Techvanto Academy

CERTIFICATE OF COMPLETION

This certifies that

KARTHIK KORRAYI

has efficiently completed ***6 Week Live Training*** *in* ***Python Programming***
conducted by ***Techvanto Academy*** *in association with* ***Alma Fiesta***, ***IIT Bhubaneswar***
with an ***A*** *grade on basis of overall performance and evaluation.*

We wish a great success for his/her future endeavours..!!

MR.SHEKHAR SAINI
Director

Alma Fiesta

CERTIFIED ISO 9001:2015 COMPANY

REG. ID
TA2145PPT2106

Date: 1st June'21 -15th July'21

SIX WEEKS SUMMER TRAINING

REPORT

On

PYTHON PROGRAMMING

Submitted by

KARTHIK KORRAYI

Registration No. 11912804

Under the Guidance of

TECHVANTO ACADEMY

School of Computer Science & Engineering

Lovely Professional University, Phagwara

(June-July 2021)

DECLARATION

I hereby declare that I have completed my six weeks summer training at Techvanto Academy from 1st June 2021 to 15th July 2021 under the guidance of Mr. Raj Kumar. I have decided that I have worked with full dedication during these six weeks of training and my learning outcomes fulfil the requirements of training for the award of degree of Computer Science Engineering (BTech), Lovely Professional University, Phagwara.

Karthik Korrayi

Reg. No.: 11912804

Date: 15/07/2021

Content

Introduction

Python is a widely used general-purpose, **high level programming language**. It was created by Guido Van Rossum in 1991 and further developed by Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.

Python is **used for** web development, AI, machine learning, operating systems, mobile application development and video games...

It is very abstract and uses natural language elements, which are easier to use and understand.

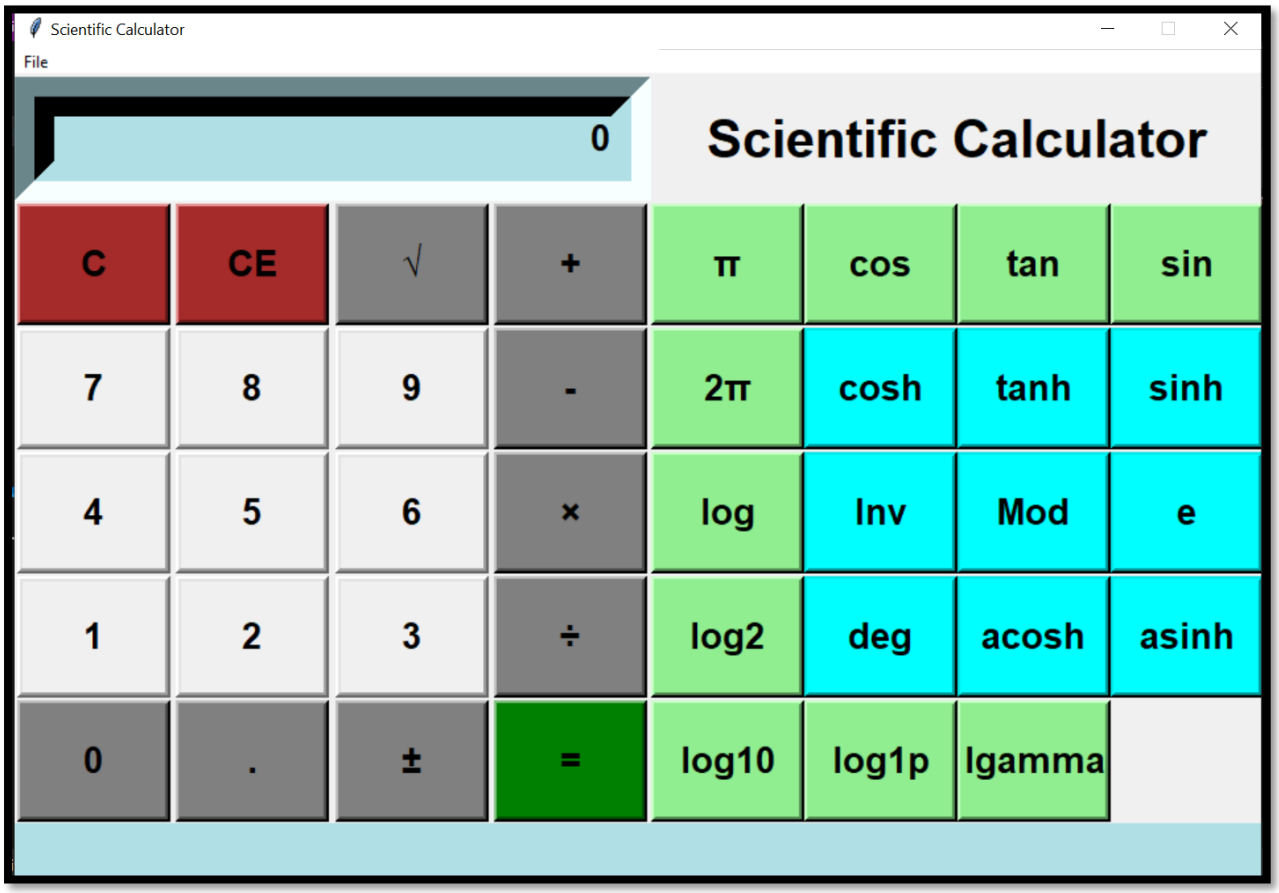
It supports modules and packages, which encourages program modularity and code reuse.

Features in python:

1. Easy to code
2. Free and open sourced
3. Object-oriented and integrated language
4. Large standard library
5. GUI programming support

Technology learnt

While coming to this phase, I have created an application programming i.e., **Scientific Calculator** and its actual purpose in it designed to help us to calculate science engineering, and mathematics problems.



It has a scope like calculating basic functions and exponents, logarithms like some formulas helps us to calculate speed, area, and much more, having built-in memory which allows us to store certain equations.

It also calculates trigonometric functions, binary functions, scientific notations too.

Reason for choosing this technology

This scientific calculator is 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.

Basic calculators are great for solving simple equations with one or two variables, but scientific calculators allow us to input a problem that has an order of operations.

Problem Analysis

Scientific Calculator – A fully featured with proper operator precedence is implemented, including trigonometric functions and logarithms, factorials, 12 level of parentheses, logs to base 2, bitwise logical operators, hex, octal, binary and ASCII display.

Feasibility Analysis –

This study reveals all the technical aspects and its corresponding results. The project entitles **"Scientific Calculator"** is technically feasible because that the project was developed in using Python Programming Language with Graphical User Interface. It provides High-Level of reliability, availability, and compatibility.

As far as the project is concerned, there are financial benefits arising due to improved reliability and accuracy of information. The computerized system will help in automate the selection leading the profits and detail of the organization.

With this application, the machine and manpower utilization are expected to go up by 80-90% approximately. The system is easier to handle and does not require special training to handle it.

Software requirement analysis

- Modern Operating System required minimum of windows 7 or 10 with 4GB RAM in minimum required i3 intel processors in 64-bit CPU as intel/AMD architecture or Linux-Ubuntu 16.04 to 17.10.

- Required 5GB free disk space for Python latest version IDLE module document to write code (in my suggestion).

- We can use PyCharm or Spyder software's instead of Python IDLE, an open-source IDE for Python development.

- Also, it needs at least one Python installation to be available on the machine as a software creates an isolated virtual environment and can change t or create new interpreters according to the previous point.

Design

Tables and their relationships-

This scientific calculator project is in python. Talking about the feature of this system, this python application is designed to calculate entered numbers to any operations and it is also capable of handling all types of exceptions. Also, the design of this system is simple. so that the user won't get any difficulties while working on it.

Pseudo code-

#Beginning by importing all libraries

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

root = Tk()
root.title("Scientific Calculator")
root.configure(background="Powder blue")
root.resizable(width=False, height=False)
root.geometry("480x624+20+20")

calc = Frame(root)
calc.grid()
```

#creating functions

```
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 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
        if self.op=="mod":
            self.total%=self.current
        if self.op=="inv":
            self.total=1/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 tanh(self):
        self.reult=False
        self.current=math.tanh(math.radians(float(txtDisplay.get())))
        self.display(self.current)

    def tan(self):
        self.reult=False
```

```
        self.current=math.tan(math.radians(float(txtDisplay.get())))
        self.display(self.current)

    def sinh(self):
        self.reult=False
        self.current=math.sinh(math.radians(float(txtDisplay.get())))
        self.display(self.current)

    def sin(self):
        self.reult=False
        self.current=math.sin(math.radians(float(txtDisplay.get())))
        self.display(self.current)

    def log(self):
        self.reult=False
        self.current=math.log(float(txtDisplay.get()))
        self.display(self.current)

    def exp(self):
        self.reult=False
        self.current=math.exp(float(txtDisplay.get()))
        self.display(self.current)

    def mathsPM(self):
        self.reult=False
```

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

    def squared(self):
        self.reult=False
        self.current=math.sqrt(float(txtDisplay.get()))
        self.display(self.current)

    def cos(self):
        self.reult=False
        self.current=math.cos(math.radians(float(txtDisplay.get())))
        self.display(self.current)

    def cosh(self):
        self.reult=False
        self.current=math.cosh(math.radians(float(txtDisplay.get())))
        self.display(self.current)



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

    def pi(self):
        self.reult=False
```

```
        self.current=math.pi
        self.display(self.current)


    def tau(self):
        self.reult=False
        self.current=math.tau
        self.display(self.current)

    def e(self):
        self.reult=False
        self.current=math.e
        self.display(self.current)

    def acosh(self):
        self.result=False
        self.current=math.acosh(float(txtDisplay.get()))
        self.display(self.current)

    def asinh(self):
        self.result=False
        self.current=math.asinh(float(txtDisplay.get()))
        self.display(self.current)

    def expm1(self):
```

```
        self.result=False
        self.current=math.expm1(float(txtDisplay.get()))
        self.display(self.current)

    def lgamma(self):
        self.result=False
        self.current=math.lgamma(float(txtDisplay.get()))
        self.display(self.current)

    def degrees(self):
        self.result=False
        self.current=math.degrees(float(txtDisplay.get()))
        self.display(self.current)

    def log2(self):
        self.result=False
        self.current=math.log2(float(txtDisplay.get()))
        self.display(self.current)

    def log10(self):
        self.result=False
        self.current=math.log10(float(txtDisplay.get()))
        self.display(self.current)

    def log1p(self):
```

```
        self.result=False
        self.current=math.log1p(float(txtDisplay.get()))
        self.display(self.current)


added_value=Calc()
```

#for result display

```
txtDisplay = Entry(calc, relief=SUNKEN, font=('arial', 20, 'bold'), bg="powder blue", bd=30, width=28, justify=RIGHT)
txtDisplay.grid(row=0, column=0, columnspan=4, pady=1)
txtDisplay.insert(0, "0")
```

#for numberpad

```
numberpad = "789456123"
i=0
btn=[]
for j in range(2,5):
    for k in range(3):
        btn.append(Button(calc, width=6, height=2, font=('arial', 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
```

#buttons for standard calculator

```
btnClear=Button(calc, text=chr(67), width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="brown", command=added_value.Clear_Entry).grid(row=1, column=0, pady=1)

btnAllClear=Button(calc, text=chr(67)+chr(69), width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="brown", command=added_value.all_Clear_Entry).grid(row=1, column=1, pady=1)

btnSq=Button(calc, text="√", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey", command=added_value.squared).grid(row=1, column=2, pady=1)

btnAdd=Button(calc, text="+", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey", command=lambda:added_value.operation("add")).grid(row=1, column=3, pady=1)

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

btnMult=Button(calc, text="×", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey", command=lambda:added_value.operation("multi")).grid(row=3, column=3, pady=1)

btnDiv=Button(calc, text=chr(247), width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey", command=lambda:added_value.operation("divide")).grid(row=4, column=3, pady=1)

btnZero=Button(calc, text="0", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey",
```

command=lambda:added_value.numberEnter(0)).grid(row=5, column=0, pady=1)

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

btnPM=Button(calc, text=chr(177), width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="grey", command=added_value.mathsPM).grid(row=5, column=2, pady=1)

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

#buttons for scientific calculator

btnPi=Button(calc, text='π', width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.pi).grid(row=1, column=4, pady=1)

btnCos=Button(calc, text="cos", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.cos).grid(row=1, column=5, pady=1)

btnTan=Button(calc, text="tan", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.tan).grid(row=1, column=6, pady=1)

btnSin=Button(calc, text="sin", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.sin).grid(row=1, column=7, pady=1)

btn2Pi=Button(calc, text='2π', width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.tau).grid(row=2, column=4, pady=1)

```
btnCosh=Button(calc, text="cosh", width=6, height=2, font=('arial',
20, 'bold'), bd=4, bg="aqua",
command=added_value.cosh).grid(row=2, column=5, pady=1)

btnTanh=Button(calc, text="tanh", width=6, height=2, font=('arial',
20, 'bold'), bd=4, bg="aqua",
command=added_value.tanh).grid(row=2, column=6, pady=1)

btnSinh=Button(calc, text="sinh", width=6, height=2, font=('arial', 20,
'bold'), bd=4, bg="aqua", command=added_value.sinh).grid(row=2,
column=7, pady=1)

btnLog=Button(calc, text='log', width=6, height=2, font=('arial', 20,
'bold'), bd=4,bg="light green",
command=added_value.log).grid(row=3, column=4, pady=1)

btninv=Button(calc, text="Inv", width=6, height=2, font=('arial', 20,
'bold'), bd=4, bg="aqua",
command=lambda:added_value.operation("inv")).grid(row=3,
column=5, pady=1)

btnMod=Button(calc, text="Mod", width=6, height=2, font=('arial',
20, 'bold'), bd=4, bg="aqua",
command=lambda:added_value.operation("mod")).grid(row=3,
column=6, pady=1)

btnE=Button(calc, text="e", width=6, height=2, font=('arial', 20,
'bold'), bd=4, bg="aqua", command=added_value.e).grid(row=3,
column=7, pady=1)

btnLog2=Button(calc, text='log2', width=6, height=2, font=('arial', 20,
'bold'), bd=4,bg="light green",
command=added_value.log2).grid(row=4, column=4, pady=1)

btnDeg=Button(calc, text="deg", width=6, height=2, font=('arial', 20,
'bold'), bd=4, bg="aqua",
command=added_value.degrees).grid(row=4, column=5, pady=1)
```

```
btnAcosh=Button(calc, text="acosh", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="aqua", command=added_value.acosh).grid(row=4, column=6, pady=1)

btnAsinh=Button(calc, text="asinh", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="aqua", command=added_value.asinh).grid(row=4, column=7, pady=1)

btnLog10=Button(calc, text='log10', width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.log10).grid(row=5, column=4, pady=1)

btnLog1p=Button(calc, text="log1p", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.log1p).grid(row=5, column=5, pady=1)

btnLgamma=Button(calc, text="lgamma", width=6, height=2, font=('arial', 20, 'bold'), bd=4, bg="light green", command=added_value.lgamma).grid(row=5, column=6, pady=1)
```

#extra presentation

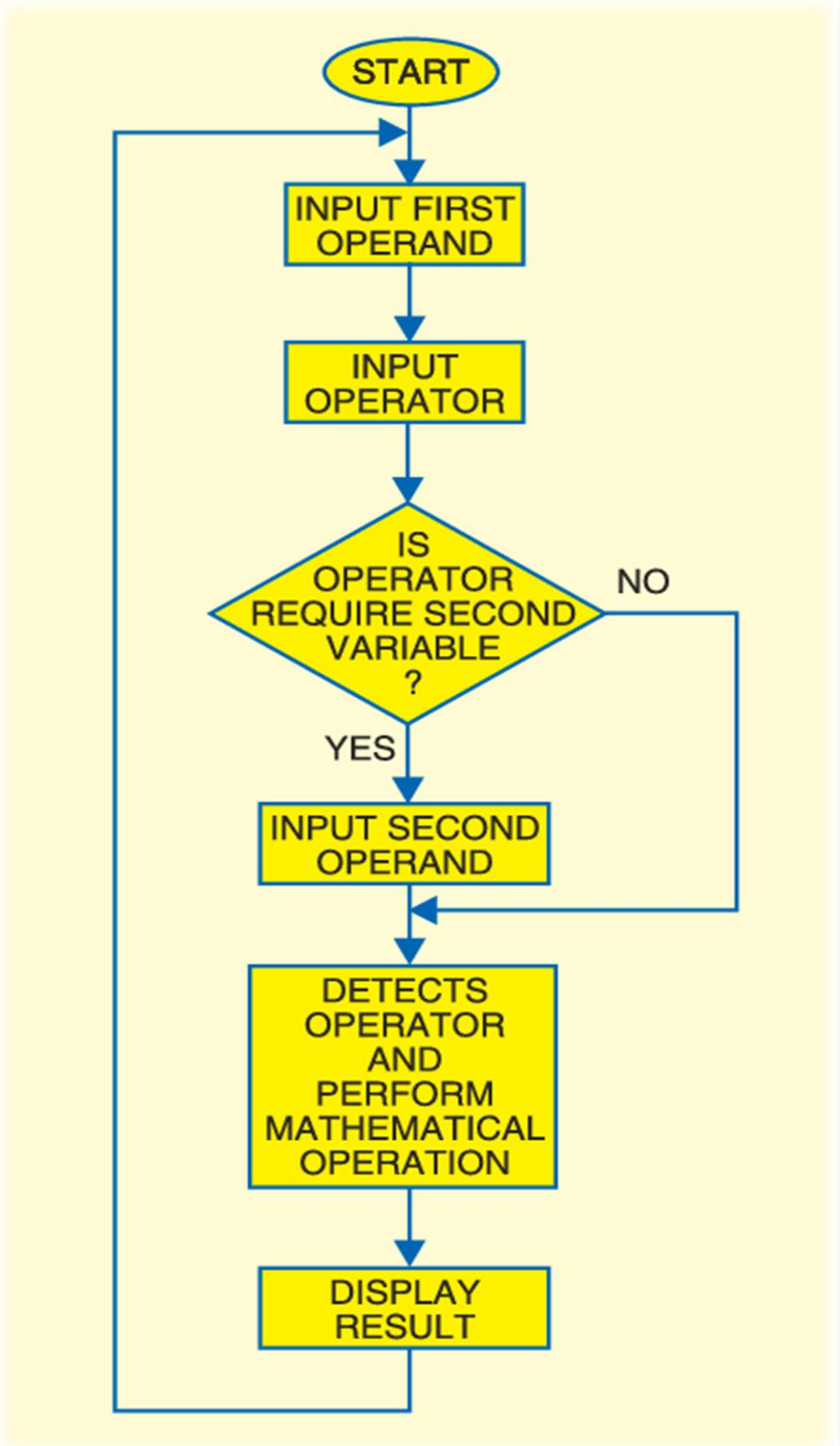
```
lblDisplay=Label(calc, text="Scientific Calculator", font=('arial', 30, 'bold'), justify =CENTER)

lblDisplay.grid(row=0, column=4, columnspan=4)
```

#adding functions

```
def iExit():
    iExit = tkinter.messagebox.askyesno("Scientific Calculator", "Confirm if you want to exit")
    if iExit>0:
        root.destroy()
        return
```

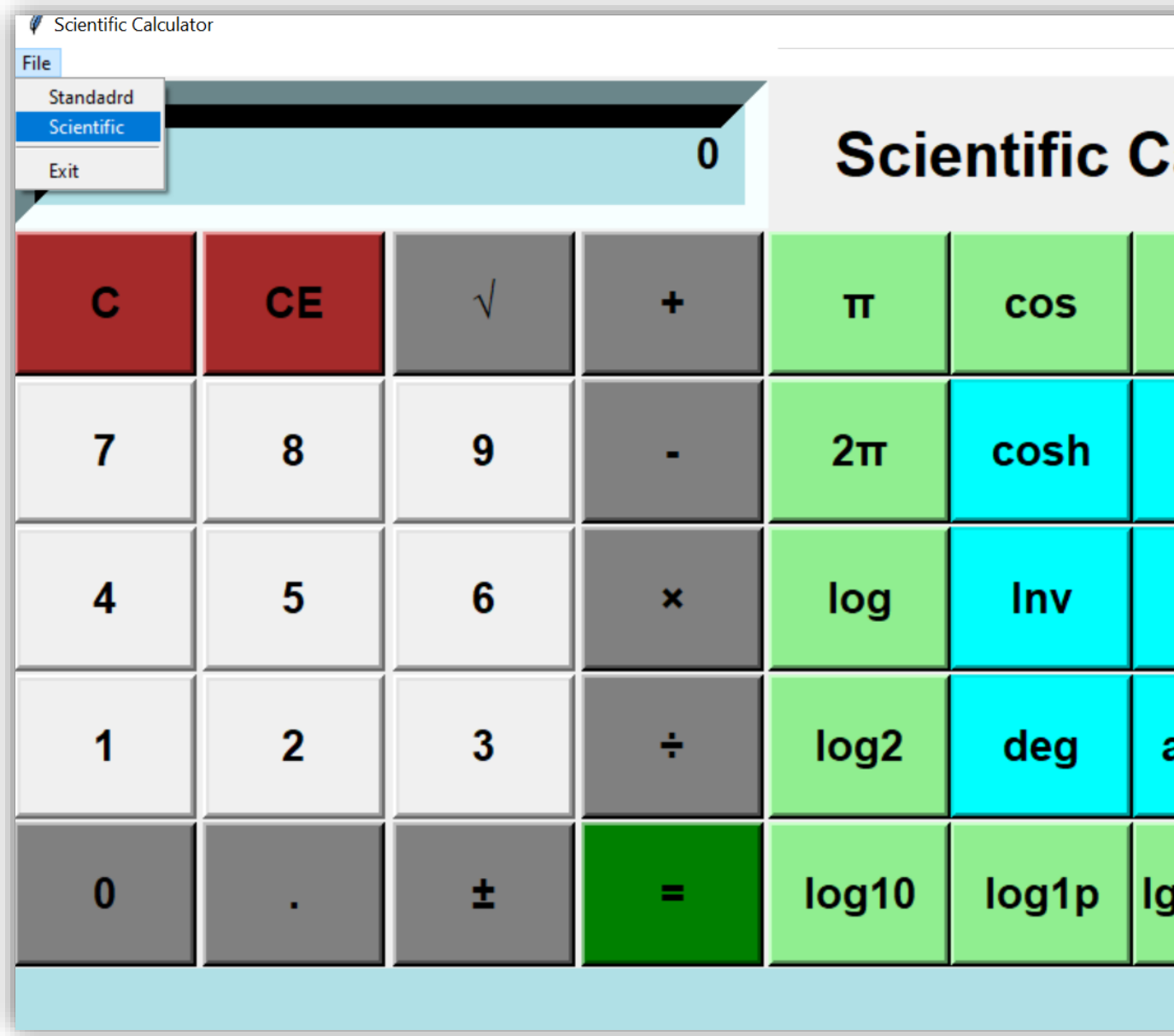
```
def Scientific():
    root.resizable(width=False, height=False)
    root.geometry("944x624+20+20")

def Standard():
    root.resizable(width=False, height=False)
    root.geometry("480x624+20+20")

menubar = Menu(calc)

filemenu = Menu(menubar, tearoff=0)
menubar.add_cascade(label = "File", menu=filemenu)
filemenu.add_command(label = "Standadrd", command = Standard)
filemenu.add_command(label = "Scientific", command = Scientific)
filemenu.add_separator()
filemenu.add_command(label = "Exit", command = iExit)

#main loop
root.config(menu=menubar)
root.mainloop()
```

Flow Chart-


START
INPUT FIRST OPERAND
INPUT OPERATOR
IS OPERATOR REQUIRE SECOND VARIABLE ?
NO
YES
INPUT SECOND OPERAND
DETECTS OPERATOR AND PERFORM MATHEMATICAL OPERATION
DISPLAY RESULT

Implementation

This application avoids the manual work and the problems concern with it. Well, I have worked hard to present an improved project better than the existing one's in my point of view regarding the information about the various activities. Still, I found out that the project can be done in better way that I can add more functions too.



Learning Outcomes from training/technology learnt

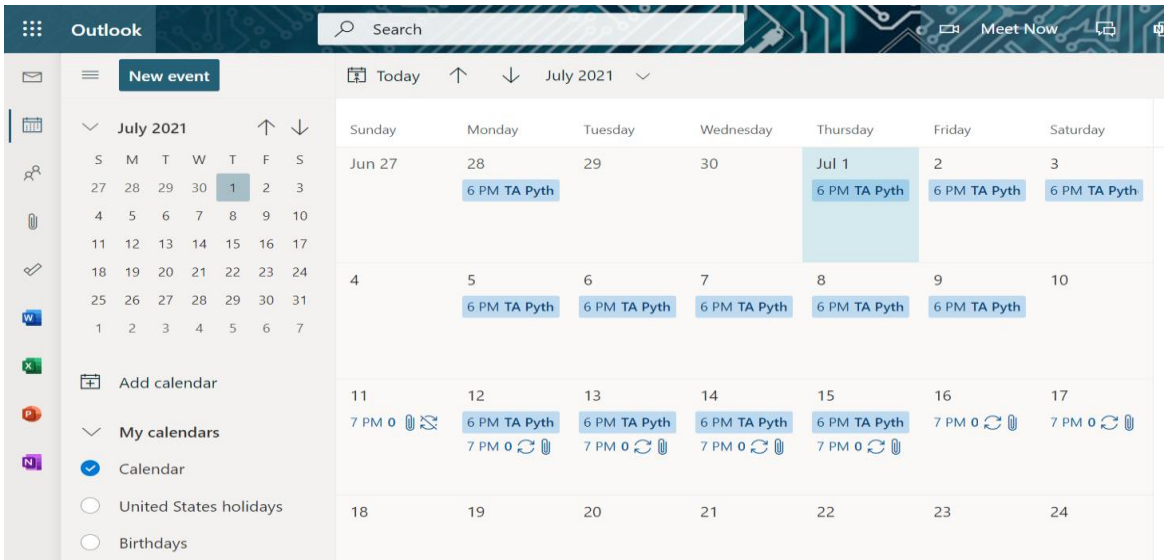
This project will be able to be understanding the way that how python code works after making this programming as well as other applications too that will be mentioned after this.

Gantt chart

(Describing a timeline of how you acted in six weeks of training?)

Today June 2021

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
May 30	31	Jun 1	2	3 7 PM TA Pyth	4 7 PM TA Pyth	5 7 PM TA Pyth
6	7 6 PM TA Pyth	8 6 PM TA Pyth	9 6 PM TA Pyth	10 6 PM TA Pyth	11	12
13	14 6 PM TA Pyth	15 6 PM TA Pyth	16 6 PM TA Pyth	17 6 PM TA Pyth	18 6 PM TA Pyth	19 6 PM TA Pyth
20	21 6 PM TA Pyth	22 6 PM TA Pyth	23 6 PM TA Pyth	24 6 PM TA Pyth	25 6 PM TA Pyth	26 6 PM TA Pyth
27	28 6 PM TA Pyth	29	30	Jul 1 6 PM TA Pyth	2 6 PM TA Pyth	3 6 PM TA Pyth



Project Legacy

- ✓ I learned to always prepare ahead for team meetings by having the goal and the objective of the meeting in mind. I also planned an agenda and questions that can be asked during meeting as well. It helped in getting value out of team meetings.

- ✓ And, I learned to give actionable, specific and kind feedback and to highlight what the programmers did well each week and also, I could improve on.

- ✓ I learned to always try to find out what is going on with my team members asides work and being patient with each and everyone.

- ✓ I learned how to schedule my day and manage my time effectively by always planning my day ahead and sticking to the plan for the day. I was more productive and accomplished more of my goals this way.

Bibliography

- In my opinion, I have successfully completed my project by taking the help from online websites and from YouTube.

- Mainly I had taken help of faculty from the Techvanto Academy, by his lecture course.