CALCULATOR

END TERM REPORT

by

V SHIVANSHU YADAV, DIKSHA SRIVASTAVA, DIVYA GUPTA

Section: K19PV

Roll Numbers: A16, A19, A36

Department of Intelligent Systems,
School of Computer Science Engineering,
Lovely Professional University, Jalandhar
November, 2020

STUDENT DECLARATION

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

NAME: V SHIVANSHU YADAV
ROLL NUMBER: RK19PVA16

NAME: DIKSHA SRIVASTAVA
ROLL NUMBER: RK19PVA19

NAME: DIVYA GUPTA
ROLL NUMBER: RK19PVA36

PLACE: JALANDHAR, PUNJAB
DATE: 30th OCTOBER, 2020

TABLE OF CONTENTS

TITLE	PAGE NO.
1. BONAFIDE CERTIFICATE	04
2. BACKGROUND AND DESCRIPTION	
2.1 BACKGROUND	05
2.2 DESCRIPTION	06
3. OBJECTIVE AND CONCRETE GOALS	
3.1 OBJECTIVE	07
3.2 CONCRETE GOALS	07
4. MOTIVATION AND OUTCOME	07
5. DESCRIPTION OF PROJECT	08
6. WORK DIVISION	09
7. IMPLEMENTATION OF PROJECT	10
8. TECHNOLOGIES AND FRAMEWORK USED	21
9. SWOT ANALYSIS	23

BONAFIDE CERTIFICATE

Certified that this project report “CALCULATOR” is the bonafide work of “V SHIVANSHU YADAV, DIKSHA SRIVASTAVA and DIVYA GUPTA” who carried out the project work under my supervision.

NAME OF SUPERVISOR:

DR. DHANPRATAP SINGH

ACADEMIC DESIGNATION:

ASSISTANT PROFESSOR

ID OF SUPERVISOR:

25706

DEPARTMENT OF SUPERVISOR:

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

BACKGROUND AND DESCRIPTION

BACKGROUND:

The first known tools used to support arithmetic calculations were: bones (used to tally items), pebbles, and counting boards, and the abacus. In 1642, the Renaissance saw the invention of the mechanical calculator (by Wilhelm Schickard and several decades later Blaise Pascal), a device that was at times somewhat over-promoted as being able to perform all four arithmetic operations(Addition, Subtraction, Multiplication and Division) with minimal human interference. It performed calculations through a clockwork-type of mechanism. The Pascal calculator, invented by French inventor and mathematician Blaise Pascal, was lauded for attempting arithmetic calculations previously thought impossible. The first mainframe computers, using firstly vacuum tubes and later transistors in the logic circuits, appeared in the 1940s and 1950s. This technology was to provide a milestone to the development of electronic calculators.

The calculator is one application that we all use in our day to day lives.

With the invention of the first smartphone in 1995, individuals began to replace expensive digital calculators with the multiuse device. This required even the most sophisticated calculator designs to be upgraded in order to remain relevant in the market. A calculator is a device that performs arithmetic operations on numbers. A simple calculator is a machine that allows people to do mathematical operations more easily. For example, most calculators will add, subtract, multiply, and divide. Some also do square roots, and more complex calculators can help with calculus and draw function graphs. Calculators are found everywhere. A smartphone or other computer can also act as a calculator. Calculators have not only greatly enhanced our ability to perform the regular computations that are involved in everyday life but provided humans with the ability to understand mathematics on a greater scale than ever imagined. Calculations that were previously cumbersome and time-consuming can now be done in minutes or even seconds, all at the push of a few buttons.

DESCRIPTION:
This is a simple calculator developed in python programming IDLE used to perform basic mathematical calculations like
Addition (+)
Subtraction (-)
Multiplication (*)
Division (/)

- This program is developed via python version 3.58.

- What should The Calculator do/have?

We have listed all of the features a calculator should have, and we used them as a guide while building. The calculator should have

  - A screen for displaying inputs and output
  - Buttons with numbers on them to get input from users
  - Should be able to perform four basic operations i.e. Addition, Subtraction, Multiplication, and Division
  - A Backspace button for deleting the number on the screen.
  - A Clear button to clear the output or result from the screen
  - Should support decimal numbers
  - Should have '=' button to compute the result
  - Should have brackets '(' and ')' so that numbers can be grouped together

We built the Simple Calculator using Tkinter as our GUI platform for python programming.

- What is Tkinter?

Tkinter

Tkinter is a Python binding to the Tk GUI toolkit. Tk is the original GUI library for the Tcl language. Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. There are several other popular Python GUI toolkits. Most popular are wxPython, PyQt, and PyGTK

OBJECTIVE AND CONCRETE GOALS

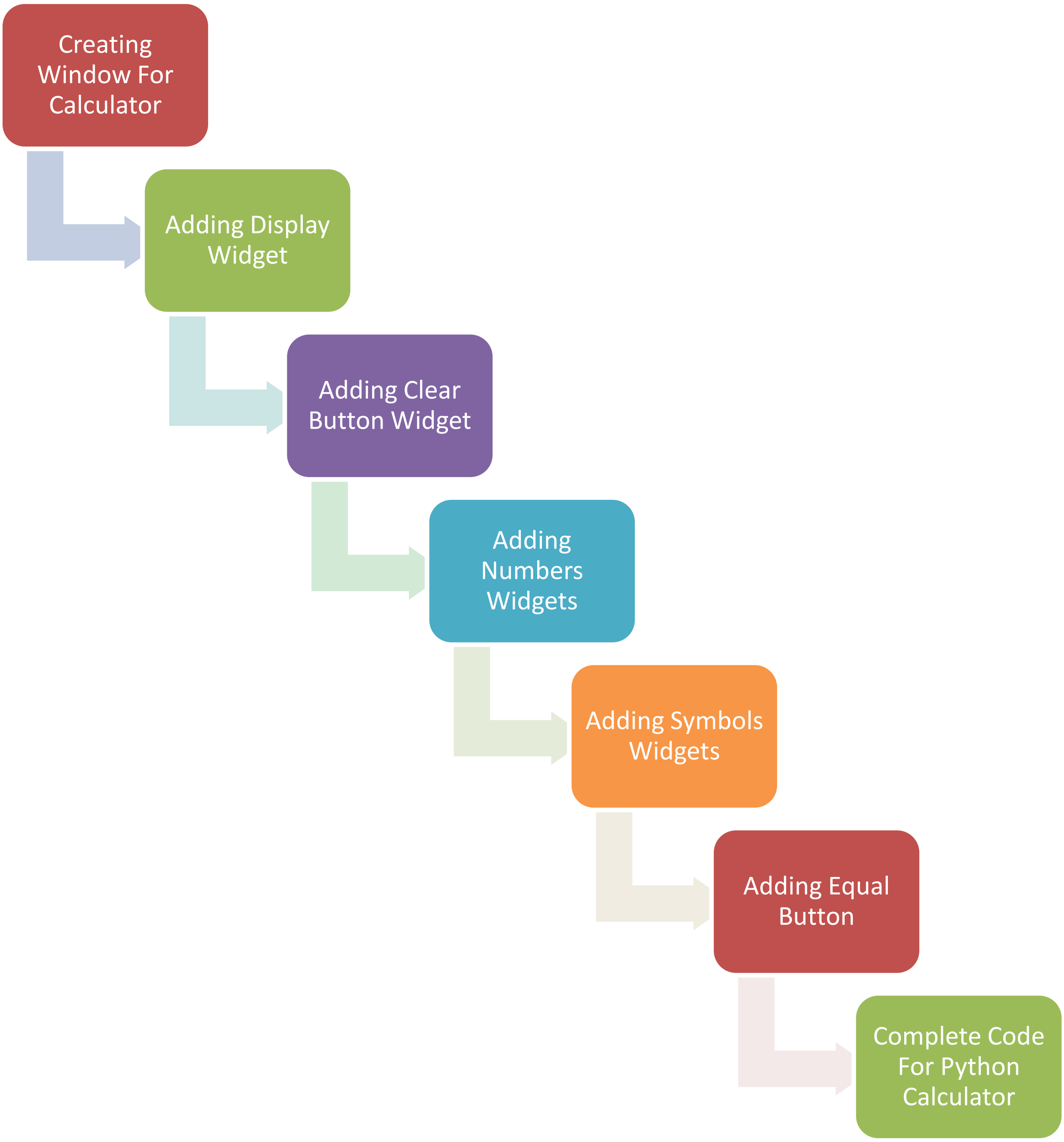
- OBJECTIVE OF THIS PROJECT IS:
  Build a graphical user interface calculator using a library in which we build buttons to perform different mathematical operations like addition(+), subtraction(-), multiplication(*), and division(/) and display results on the screen. We can further add functionalities for scientific calculations. Furthermore, we have built an error-free calculator.

- CONCRETE GOALS OF THIS PROJECT ARE:
  - To learn how to create a simple calculator that can add, subtract, multiply or divide depending upon the input from the user.
  - To have a better understanding of the way python programming language is used to build Graphical User Interface (GUI) applications.

MOTIVATION AND OUTCOME

- Through this project, we learned to create a simple calculator that can add, subtract, multiply, or divide depending upon the input from the user.
- We gain a better understanding of the way python is used to build GUI applications.

DESCRIPTION OF PROJECT USING PICTORIAL FLOW CHART REPRESENTATION


Creating Window For Calculator
Adding Display Widget
Adding Clear Button Widget
Adding Numbers Widgets
Adding Symbols Widgets
Adding Equal Button
Complete Code For Python Calculator

WORK DIVISION

V. Shivanshu Yadav
Registration No – 11902839
Roll No-A16
He did all the implementation regarding the structure of the Graphical user interface; also, he performs all the linking of application functions to the main window (Graphical interface) for the desired output. And he discussed all the functioning, for better understanding and implementation of this project. He also fixed bugs and errors.

Diksha Srivastava
Registration No – 11902586
Roll No-19
She implements all the functional aspects of the project from the evaluation of data to error handling and bug fixing. She got indulged in connecting functioning and graphical interface all together for precise output. Also, she helped us out in explaining each bit of the program and its variable’s data throughout and their respective declarations.

Divya Gupta
Registration No – 11907192
Roll No-A36
She created buttons for the input-output among users and calculator. She performed tasks related to documentation and giving us appropriate knowledge of all the modules available to create a meaningful layout. She made flowcharts and guided us to works accordingly in a managed pathway. Also, she performed the fixing of bugs and error.

IMPLEMENTATION OF PROJECT

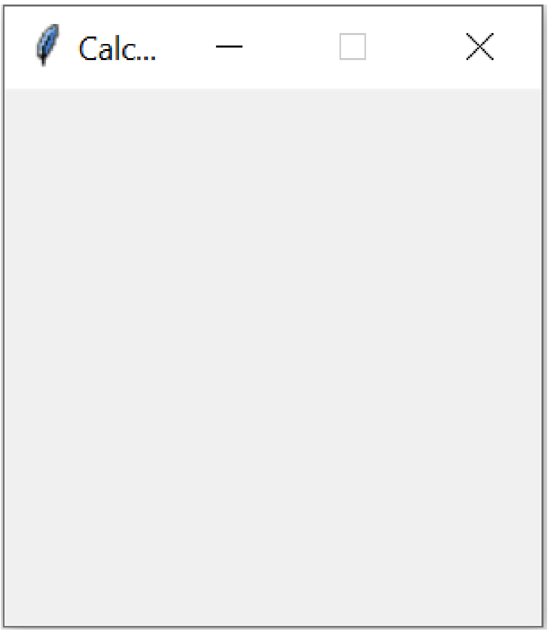
STAGE ONE:

We have created a box where all the buttons will be added to build a Simple Calculator.

```
from tkinter import *
Error_msg="ERROR"
def btnClick(num):
    global op
    op=op+str(num)
    n.set(op)
def allClear():
    global op
    op=""
    n.set("")
def equals():
    global op
    try:
        rslt=str(eval(op))
    except Exception:
        rslt = Error_msg
    n.set(rslt)
    op=""
def backspace():
    global op
    op=op[:-1]
    n.set(op)
root = Tk()
```

```
root.title("Calculator")

root.resizable(0,0)

root.mainloop()
```

What we’ve done here is set up our empty board or canvas if you will, where our calculator buttons and screen will live. If you run the program now, you get this

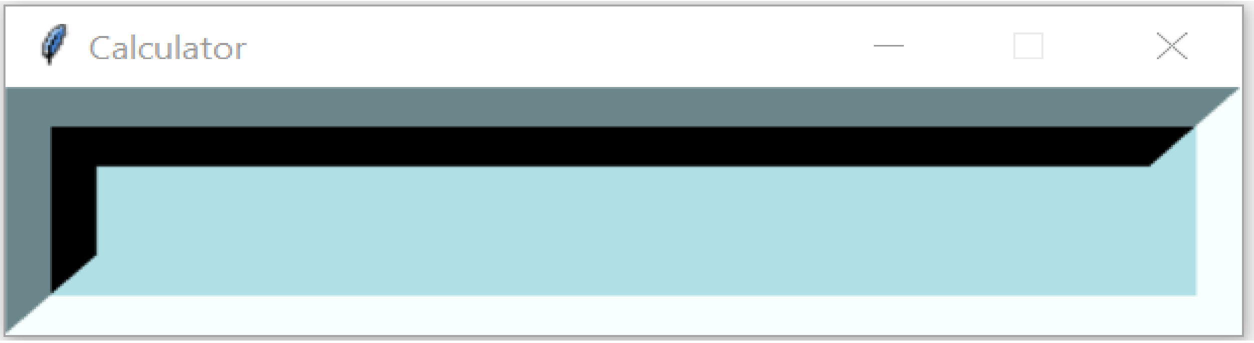


STAGE TWO

We are going to create our screen in this step.

```
from tkinter import *

Error_msg="ERROR"

def btnClick(num):

    global op

    op=op+str(num)

    n.set(op)

def allClear():
    global op
    op=""
    n.set("")
def equals():
    global op
    try:
        rslt=str(eval(op))
    except Exception:
        rslt = Error_msg
    n.set(rslt)
    op=""
def backspace():
    global op
    op=op[:-1]
    n.set(op)
root = Tk()
root.title("Calculator")
root.resizable(0,0)
op = ""
n = StringVar()
text = Entry(root,textvariable=n,width=23,bd=30,font=("Arial",20,"bold"),bg="powder blue",insertwidth=4,justify="right").grid(columnspan=4)
root.mainloop()
```

In the above code, we use the text class from the Tkinter module as our screen. It takes height and width and background. These properties determine the appearance of our screen. You can experiment with different dimensions and background colors to get familiar with it. In our code, we use the Tkinter grid system of placing widgets on a surface. It is pretty straight-forward. On line 30, we are simply asking Tkinter to place our POWER BLUE screen on the zeroth row and zeroth column and that the screen should take the space of 4 columns, hence columnspan=4.On line 28, n, is the variable that is going to represent what is typed on the screen. For now, it is empty, but don’t forget about it because we are coming back to it. At this point, we have this



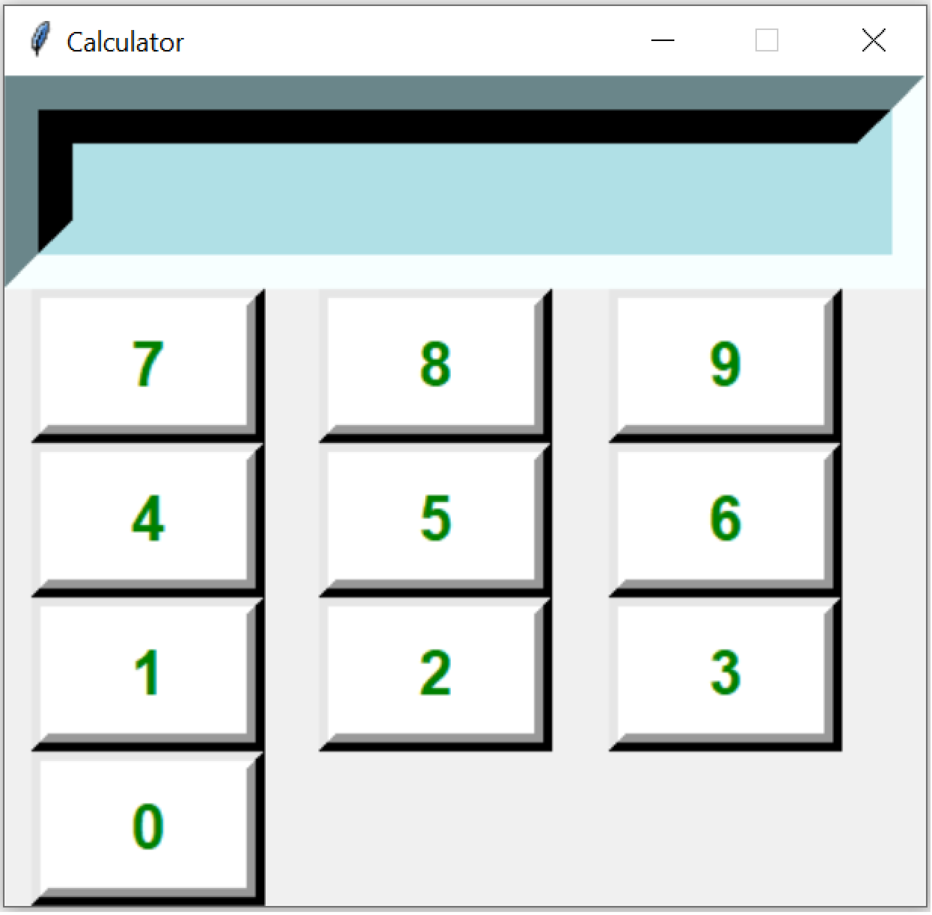
STAGE THREE

Our next goal, is to add buttons to the calculator board, so that we can, you know, calculate numbers.

```
from tkinter import *

Error_msg="ERROR"

def btnClick(num):

   global op

   op=op+str(num)

   n.set(op)

def allClear():

   global op

   op=""

   n.set("")

def equals():

   global op

   try:
        rslt=str(eval(op))

    except Exception:

        rslt = Error_msg

    n.set(rslt)

    op=""

def backspace():

    global op

    op=op[:-1]

    n.set(op)

root = Tk()

root.title("Calculator")

root.resizable(0,0)

op = ""

n = StringVar()

text = Entry(root,textvariable=n,width=23,bd=30,font=("Arial",20,"bold"),bg="powder
blue",insertwidth=4,justify="right").grid(columnspan=4)

btn7 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="7",bg="white",command=lambda :
btnClick(7)).grid(row=1,column=0)

btn8 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="8",bg="white",command=lambda :
btnClick(8)).grid(row=1,column=1)

btn9 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="9",bg="white",command=lambda :
btnClick(9)).grid(row=1,column=2)

btn4 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="4",bg="white",command=lambda :
btnClick(4)).grid(row=2,column=0)

btn5 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="5",bg="white",command=lambda :
btnClick(5)).grid(row=2,column=1)
```

```
btn6 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="6",bg="white",command=lambda :
btnClick(6)).grid(row=2,column=2)

btn1 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="1",bg="white",command=lambda :
btnClick(1)).grid(row=3,column=0)

btn2 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="2",bg="white",command=lambda :
btnClick(2)).grid(row=3,column=1)

btn3 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="3",bg="white",command=lambda :
btnClick(3)).grid(row=3,column=2)

btn0 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="0",bg="white",command=lambda :
btnClick(0)).grid(row=4,column=0)

root.mainloop()
```

We have created all our buttons using the method createButton. createButton makes use of the Tkinter Button widget. The Button widget takes the board in which it will be displayed. The command attribute is the most important part of the button widget because without it the button is useless. We now need to place our buttons on our board using the grid system which is all about placing items on available rows and columns on a board. Our calculator is now with buttons, albeit useless ones as we have not defined the operations to be used.



STAGE FOUR

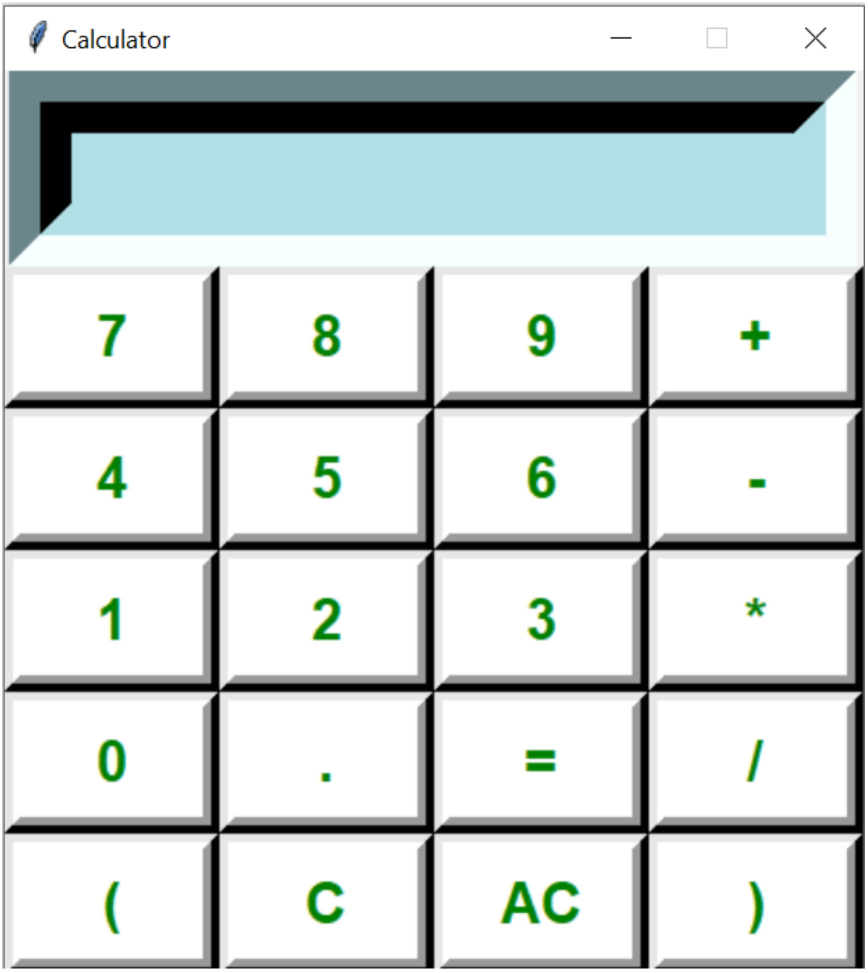
We are about to make our buttons less useless. As we are inserted the operators so that our CALCULATOR ca perform all simple MATHEMATICAL OPERATIONS like: ADDITION, SUBTRACTION and so onn..

```
from tkinter import *

Error_msg="ERROR"

def btnClick(num):

    global op

    op=op+str(num)

    n.set(op)

def allClear():
    global op
    op=""
    n.set("")
def equals():
    global op
    try:
        rslt=str(eval(op))
    except Exception:
        rslt = Error_msg
    n.set(rslt)
    op=""
def backspace():
    global op
    op=op[:-1]
    n.set(op)
root = Tk()
root.title("Calculator")
root.resizable(0,0)
op = ""
n = StringVar()


text = Entry(root,textvariable=n,width=23,bd=30,font=("Arial",20,"bold"),bg="powder blue",insertwidth=4,justify="right").grid(columnspan=4)

btn7 = Button(root,padx=16,width=3,bd=8,fg="Green", font=("Arial",20,"bold"),text="7",bg="white",command=lambda : btnClick(7)).grid(row=1,column=0)

btn8 = Button(root,padx=16,width=3,bd=8,fg="Green", font=("Arial",20,"bold"),text="8",bg="white",command=lambda : btnClick(8)).grid(row=1,column=1)
```

```
btn9 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="9",bg="white",command=lambda :
btnClick(9)).grid(row=1,column=2)

addition = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="+",bg="white",command=lambda :
btnClick("+")).grid(row=1,column=3)

btn4 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="4",bg="white",command=lambda :
btnClick(4)).grid(row=2,column=0)

btn5 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="5",bg="white",command=lambda :
btnClick(5)).grid(row=2,column=1)

btn6 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="6",bg="white",command=lambda :
btnClick(6)).grid(row=2,column=2)

sub = Button(root,padx=16,width=3,bd=8,fg="Green", font=("Arial",20,"bold"),text="-
",bg="white",command=lambda : btnClick("-")).grid(row=2,column=3)

btn1 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="1",bg="white",command=lambda :
btnClick(1)).grid(row=3,column=0)

btn2 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="2",bg="white",command=lambda :
btnClick(2)).grid(row=3,column=1)

btn3 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="3",bg="white",command=lambda :
btnClick(3)).grid(row=3,column=2)

mul = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="*",bg="white",command=lambda :
btnClick("*")).grid(row=3,column=3)

btn0 = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="0",bg="white",command=lambda :
btnClick(0)).grid(row=4,column=0)

point = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text=".",bg="white",command=lambda :
btnClick(".")).grid(row=4,column=1)
```

```
equal = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="=",bg="white",command=equals).grid(row=4,column=2)divis
ion = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="/",bg="white",command=lambda :
btnClick("/")).grid(row=4,column=3)


o_brkt = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="(",bg="white",command=lambda :
btnClick("(")).grid(row=5,column=0)

btn_bckspc = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="C",bg="white",command=backspace).grid(row=5,column=1)

btn_ac = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text="AC",bg="white",command=allClear).grid(row=5,column=2)

c_brkt = Button(root,padx=16,width=3,bd=8,fg="Green",
font=("Arial",20,"bold"),text=")",bg="white",command=lambda :
btnClick(")")).grid(row=5,column=3)

root.mainloop()
```

Now all the operators are inserted. Here we have inserted other special buttons like 'C' to clear recently inserted operand or operator, 'AC' to Clear, '=' to Evaluate and brackets. For Invalid Input the output is will be ERROR as written in 2 lines. Now our Code is ready to Run


Calculator
7
8
9
+
4
5
6
-
1
2
3
*
0
.
=
/
(
C
AC
)

TECHNOLOGY AND FRAMEWORK

Python

Python is a general-purpose, dynamic, object-oriented programming language. The design purpose of the Python language emphasizes programmer productivity and code readability. Python was initially developed by *Guido van Rossum*. It was first released in 1991. Python was inspired by ABC, Haskell, Java, Lisp, Icon, and Perl programming languages. Python is a high-level, general-purpose, multiplatform, interpreted language. Python is well suited for learning about GUI programming.

It is a python project to do simple arithmetic operations like addition, subtraction, multiplication, division, and exponentials. This project is functional only on Python version more excellent or equivalent to 3.0.

We used standard libraries/functions to do so, named the following:

- Tcl/Tkinter for Graphical user interface.
- Default functions like eval.
- Typecasting, etc.
- Exception handling

What we did is:- we took a read-only text field, and one can insert and delete a single character at a time just like stack(LIFO) concept, user can write a meaningful mathematical half of an equation of course without a defined variable(x, y, z, a,….etc.) to get the correct output. Then, when the user needs to get a result, he/she should press the equals-two("=") symbol present at one of the buttons defined in the layout, as in routine the string at text field will get transferred or rather I would say passed to a function. In it the string will get evaluated with the use of a predefined function, i.e. eval(), which takes one necessary argument to perform its task and return an output if the string is a valid and structured combination of operator and operands else, it will throw a "keyError:" which we handled to give a default error message. Whatever be the output after evaluation, it will again get displayed in the same read-only text field.

Tkinter

Python offers various utilities to design the Graphical User Interface (GUI), and one such utility is Tkinter which is most commonly used. It is indeed one of the fastest and easiest ways to build a GUI application. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with Tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using Tkinter is an easy task. Moreover, Tkinter is cross-platform, hence the same code works on macOS, Windows, and Linux.

- How to create a Tkinter:

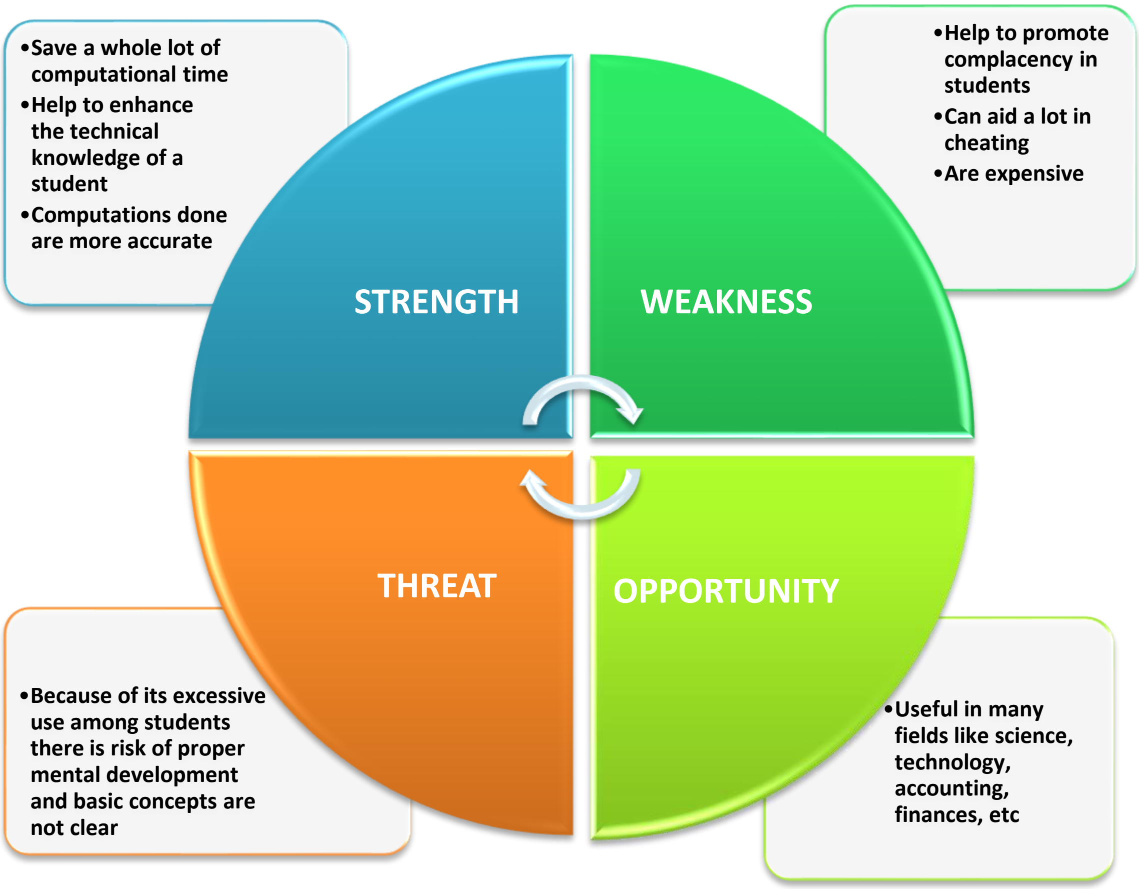
1. Importing the module – Tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

- Tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes' class.

1. **pack() method:** It organizes the widgets in blocks before placing in the parent widget.
2. **grid() method:** It organizes the widgets in a grid (table-like structure) before placing in the parent widget.
3. **place() method:** It organizes the widgets by placing them in specific positions directed by the programmer.

- Tkinter also provides various widgets for performing more tasks and enhancing the outcome of any python program.

SWOT ANALYSIS



STRENGTH

- Save a whole lot of computational time
- Help to enhance the technical knowledge of a student
- Computations done are more accurate

WEAKNESS

- Help to promote complacency in students
- Can aid a lot in cheating
- Are expensive

THREAT

- Because of its excessive use among students there is risk of proper mental development and basic concepts are not clear

OPPORTUNITY

- Useful in many fields like science, technology, accounting, finances, etc