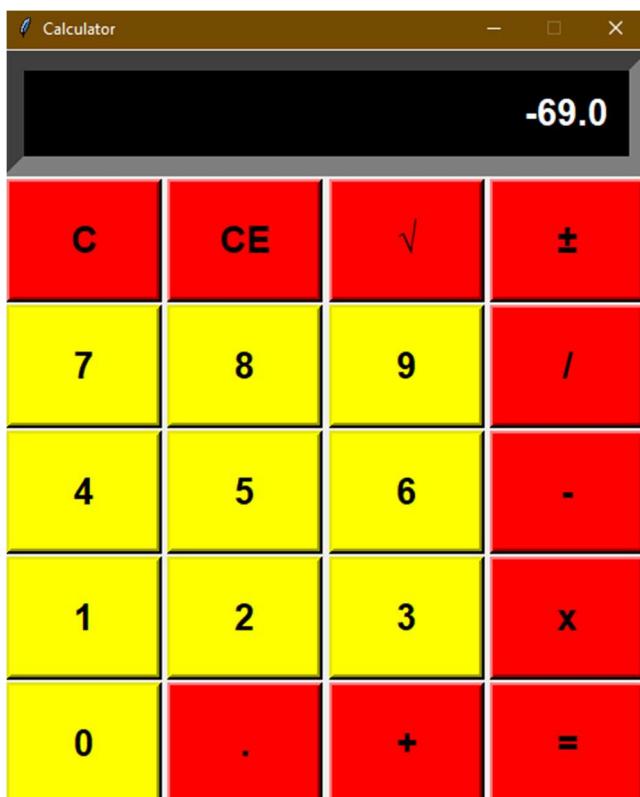
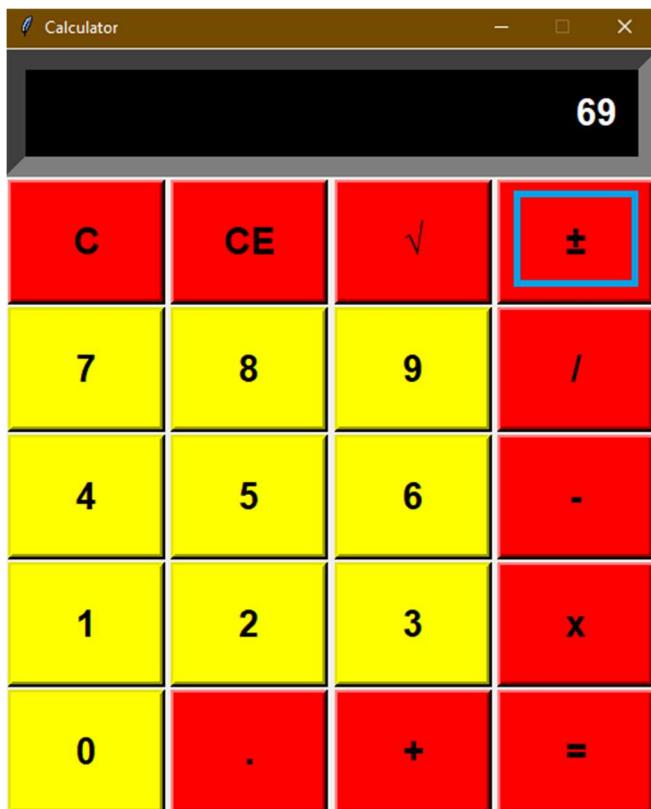


4) “divide” $25 \div 5 = 5$

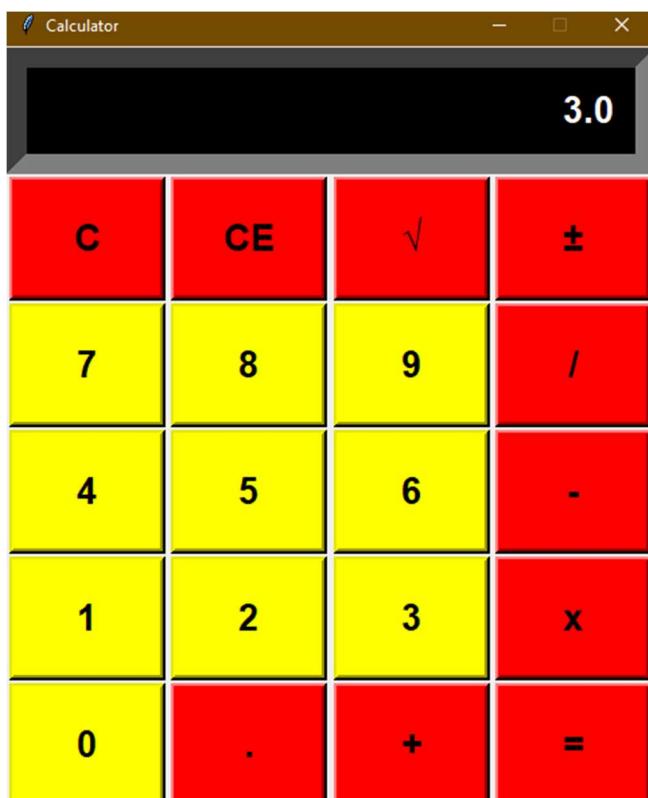
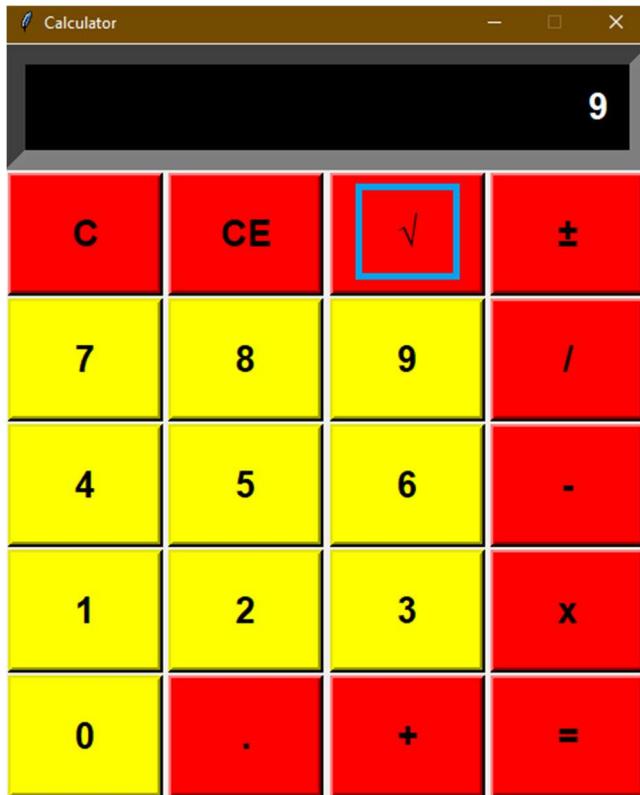




5) "mathPM" $69 \pm$

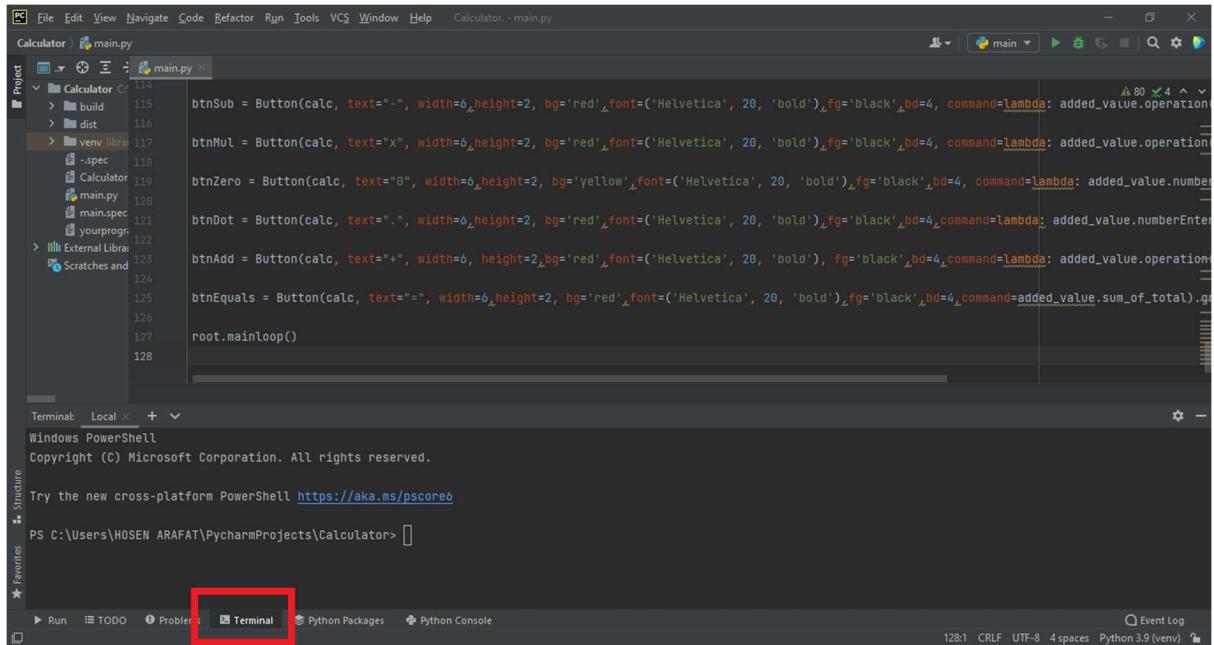


6) “Squared” $9\sqrt{ }$



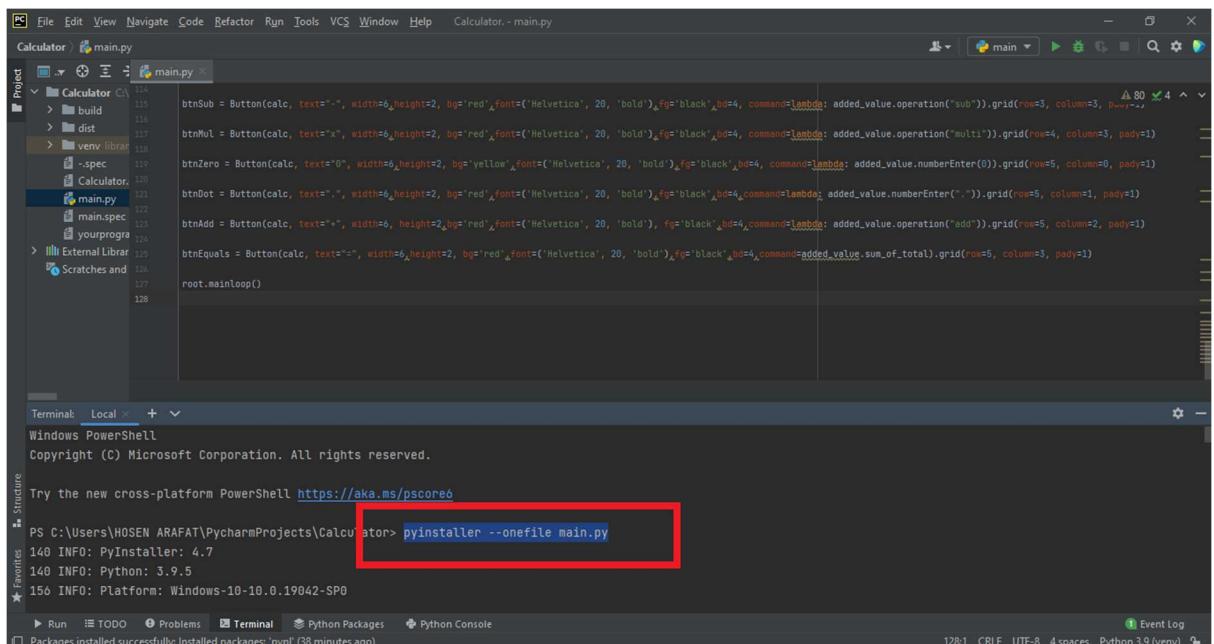
III. Packaged applications

1. Click terminal to open the terminal



A screenshot of the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Calculator. Below the menu is a toolbar with icons for file operations. The main area shows a code editor with Python code for a calculator application, specifically the main.py file. The code defines various buttons for arithmetic operations and numbers. At the bottom of the screen is a terminal window titled "Windows PowerShell". The terminal tab is highlighted with a red box. The terminal output shows the command "PS C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator>" followed by a blank line. The status bar at the bottom right indicates "128:1 CRLF UTF-8 4 spaces Python 3.9 (venv) 9m".

2. Enter the code "pyinstaller --onefile main.py" package "



A screenshot of the PyCharm IDE interface, similar to the previous one but with a command entered in the terminal. The terminal tab is highlighted with a red box. The terminal output shows the command "PS C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator> pyinstaller --onefile main.py" followed by several informational log lines: "140 INFO: PyInstaller: 4.7", "140 INFO: Python: 3.9.5", and "156 INFO: Platform: Windows-10-10.0.19042-SP0". The status bar at the bottom right indicates "128:1 CRLF UTF-8 4 spaces Python 3.9 (venv) 9m".

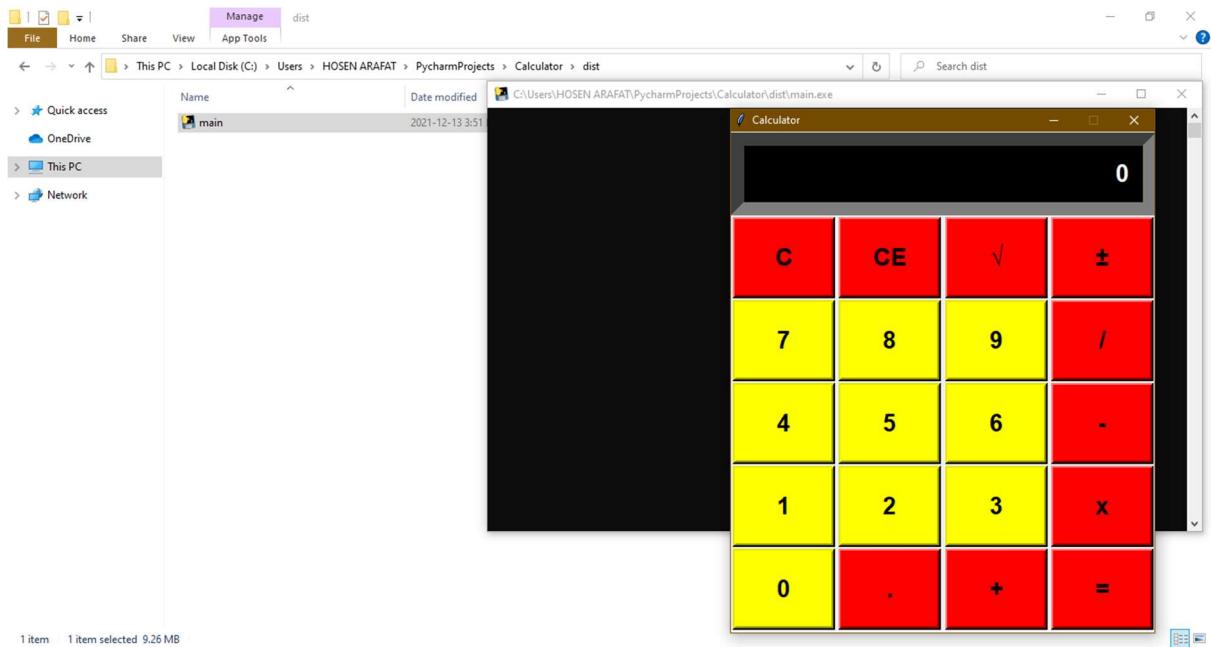
3. Package succeeded

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Calculator. The main window displays the code for 'main.py' under the 'Calculator' project. The code defines various buttons for a calculator, such as btnSub, btnMul, btnZero, btnDot, btnAdd, and btnEquals, with their respective text, width, height, background color ('red' or 'yellow'), font ('Helvetica' or 'bold'), and command (lambda functions). The 'Terminal' tab at the bottom shows the build log:

```
12967 INFO: Writing RT_ICON 3 resource with 1384 bytes
12967 INFO: Writing RT_ICON 4 resource with 37019 bytes
12967 INFO: Writing RT_ICON 5 resource with 9640 bytes
12967 INFO: Writing RT_ICON 6 resource with 4264 bytes
12967 INFO: Writing RT_ICON 7 resource with 1128 bytes
13045 INFO: Copying 0 resources to EXE
13045 INFO: Embedding manifest in EXE
13045 INFO: Updating manifest in C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator\dist\main.exe
13139 INFO: Updating resource type 24 name 1 language 0
13139 INFO: Appending PKG archive to EXE
10811 INFO: Building EXE from EXE-00.toc completed successfully.
PS C:\Users\HOSEN ARAFAT\PycharmProjects\Calculator>
```

The status bar at the bottom indicates 'Packages installed successfully: Installed packages: "pyqt" (39 minutes ago)'. The bottom right corner shows the event log with 128:1 CRLF UTF-8 4 spaces Python 3.9 (venv).

4. Open the main.exe file in the dist folder



IV.Codes

```
from tkinter import *
```

```
import math
```

```
root = Tk()
```

```
root.title("Calculator")
```

```
root.configure(background='white')
```

```
root.resizable(width=False, height=False)
```

```
root.geometry("480x568+450+90")
```

```
calc = Frame(root)
```

```
calc.grid()
```

```
txtDisplay = Entry(calc, font=('Helvetica', 20, 'bold'), bg='black',  
fg='white', bd=30, width=28, justify=RIGHT)
```

```
txtDisplay.grid(row=0, column=0, columnspan=4, pady=1)
```

```
txtDisplay.insert(0, "0")
```

```
class Calc():
```

```
    def __init__(self):
```

```
        self.total = 0
```

```
    self.current = ""  
  
    self.input_value = True  
  
    self.check_sum = False  
  
    self.op = ""  
  
    self.result = False
```

```
def numberEnter(self, num):  
  
    self.result = False  
  
    firstnum = txtDisplay.get()  
  
    secondnum = str(num)  
  
    if self.input_value:  
  
        self.current = secondnum  
  
        self.input_value = False  
  
    else:  
  
        if secondnum == '.':  
  
            if secondnum in firstnum:  
  
                return  
  
        self.current = firstnum + secondnum  
  
    self.display(self.current)
```

```
def sum_of_total(self):  
    self.result = True  
  
    self.current = float(self.current)  
  
    if self.check_sum == True:  
        self.valid_function()  
  
    else:  
        self.total = float(txtDisplay.get())
```

```
def display(self, value):  
    txtDisplay.delete(0, END)  
  
    txtDisplay.insert(0, value)
```

```
def valid_function(self):  
    if self.op == "add":  
        self.total += self.current  
  
    if self.op == "sub":  
        self.total -= self.current  
  
    if self.op == "multi":  
        self.total *= self.current  
  
    if self.op == "divide":
```

```
    self.total /= self.current
```

```
    self.input_value = True
```

```
    self.check_sum = False
```

```
    self.display(self.total)
```

```
def operation(self, op):
```

```
    self.current = float(self.current)
```

```
    if self.check_sum:
```

```
        self.valid_function()
```

```
    elif not self.result:
```

```
        self.total = self.current
```

```
        self.input_value = True
```

```
        self.check_sum = True
```

```
        self.op = op
```

```
        self.result = False
```

```
def Clear_Entry(self):
```

```
    self.result = False
```

```
    self.current = "0"
```

```
    self.display(0)
```

```
    self.input_value = True
```

```
def All_Clear_Entry(self):
```

```
    self.Clear_Entry()
```

```
    self.total = 0
```

```
def mathPM(self):
```

```
    self.result = False
```

```
    self.current = -(float(txtDisplay.get()))
```

```
    self.display(self.current)
```

```
def squared(self):
```

```
    self.result = False
```

```
    self.current = math.sqrt(float(txtDisplay.get()))
```

```
    self.display(self.current)
```

```
added_value = Calc()
```

```
numberpad = "789456123"
```

```
i = 0
```

```
btn = []
```

```

for j in range(2, 5):
    for k in range(3):
        btn.append(Button(calc, width=6, height=2, bg='yellow',
fg='black', font=('Helvetica', 20,
'bold'), bd=4, text=numberpad[i])), btn[i].grid(row=j, column=k,
pady=1)

        btn[i]["command"] = lambda x=numberpad[i]:
added_value.numberEnter(x)

        i += 1

```

```

btnClear = Button(calc, text=chr(67), width=6, height=2,
bg='red', font=('Helvetica', 20, 'bold'), fg='black', bd=4,
command=added_value.Clear_Entry).grid(row=1, column=0,
pady=1)

```

```

btnAllClear = Button(calc, text=chr(67) + chr(69), width=6,
height=2, bg='red', font=('Helvetica', 20,
'bold'), fg='black', bd=4, command=added_value.All_Clear_Entry).g
rid(row=1, column=1, pady=1)

```

```

btnsq = Button(calc, text="\u221A", width=6,
height=2, bg='red', font=('Helvetica', 20,
'bold'), fg='black', bd=4, command=added_value.squared).grid(row=1,

```

```
column=2, pady=1)
```

```
btnPM = Button(calc, text=chr(177), width=6,height=2,  
bg='red', font=('Helvetica', 20,  
'bold'),fg='black',bd=4,command=added_value.mathPM).grid(ro  
w=1, column=3, pady=1)
```

```
btnDiv = Button(calc, text="/", width=6,height=2,  
bg='red',font=('Helvetica', 20, 'bold'),  
fg='black',bd=4,command=lambda:  
added_value.operation("divide")).grid(row=2, column=3, pady=1)
```

```
btnSub = Button(calc, text="-", width=6,height=2,  
bg='red',font=('Helvetica', 20, 'bold'),fg='black',bd=4,  
command=lambda: added_value.operation("sub")).grid(row=3,  
column=3, pady=1)
```

```
btnMul = Button(calc, text="x", width=6,height=2,  
bg='red',font=('Helvetica', 20, 'bold'),fg='black',bd=4,  
command=lambda: added_value.operation("multi")).grid(row=4,  
column=3, pady=1)
```

```
btnZero = Button(calc, text="0", width=6,height=2,
```

```
bg='yellow',font=('Helvetica', 20, 'bold'),fg='black',bd=4,  
command=lambda: added_value.numberEnter(0)).grid(row=5,  
column=0, pady=1)
```

```
btnDot = Button(calc, text=". ", width=6,height=2,  
bg='red',font=('Helvetica', 20,  
'bold'),fg='black',bd=4,command=lambda:  
added_value.numberEnter(".")).grid(row=5, column=1, pady=1)
```

```
btnAdd = Button(calc, text="+", width=6,  
height=2,bg='red',font=('Helvetica', 20, 'bold'),  
fg='black',bd=4,command=lambda:  
added_value.operation("add")).grid(row=5, column=2, pady=1)
```

```
btnEquals = Button(calc, text="=", width=6,height=2,  
bg='red',font=('Helvetica', 20,  
'bold'),fg='black',bd=4,command=added_value.sum_of_total).grid  
(row=5, column=3, pady=1)
```

```
root.mainloop()
```

