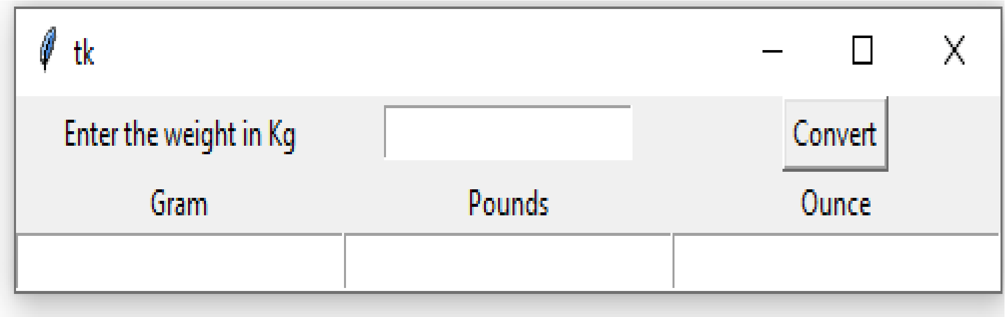
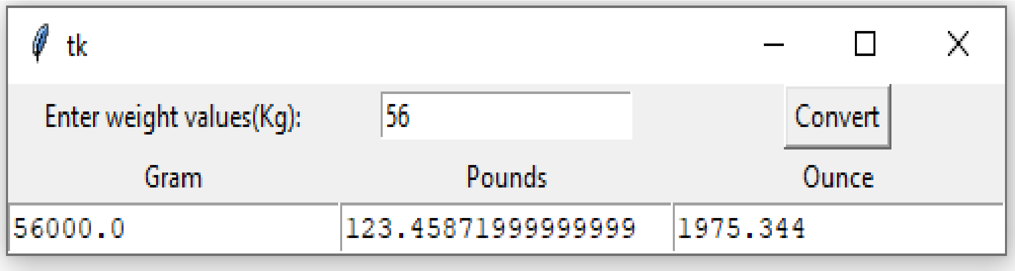
1. Create a GUI based weight converter that accepts a kilogram input value and converts that value to grams, pounds, and ounces when the user clicks the Convert button.





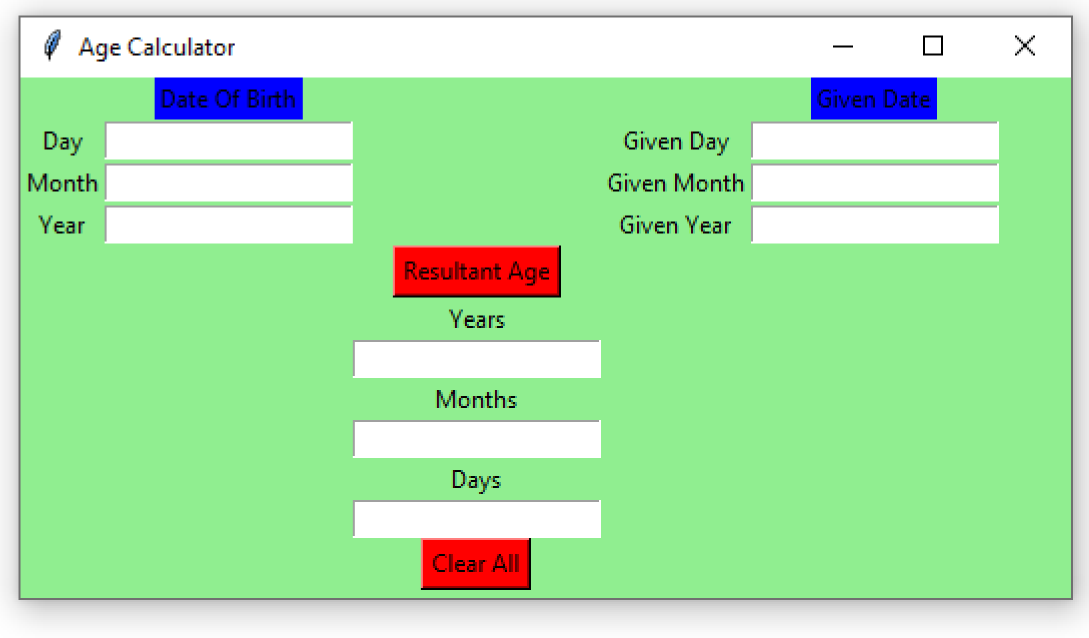
```
56
56000.0
123.45871999999999
1975.344
```

2. Create a GUI program to accept three subjects' marks in input value and calculate the pass or fail student result. All subject must be greater than or equal 50, the student will pass the exam. Otherwise the student will fail the exam grade. The result will display in label when the user clicks the Result button.

```
# Python program to create a simple GUI

# weight converter using Tkinter

from tkinter import *

# Create a GUI window

window = Tk()

# Function to convert weight # given in kg to grams, pounds # and ounces

def from_kg():


    # convert kg to gram

    gram = float(e2_value.get())*1000
    # convert kg to pound
    pound = float(e2_value.get())*2.20462

    # convert kg to ounce
    ounce = float(e2_value.get())*35.274

    # Enters the converted weight to
    # the text widget
    t1.delete("1.0", END)
    t1.insert(END,gram)

    t2.delete("1.0", END)
    t2.insert(END,pound)

    t3.delete("1.0", END)
    t3.insert(END,ounce)

# Create the Label widgets
e1 = Label(window, text = "Enter the weight in Kg")
e2_value = StringVar()
e2 = Entry(window, textvariable = e2_value)
e3 = Label(window, text = 'Gram')
```

```
e4 = Label(window, text = 'Pounds')
e5 = Label(window, text = 'Ounce')

# Create the Text Widgets
t1 = Text(window, height = 1, width = 20)
t2 = Text(window, height = 1, width = 20)
t3 = Text(window, height = 1, width = 20)

# Create the Button Widget
b1 = Button(window, text = "Convert", command = from_kg)

# grid method is used for placing # the widgets at respective positions # in table like structure
e1.grid(row = 0, column = 0)
e2.grid(row = 0, column = 1)
e3.grid(row = 1, column = 0)
e4.grid(row = 1, column = 1)
e5.grid(row = 1, column = 2)
t1.grid(row = 2, column = 0)
t2.grid(row = 2, column = 1)
t3.grid(row = 2, column = 2)
b1.grid(row = 0, column = 2)
# Start the GUI
window.mainloop()
```

2. Create a GUI based simple Age Calculator application that can calculate the age with respect to the given date and birth date, given by the user



```
# import all functions from the tkinter

from tkinter import *



# import messagebox class from tkinter

from tkinter import messagebox



# Function for clearing the

# contents of all text entry boxes

def clearAll() :



        # deleting the content from the entry box

        dayField.delete(0, END)

        monthField.delete(0, END)
```

```
    yearField.delete(0, END)

    givenDayField.delete(0, END)

    givenMonthField.delete(0, END)

    givenYearField.delete(0, END)

    rsltDayField.delete(0, END)

    rsltMonthField.delete(0, END)

    rsltYearField.delete(0, END)


# function for checking error

def checkError() :


    # if any of the entry field is empty

    # then show an error message and clear

    # all the entries

    if (dayField.get() == "" or monthField.get() == ""

        or yearField.get() == "" or givenDayField.get() == ""

        or givenMonthField.get() == "" or givenYearField.get() == "") :


        # show the error message

        messagebox.showerror("Input Error")


        # clearAll function calling

        clearAll()
        return -1


# function to calculate Age
def calculateAge() :


    # check for error
    value = checkError()


    # if error is occur then return
    if value == -1 :
        return


    else :


    # take a value from the respective entry boxes # get method returns current text as string
        birth_day = int(dayField.get())
        birth_month = int(monthField.get())
        birth_year = int(yearField.get())
        given_day = int(givenDayField.get())
        given_month = int(givenMonthField.get())
        given_year = int(givenYearField.get())
```

```
# if birth date is greater then given birth_month
# then donot count this month and add 30 to the date so
# as to subtract the date and get the remaining days
month =[31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

if (birth_day > given_day):
    given_month = given_month - 1
    given_day = given_day + month[birth_month-1]


# if birth month exceeds given month, then
# donot count this year and add 12 to the
# month so that we can subtract and find out
# the difference
if (birth_month > given_month):
    given_year = given_year - 1
    given_month = given_month + 12

# calculate day, month, year
calculated_day = given_day - birth_day;
calculated_month = given_month - birth_month;
calculated_year = given_year - birth_year;
```

```
        # calculated day, month, year write back
        # to the respective entry boxes

        # insert method inserting the
        # value in the text entry box.

        rsltDayField.insert(10, str(calculated_day))
        rsltMonthField.insert(10, str(calculated_month))
        rsltYearField.insert(10, str(calculated_year))


# Driver Code
if __name__ == "__main__" :

    # Create a GUI window
    gui = Tk()

    # Set the background colour of GUI window
    gui.configure(background = "light green")

    # set the name of tkinter GUI window
    gui.title("Age Calculator")
```

```
# Set the configuration of GUI window
gui.geometry("525x260")

# Create a Date Of Birth : label
dob = Label(gui, text = "Date Of Birth", bg = "blue")

# Create a Given Date : label
givenDate = Label(gui, text = "Given Date", bg = "blue")

# Create a Day : label
day = Label(gui, text = "Day", bg = "light green")

# Create a Month : label
month = Label(gui, text = "Month", bg = "light green")

# Create a Year : label
year = Label(gui, text = "Year", bg = "light green")

# Create a Given Day : label
givenDay = Label(gui, text = "Given Day", bg = "light green")

# Create a Given Month : label
```

```
    givenMonth = Label(gui, text = "Given Month", bg = "light green")


    # Create a Given Year : label

    givenYear = Label(gui, text = "Given Year", bg = "light green")


    # Create a Years : label

    rsltYear = Label(gui, text = "Years", bg = "light green")


    # Create a Months : label

    rsltMonth = Label(gui, text = "Months", bg = "light green")


    # Create a Days : label

    rsltDay = Label(gui, text = "Days", bg = "light green")


    # Create a Resultant Age Button and attached to calculateAge function

    resultantAge = Button(gui, text = "Resultant Age", fg = "Black", bg = "Red", command = calculateAge)


    # Create a Clear All Button and attached to clearAll function

    clearAllEntry = Button(gui, text = "Clear All", fg = "Black", bg = "Red", command = clearAll)


    # Create a text entry box for filling or typing the information.

    dayField = Entry(gui)
```

```
monthField = Entry(gui)
yearField = Entry(gui)

givenDayField = Entry(gui)
givenMonthField = Entry(gui)
givenYearField = Entry(gui)

rsltYearField = Entry(gui)
rsltMonthField = Entry(gui)
rsltDayField = Entry(gui)


# grid method is used for placing
# the widgets at respective positions
# in table like structure .
dob.grid(row = 0, column = 1)

day.grid(row = 1, column = 0)
dayField.grid(row = 1, column = 1)

month.grid(row = 2, column = 0)
monthField.grid(row = 2, column = 1)
```

```
year.grid(row = 3, column = 0)
yearField.grid(row = 3, column = 1)

givenDate.grid(row = 0, column = 4)

givenDay.grid(row = 1, column = 3)
givenDayField.grid(row = 1, column = 4)

givenMonth.grid(row = 2, column = 3)
givenMonthField.grid(row = 2, column = 4)

givenYear.grid(row = 3, column = 3)
givenYearField.grid(row = 3, column = 4)

resultantAge.grid(row = 4, column = 2)

rsltYear.grid(row = 5, column = 2)
rsltYearField.grid(row = 6, column = 2)

rsltMonth.grid(row = 7, column = 2)
rsltMonthField.grid(row = 8, column = 2)

rsltDay.grid(row = 9, column = 2)
```

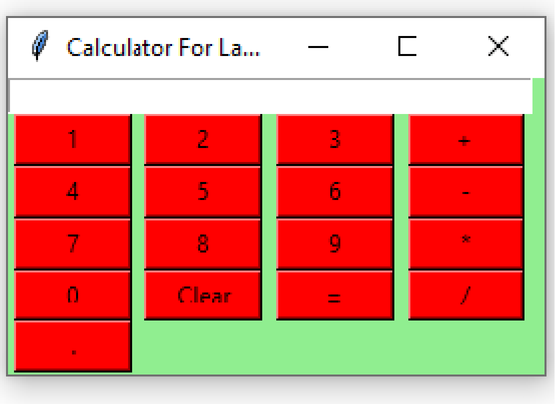
```
rsltDayField.grid(row = 10, column = 2)


clearAllEntry.grid(row = 12, column = 2)


# Start the GUI

gui.mainloop()
```

3. Create a GUI based simple calculator using the Python Tkinter module, which can perform basic arithmetic operations addition, subtraction, multiplication, and division.

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.

Below is what the GUI looks like:



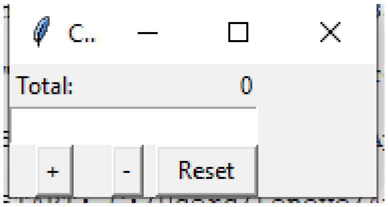
```
# Python program to create a simple GUI

# calculator using Tkinter
# import everything from tkinter module
from tkinter import *

# globally declare the expression variable
expression = ""


# Function to update expression
# in the text entry box
def press(num):
    # point out the global expression variable
    global expression

    # concatenation of string
    expression = expression + str(num)

    # update the expression by using set method
    equation.set(expression)


# Function to evaluate the final expression
def equalpress():
    # Try and except statement is used
    # for handling the errors like zero

    # division error etc.


    # Put that code inside the try block

    # which may generate the error

    try:


            global expression


            # eval function evaluate the expression

            # and str function convert the result

            # into string

            total = str(eval(expression))


            equation.set(total)


            # initialize the expression variable

            # by empty string

            expression = ""


    # if error is generate then handle

    # by the except block

    except:
            equation.set(" error ")
            expression = ""


    # Function to clear the contents
    # of text entry box
    def clear():
        global expression
        expression = ""
        equation.set("")


    # Driver code
    if __name__ == "__main__":
        # create a GUI window
        gui = Tk()

        # set the background colour of GUI window
        gui.configure(background="light green")

        # set the title of GUI window
        gui.title("Calculator For Lab Exam")
```

```
# set the configuration of GUI window
gui.geometry("270x150")

# StringVar() is the variable class
# we create an instance of this class
equation = StringVar()

# create the text entry box for
# showing the expression .
expression_field = Entry(gui, textvariable=equation)

# grid method is used for placing
# the widgets at respective positions
# in table like structure .
expression_field.grid(columnspan=4, ipadx=70)

# create a Buttons and place at a particular
# location inside the root window .
# when user press the button, the command or
# function affiliated to that button is executed .
button1 = Button(gui, text=' 1 ', fg='black', bg='red',
                                 command=lambda: press(1), height=1, width=7)
```

```
button1.grid(row=2, column=0)

button2 = Button(gui, text=' 2 ', fg='black', bg='red',
                           command=lambda: press(2), height=1, width=7)
button2.grid(row=2, column=1)

button3 = Button(gui, text=' 3 ', fg='black', bg='red',
                           command=lambda: press(3), height=1, width=7)
button3.grid(row=2, column=2)

button4 = Button(gui, text=' 4 ', fg='black', bg='red',
                           command=lambda: press(4), height=1, width=7)
button4.grid(row=3, column=0)

button5 = Button(gui, text=' 5 ', fg='black', bg='red',
                           command=lambda: press(5), height=1, width=7)
button5.grid(row=3, column=1)

button6 = Button(gui, text=' 6 ', fg='black', bg='red',
                           command=lambda: press(6), height=1, width=7)
button6.grid(row=3, column=2)

button7 = Button(gui, text=' 7 ', fg='black', bg='red',
                    command=lambda: press(7), height=1, width=7)
button7.grid(row=4, column=0)

button8 = Button(gui, text=' 8 ', fg='black', bg='red',
                    command=lambda: press(8), height=1, width=7)
button8.grid(row=4, column=1)

button9 = Button(gui, text=' 9 ', fg='black', bg='red',
                    command=lambda: press(9), height=1, width=7)
button9.grid(row=4, column=2)

button0 = Button(gui, text=' 0 ', fg='black', bg='red',
                    command=lambda: press(0), height=1, width=7)
button0.grid(row=5, column=0)

plus = Button(gui, text=' + ', fg='black', bg='red',
               command=lambda: press("+"), height=1, width=7)
plus.grid(row=2, column=3)

minus = Button(gui, text=' - ', fg='black', bg='red',
               command=lambda: press("-"), height=1, width=7)
minus.grid(row=3, column=3)
```

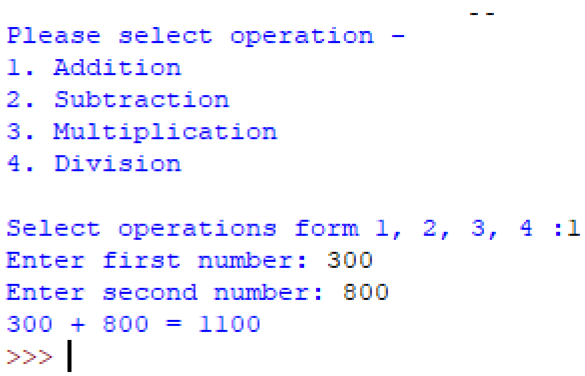
```
multiply = Button(gui, text=' * ', fg='black', bg='red',
                                        command=lambda: press("*"), height=1, width=7)
multiply.grid(row=4, column=3)


divide = Button(gui, text=' / ', fg='black', bg='red',
                                        command=lambda: press("/"), height=1, width=7)
divide.grid(row=5, column=3)


equal = Button(gui, text=' = ', fg='black', bg='red',
                              command=equalpress, height=1, width=7)
equal.grid(row=5, column=2)


clear = Button(gui, text='Clear', fg='black', bg='red',
                              command=clear, height=1, width=7)
clear.grid(row=5, column='1')


Decimal= Button(gui, text='.', fg='black', bg='red',
                                        command=lambda: press('.'), height=1, width=7)
Decimal.grid(row=6, column=0)
# start the GUI
gui.mainloop()
```

4.



```
from tkinter import Tk, Label, Button, Entry, IntVar, END, W, E


class Calculator:


    def __init__(self, master):

        self.master = master

        master.title("Calculator")


        self.total = 0

        self.entered_number = 0


        self.total_label_text = IntVar()

        self.total_label_text.set(self.total)

        self.total_label = Label(master, textvariable=self.total_label_text)


        self.label = Label(master, text="Total:")


        vcmd = master.register(self.validate) # we have to wrap the command

        self.entry = Entry(master, validate="key", validatecommand=(vcmd, '%P'))
        self.add_button = Button(master, text="+", command=lambda: self.update("add"))
        self.subtract_button = Button(master, text="-", command=lambda: self.update("subtract"))
        self.reset_button = Button(master, text="Reset", command=lambda: self.update("reset"))

        # LAYOUT


        self.label.grid(row=0, column=0, sticky=W)
        self.total_label.grid(row=0, column=1, columnspan=2, sticky=E)


        self.entry.grid(row=1, column=0, columnspan=3, sticky=W+E)


        self.add_button.grid(row=2, column=0)
        self.subtract_button.grid(row=2, column=1)
        self.reset_button.grid(row=2, column=2, sticky=W+E)


    def validate(self, new_text):
        if not new_text: # the field is being cleared
            self.entered_number = 0
            return True


        try:
            self.entered_number = int(new_text)
            return True
        except ValueError:
            return False

    def update(self, method):
        if method == "add":
            self.total += self.entered_number
        elif method == "subtract":
            self.total -= self.entered_number
        else: # reset
            self.total = 0

        self.total_label_text.set(self.total)
        self.entry.delete(0, END)

root = Tk()
my_gui = Calculator(root)
root.mainloop()
```

i. Write a simple calculator program that can perform four arithmetic operations like addition, subtraction, multiplication or division depending upon the user input. User choose the desired operation. Options 1, 2, 3 and 4 are valid operations. Two numbers are taken from user input and an if…elif…else branching is used to execute a particular operations. Using functions add (), subtract (), multiply () and divide () evaluate respective operations. The interactive program design is described as in Figure.1.

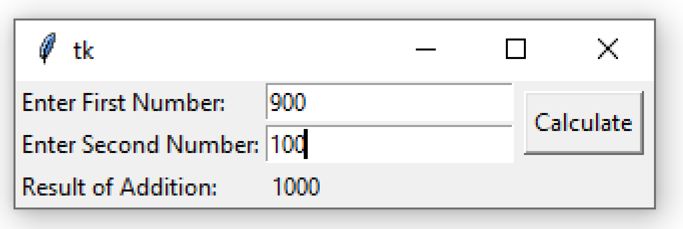


```
                                              --
Please select operation -
1. Addition
2. Subtraction
3. Multiplication
4. Division

Select operations form 1, 2, 3, 4 :1
Enter first number: 300
Enter second number: 800
300 + 800 = 1100
>>> 
```

```
# Python program for simple calculator


# Function to add two numbers

def add(num1, num2):

	return num1 + num2


# Function to subtract two numbers

def subtract(num1, num2):

	return num1 - num2


# Function to multiply two numbers

def multiply(num1, num2):

	return num1 * num2


# Function to divide two numbers

def divide(num1, num2):

	return num1 / num2
```

```
print("Please select operation -\n" \
        "1. Add\n" \
        "2. Subtract\n" \
        "3. Multiply\n" \
        "4. Divide\n")


# Take input from the user
select = int(input("Select operations form 1, 2, 3, 4 :"))

number_1 = int(input("Enter first number: "))
number_2 = int(input("Enter second number: "))

if select == 1:
    print(number_1, "+", number_2, "=",
                    add(number_1, number_2))

elif select == 2:
    print(number_1, "-", number_2, "=",
                    subtract(number_1, number_2))

elif select == 3:
        print(number_1, "*", number_2, "=",

                                        multiply(number_1, number_2))



elif select == 4:

        print(number_1, "/", number_2, "=",

                                        divide(number_1, number_2))

else:

        print("Invalid input")
```

5. Create a GUI program uses tkinter module to create 4 label widgets, two entry widgets and one button. The user will enter two numbers in the two entry widgets. The result of addition will be shown in result label when the button is clicked by the user.



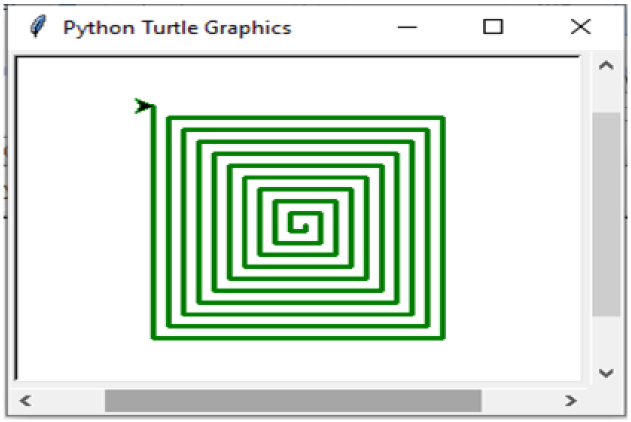
Enter First Number: 900

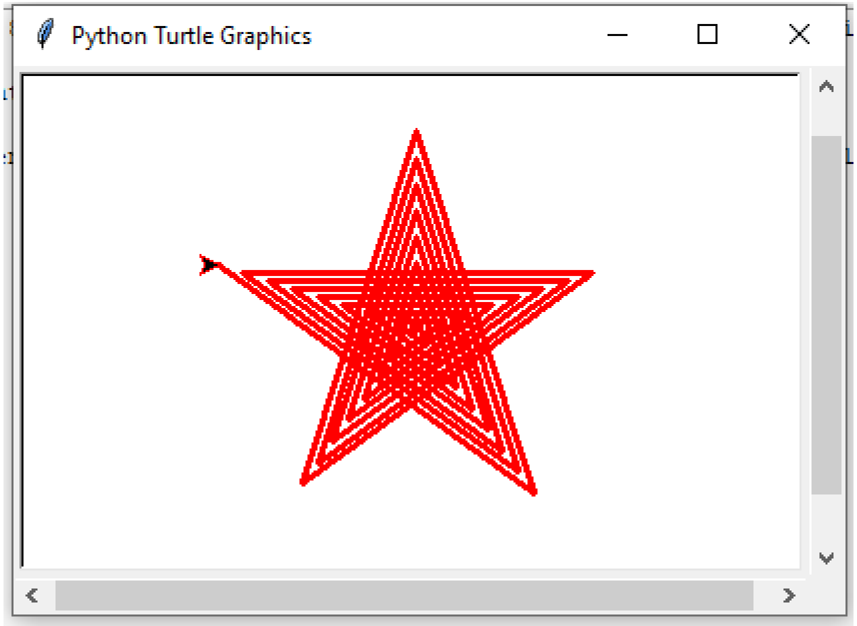
Enter Second Number: 100

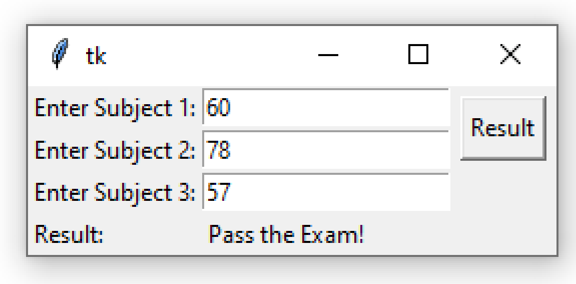
Result of Addition: 1000

```
# Python GUI program to

# add two numbers

# Using Labels, Entry and Button

# widgets - Python 3 tkinter module



from tkinter import *



def add_numbers():

   res=int(e1.get())+int(e2.get())
```

```
    label_text.set(res)

window = Tk()
label_text=StringVar();
Label(window, text="Enter First Number:").grid(row=0, sticky=W)
Label(window, text="Enter Second Number:").grid(row=1, sticky=W)
Label(window, text="Result of Addition:").grid(row=3, sticky=W)
result=Label(window, text="", textvariable=label_text).grid(row=3,column=1, sticky=W)

e1 = Entry(window)
e2 = Entry(window)

e1.grid(row=0, column=1)
e2.grid(row=1, column=1)

b = Button(window, text="Calculate", command=add_numbers)
b.grid(row=0, column=2,columnspan=2, rowspan=2,sticky=W+E+N+S, padx=5, pady=5)


mainloop()
```

6. 

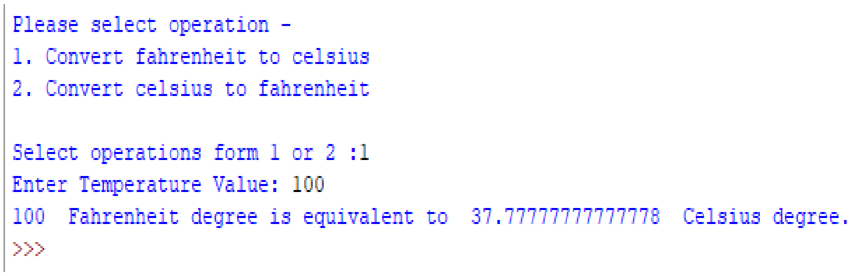




```
Enter Subject 1: 60
Enter Subject 2: 78
Enter Subject 3: 57
Result:          Pass the Exam!
```

```
# Python GUI program to

# calculate the exam result
# Using Labels, Entry and Button
# widgets - Python 3 tkinter module

from tkinter import *

def exam_result():

    s1=int(e1.get())
    s2=int(e2.get())
    s3=int(e3.get())

    if(s1>=50 and s2>=50 and s3>=50):
        label_text.set("Pass the Exam!")
    else:
        label_text.set("Fail the Exam!")

window = Tk()
label_text=StringVar();
Label(window, text="Enter Subject 1:").grid(row=0, sticky=W)
Label(window, text="Enter Subject 2:").grid(row=1, sticky=W)
Label(window, text="Enter Subject 3:").grid(row=2, sticky=W)
Label(window, text="Result:").grid(row=3, sticky=W)
result=Label(window, text="", textvariable=label_text).grid(row=3,column=1, sticky=W)
```

```
e1 = Entry(window)
e2 = Entry(window)
e3 = Entry(window)

e1.grid(row=0, column=1)
e2.grid(row=1, column=1)
e3.grid(row=2, column=1)

b = Button(window, text="Result", command=exam_result)
b.grid(row=0, column=2,columnspan=2, rowspan=2,sticky=W+E+N+S, padx=5, pady=5)


mainloop()
```



```
Please select operation -
1. Convert fahrenheit to celsius
2. Convert celsius to fahrenheit

Select operations form 1 or 2 :1
Enter Temperature Value: 100
100  Fahrenheit degree is equivalent to  37.77777777777778  Celsius degree.
>>>
```

```
# Python program for Temperature Conversion

# Function to convert celsius
def convertcelsius(num):
    return (5.0/9.0)*(num-32)

# Function to convert fahrenheit
def convertfahrenheit(num):
    return (num*(9.0/5.0))+32

print("Please select operation -\n" \
        "1. Convert fahrenheit to celsius\n" \
        "2. Convert celsius to fahrenheit\n")

# Take input from the user
select = int(input("Select operations form 1 or 2 :"))

num = int(input("Enter Temperature Value: "))

if select == 1:
    print(num, " Fahrenheit degree is equivalent to ", convertcelsius(num)," Celsius degree.")

elif select == 2:
    print(num, " Celsius degree is equivalent to ", convertfahrenheit(num)," Fahrenheit degree.")

else:
    print("Invalid input")
```