

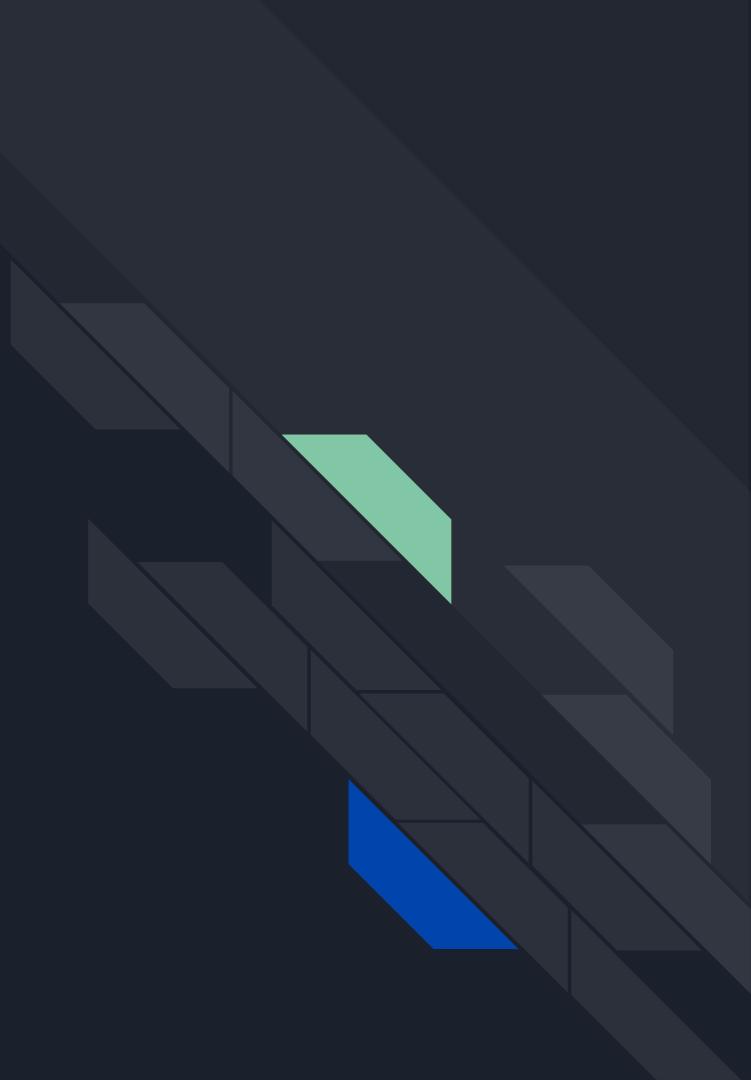


Python Assessments 1

Christian Patrick Z. Calumberan

Chapter I

Python Refresher



E1: User Input Output

```
# Exercise 1: User input output      You, last week • Submittin  
  
# Write code to prompt the user to input her/his name and age ar  
  
# Let's import the string first  
import string  
  
# Then we would take input from user  
name = input("Hello, user! What is your name?: ")  
age = int(input("What is your age?: "))  
  
# Then we add one in the input and find the length of the name  
age = age + 1  
length = len(name)  
  
# Finally, we print to see the results  
print("It's good to meet you, " + name)  
print("The length of your name is: " + str(length))  
print("You will be " + str(age) + " in a year")
```

```
Hello, user! What is your name?: Palpha  
What is your age?: 21  
It's good to meet you, Palpha  
The length of your name is: 6  
You will be 22 in a year
```

E2: Maths

```
# Just like last time, let's import math first
    You, last week • Submitting my exercises so far
import math
    ----

# Next we put the inputs for the user to type in

a=int(input("Enter first number: "))
b=int(input("Enter second number: "))

# Finally, we print to see the results

print("Sum: ",a+b)
print("Difference: ",a-b)
print("Product: ",a*b)
print("Quotient: ",a/b)
print("Remainder: ",a%b)
```

```
Enter first number: 42
Enter second number: 36
Sum: 78
Difference: 6
Product: 1512
Quotient: 1.1666666666666667
Remainder: 6
```

Exercise 3: Is it a Triangle

```
# Let's make three inputs to measure the sides each
a = int(input("Enter the length of the first side: "))
b = int(input("Enter the length of the second side: "))
c = int(input("Enter the length of the third side: "))

# Then we see if two sides are greater than one side using the if-else met
if a+b>c and b+c>a and c+a>b:

    # This should also identify what type of triangle is this
    if a==b==c:
        print("This is an equilateral triangle.")
    elif a==b or b==c or c==a:
        print("This is an isosceles triangle")
    else:
        print ("This is a scalene triangle")

    # Otherwise, it's not even a triangle if typed something unexpected
else:
    print("Oof... this is not a triangle")
```

```
Enter the length of the first side: 3
Enter the length of the second side: 5
Enter the length of the third side: 5
This is an isosceles triangle
```

E4: Largest Number

```
# Three inputs to insert three numbers
a = int(input("a = "))
b = int(input("b = "))
c = int(input("c = "))

# And just like the last time, we compare them using the if-else method
# this time with elif to see the results
if (a>b) and (a>c):
    print("a has the largest number.")
elif (b>a) and (b>c):
    print("b has the largest number.")
else:
    print("c has the largest number.")
```

```
a = 3
b = 4
c = 7
c has the largest number.
```

You, last week • Submitting

E5: Continue

```
# Exercise 5: Continue

# Write a program that implements a while loop. This program should ask i
# range() is vital for this one.
# Any destinative number that makes the limit will cause a continous loop
# until it reach it's destination before the targeted number
for Y in range(13):
    if Y == 10:
        continue
    print(Y)
```

```
0
1
2
3
4
5
6
7
8
9
11
12
```

You, last week • Submitting my exercises so far

E6: FizzBuzz

```
# Exercise 6: FizzBuzz      You, last week • Submitting my exercises so far

# Write a program that prints the numbers from 1 to 100. But for multiples

# fizzbuzz is just... oddly new to me
for fizzbuzz in range(101):
    # if I could tell... numbers that are multiplied by 3 are fizz
    if fizzbuzz % 3 == 0 and fizzbuzz % 5 == 0:
        print("FizzBuzz")
        continue
    # Continuing... numbers that are multiplied by 5 are buzz
    elif fizzbuzz % 5 == 0:
        print("buzz")
        continue
    # And numbers that are both multiplied by 3 and 5, ex. 50, are fizzbuzz
    elif fizzbuzz % 3 == 0:
        print("fizz")
        continue
    print(fizzbuzz)
```

FizzBuzz	
1	71
fizz	fizz
2	73
	74
buzz	FizzBuzz
fizz	76
4	77
fizz	fizz
buzz	79
7	buzz
8	fizz
fizz	82
buzz	buzz
11	fizz
fizz	83
13	FizzBuzz
14	fizz
FizzBuzz	buzz
16	86
17	fizz
fizz	88
19	buzz
buzz	89
fizz	FizzBuzz
22	91
23	92
fizz	fizz
buzz	94
26	buzz
fizz	fizz
28	97
29	FizzBuzz
31	98
32	fizz
	buzz



E7: Even Numbers

```
# Like last time, range() is useful for this one. Making 101 the ending number
for i in range (1, 101):
    if i % 2 != 0:
        continue
    print(i)
```

2	60
4	62
6	64
8	66
10	68
12	70
14	72
16	74
18	76
20	78
22	80
24	82
26	84
28	86
30	88
32	90
34	92
36	94
38	96
40	98
42	100
44	

->

E8: Print Pattern

Exercise 8: Print Pattern You, last week • Submitting my exercises

Write a program to display the following pattern using nested loops.

```
# This one is interesting. We're making a pyramid using input() and range()
e = int(input("enter the number of rows:"))
for i in range(1, e+1):
    for j in range(1, i+1):
        print(j, end="")
    print()
```

```
enter the number of rows:5
1
12
123
1234
12345
```

E9: Integer List

```
# Create an int list with 10 values

ten = [50, 40, 30, 20, 10, 60, 70, 80, 90, 100]

# Output the list using 'a' for loop

for a in ten:
    print(a)

# Output the highest and lowest value

high = max(ten)
low = min(ten)
print ("The highest number is",high)
print ("The lowest number is",low)

# Sort the elements in ascending order

ascend = sorted(ten)
print ("This list is ascended", ascend)

# Sort the elements in descending order

ten.sort(reverse=True)
print ("This list is descended", ten)

# Append two elements
ten.append ([110])      You, 1 second ago • Uncomm
ten.append (120)

# Print the list after appending

print(ten)
```

```
50
40
30
20
10
60
70
80
90
100
```

The highest number is 100

The lowest number is 10

This list is ascended [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

This list is descended [100, 90, 80, 70, 60, 50, 40, 30, 20, 10]
[100, 90, 80, 70, 60, 50, 40, 30, 20, 110, 120]

E10: Film Dictionary

```
# Let's do this with one of my favorite movies to watch from the past 2 years
films = {
    "Movie": "1917",
    "Director": "Sam Mendes",
    "Year": "2019"
}

for key, value in films.items():
    print(key, value)
```

Movie 1917
Director Sam Mendes
Year 2019

You, last week • Submitting my exercises so far

E11: Year Tuples

```
# Exercise 11: Year Tuples      You, last week • Submitting my exercise

# Create a tuple with values

year = (2017, 2003, 2011, 2005, 1987, 2009, 2020, 2018, 2009)

print(year)

# Access the value at index -3

print(year[-3])

# Reverse the tuple and print the original tuple and reversed tuple

r = reversed(year)
print(tuple(r))

# Count number of times 2009 is in the tuple (use count() method)

c = year.count(2009)
print(c)

# Get the index value of 2018(Use index method)

i = year.index(2018)
print(i)

# Find length of given tuple(Use len() method)

print(len(year))
```

```
(2017, 2003, 2011, 2005, 1987, 2009, 2020, 2018, 2009)
2020
(2009, 2018, 2020, 2009, 1987, 2005, 2011, 2003, 2017)
2
7
9
```



E12: Area Functions

```
# Here we go again, this time with Math by importing pi

from math import pi

# Calculating the area of the Square

s = float(input("Enter the side length of the square: "))

area = s * 2

print("The area of the square is: ", area)

# Calculating the area of the Triangle

l = float(input("Enter the base length of the triangle: "))
h = float(input("Enter the base height of the triangle: "))

area = 0.5 * l * h

print("The area of the triangle is: ", area)

# Calculating the area of the Circle

r = float(input("Enter the area of the circle: "))

area = pi * r ** 2

print("The area of the circle with radius ", r , " is: ", area)
```

```
Enter the side length of the square: 5
The area of the square is: 10.0
Enter the base length of the triangle: 4
Enter the base height of the triangle: 5
The area of the triangle is: 10.0
Enter the area of the circle: 1
The area of the circle with radius  1.0  is:  3.141592653589793
```

E13: Product of list items

```
# Exercise 13: Product of list items

# Write a program that passes a list as an argument to a function. The

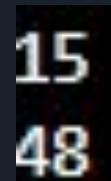
# NumPy, as the name suggests, is for working with numbers in python,
# for operations like multiplication

import numpy

odd = [1, 3, 5]
even = [2, 4, 6]      You, last week • Submitting my exercises so far

result1 = numpy.prod(odd)
result2 = numpy.prod(even)

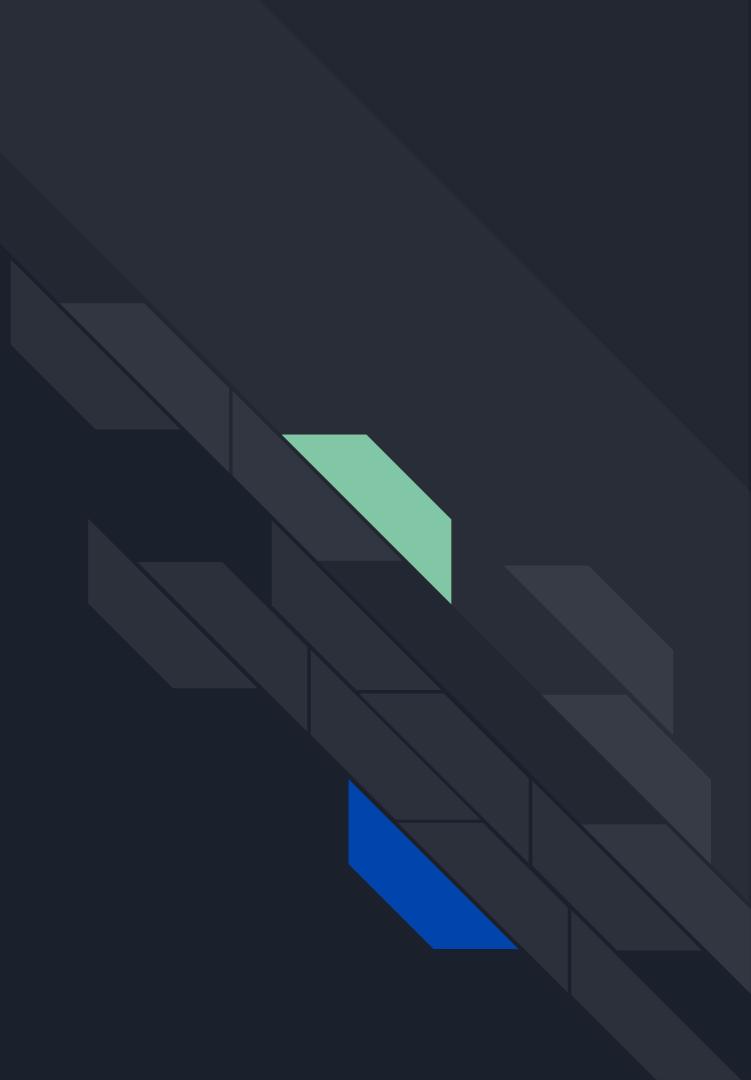
print(result1)
print(result2)
```



```
15
48
```

Chapter II

Graphical User Interface



E1: Welcome

```
# Exercise 1: Welcome

# Develop a GUI program to do the following using the tkinter module

import tkinter
from tkinter import *

main = Tk()

main.title('Exercise 1')      You, last week • Submitting my exercises so far
main.geometry('500x200')

message=Label(main,text="Welcome to Tkinter", bg='skyblue', width=20, height=30, font=('Times',20,'bold'))
message.pack()

main.configure(bg='dodgerblue')
main.resizable(False,False)
main.mainloop()
```



E2: Managing Layout

```
# Exercise 2: Managing Layout

# a: Using Pack
# Create a GUI with 4 labels using the pack() geometry

import tkinter
from tkinter import *

main = Tk()
main.title('Exercise 2 a')

Label(main, text="A", width=12, bg="red", relief=GROOVE, bd=5).pack(side=TOP, expand=1, fill=x)
Label(main, text="B", width=12, bg="yellow").pack(side=BOTTOM)
Label(main, text="C", width=12, bg="blue").pack(side=LEFT, expand=1, fill=y)
Label(main, text="D", width=12, bg="white").pack(side=RIGHT)

main.resizable(True,True)
main.mainloop()
```

```
# B: Square Grid
# With the pack layout manager, create the following

import tkinter
from tkinter import *

main = Tk()
main.title('Exercise 2 b')

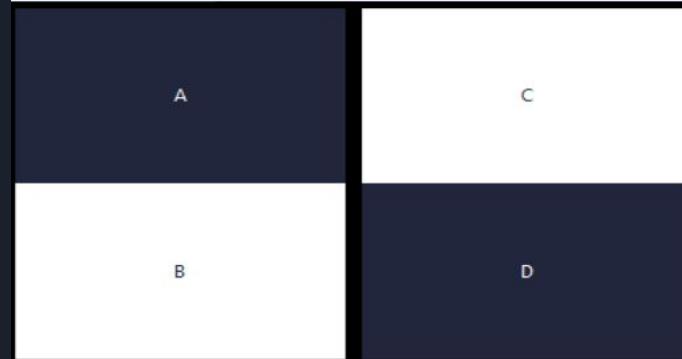
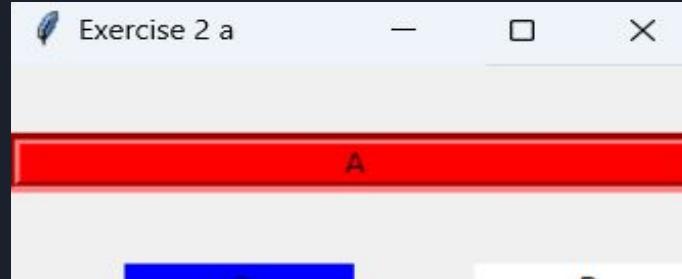
Frame1 = Frame(main,border=5,relief="solid")
Frame1.pack(side=LEFT,expand=True,fill="both")

A = Label(Frame1,text="A",bg="#22263d",fg="white")
B = Label(Frame1,text="B",bg="white",fg="#22263d")
A.pack(side=TOP,expand=True,fill="both")
B.pack(side=BOTTOM,expand=True,fill="both")
You, 5 days ago • Update Exercise 2 b.py

Frame2 = Frame(main,border=5,relief="solid")
Frame2.pack(side=RIGHT,expand=True,fill="both")

C = Label(Frame2,text="C",bg="white",fg="#22263d")
D = Label(Frame2,text="D",bg="#22263d",fg="white")
C.pack(side=TOP,expand=True,fill="both")
D.pack(side=BOTTOM,expand=True,fill="both")

main.resizable(True,True)
main.mainloop()
```



E3: Login Page

```
# Exercise 3: Login Page

# Create a login page using the Grid geometry
import tkinter
from tkinter import *
main = Tk()

main.title('Exercise 3')
main.geometry("220x150") You, 5 days ago • Completed Exercise 3.py

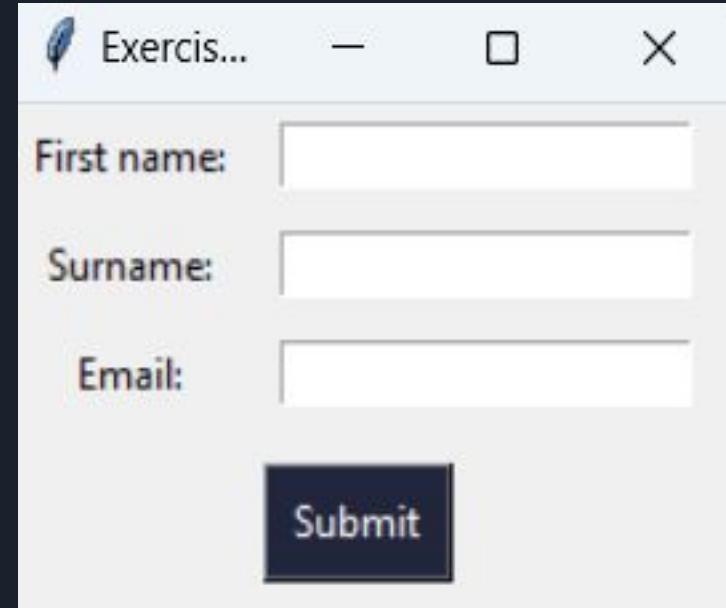
NL = Label(main, text="First name: ")
NE = Entry(main)
NL.grid(row=0, column=0, padx=5, pady=5)
NE.grid(row=0, column=1, padx=5, pady=5)

SL = Label(main, text="Surname: ")
SE = Entry(main)
SL.grid(row=1, column=0, padx=5, pady=5)
SE.grid(row=1, column=1, padx=5, pady=5)

EL = Label(main, text="Email: ")
EE = Entry(main)
EL.grid(row=2, column=0, padx=5, pady=5)
EE.grid(row=2, column=1, padx=5, pady=5)

Submit = Button(main, text="Submit", padx=5, pady=5, bg="#22263d", fg="white")
Submit.grid(columnspan=2, pady=10)

main.mainloop()
```



E4: Registration Page

```
# Exercise 4: Registration Page
# Use widgets to create a GUI

import tkinter
from tkinter import *
from PIL import Image, ImageTk
import os
main = Tk()
main.title("Exercise 4")
Banner = ImageTk.PhotoImage(Image.open("python-Assessments-1-Github/Chapter_11/RAK_Bathspa.jpg"))
Heading = Label(main,image=Banner,bg="#22263d")
Heading.pack(side=TOP,fill=BOTH,expand=NO)
Framehead = Frame(main,bg="#f5f5f5")
Framehead.pack()
Header = Label(Framehead,text="Student Management System",font=("Arial",14,"bold"),fg="#22263d",bg="#f5f5f5",padx=5)
Header2 = Label(Framehead,text="New Student Registration",font=("Arial",12,"bold"),fg="#22263d",bg="#f5f5f5",padx=5)
Header.pack()
Header2.pack()

Framebody = Frame(main,bg="#f5f5f5")
Framebody.pack()
Framecourses = Frame(main,bg="#f5f5f5")
Framecourses.pack()
Courses = Label(Framecourses,text="Course Enrolled:",font=("Arial",10),fg="#22263d",bg="#f5f5f5")
Courses.grid(row=0,column=0)
Courses.grid(row=0,column=0,padx=20)
Variablecourses = {}
for (text,value) in Variablecourses.items():
    Radiobutton(Framecourses, text=text,value=value,font=("Arial",10),fg="#22263d",bg="#f5f5f5").grid(row=0,column=1)
    Radiobutton(Framecourses, text="BSc CC : "+value,font=("Arial",10),fg="#22263d",bg="#f5f5f5").grid(row=0,column=2)
    Radiobutton(Framecourses, text="BSc PSY : "+value,font=("Arial",10),fg="#22263d",bg="#f5f5f5").grid(row=0,column=3)
    Radiobutton(Framecourses, text="BA & BM : "+value,font=("Arial",10),fg="#22263d",bg="#f5f5f5").grid(row=0,column=4)
Radiobutton(Framecourses, text="Other : "+value,font=("Arial",10),fg="#22263d",bg="#f5f5f5").grid(row=0,column=5)
Framecourses.grid(row=1,sticky=W)
Framecourses.pack()

Framelanguage = Frame(main,bg="#f5f5f5")
Framelanguage.pack()
Languages = Label(Framelanguage,text="Languages Known",font=("Arial",10),fg="#22263d",bg="#f5f5f5")
Languages.grid(row=0,column=0)
Buttons = []
for i in range(2):
    English = Checkbutton(Framelanguage, text="English",variable=Buttons[i],font=("Arial",10),fg="#22263d",bg="#f5f5f5")
    English.grid(row=0,column=i)
    Tagalog = Checkbutton(Framelanguage, text="Tagalog",variable=Buttons[i],font=("Arial",10),fg="#22263d",bg="#f5f5f5")
    Tagalog.grid(row=0,column=i)
    Indian = Checkbutton(Framelanguage, text="Hindi/Urdu",variable=Buttons[i],font=("Arial",10),fg="#22263d",bg="#f5f5f5")
    Indian.grid(row=0,column=i)

Framecheckbox = Frame(main,bg="#f5f5f5")
Framecheckbox.pack()
main.mainloop()
```

Exercise 4



Student Management System

New Student Registration

Student Name:

Mobile Number:

Email ID:

Home Address:

Gender: BSc CC BSc CY BSc PSY BA & BM

Course Enrolled: BSc CC BSc CY BSc PSY BA & BM

Languages Known: English Tagalog Hindi/Urdu

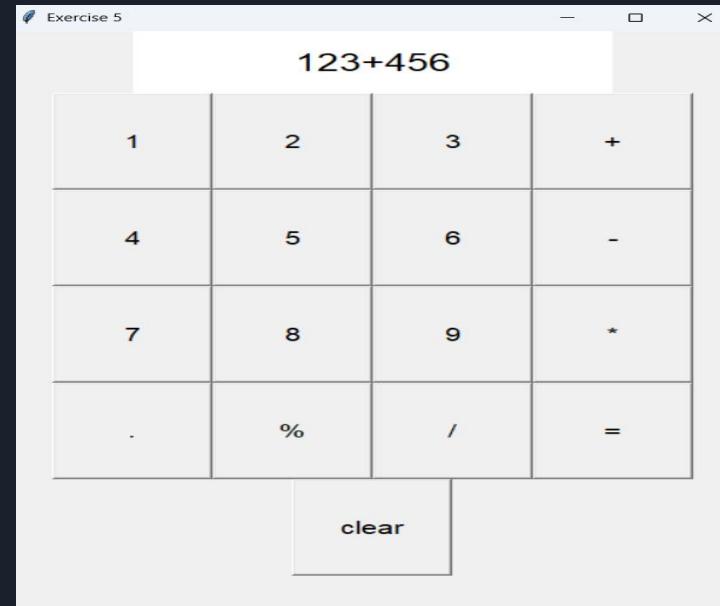
Rate your English communication skills

1

Submit **Clear**

E5: Calculator

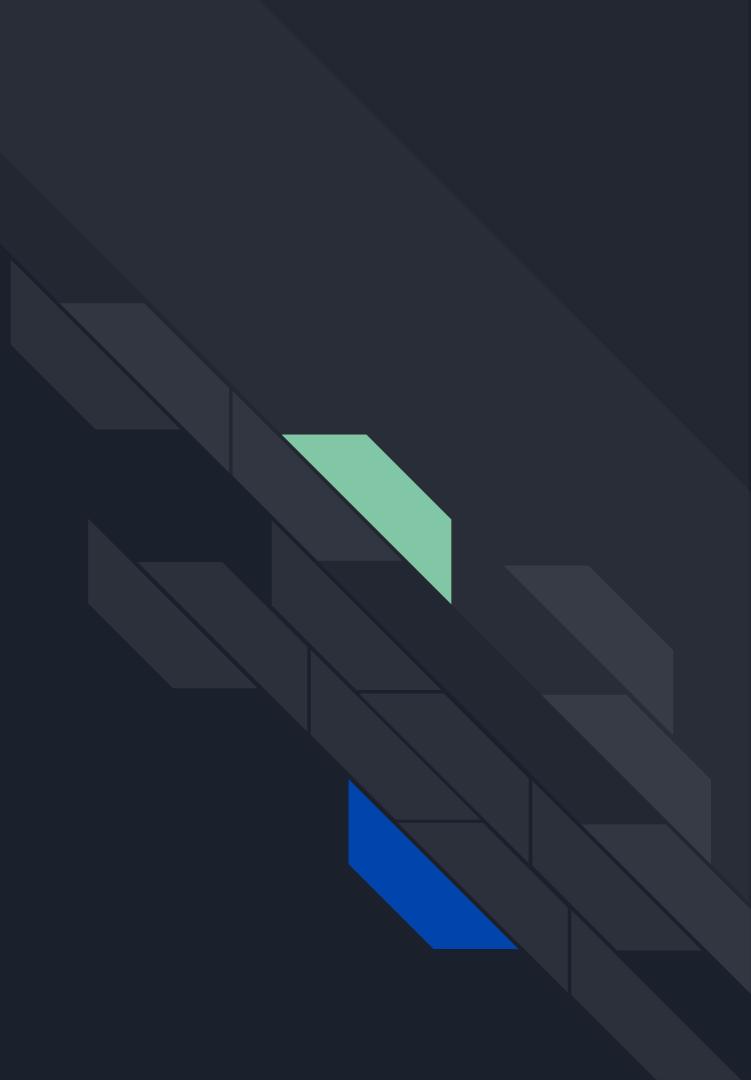
```
import tkinter
from tkinter import *
def button_press(num):
    global texquation
    texquation = texquation + str(num)
    labequation.set(texquation)
def equals():
    try:
        global texquation
        total = str(eval(texquation))
        labequation.set(total)
        texquation = total
    except SyntaxError:
        labequation.set("Syntax Error")
        texquation = ""
button7 = Button(frame, text='7', height=4, width=9, font=35, command=lambda: button_press(7))
button7.grid(row=2, column=0)
button8 = Button(frame, text='8', height=4, width=9, font=35, command=lambda: button_press(8))
button8.grid(row=2, column=1)
button9 = Button(frame, text='9', height=4, width=9, font=35, command=lambda: button_press(9))
button9.grid(row=2, column=2)
button0 = Button(frame, text='0', height=4, width=9, font=35, command=lambda: button_press(0))
button0.grid(row=2, column=3)
plus = Button(frame, text='+', height=4, width=9, font=35, command=lambda: button_press('+'))
plus.grid(row=0, column=3)
minus = Button(frame, text='-', height=4, width=9, font=35, command=lambda: button_press('-'))
minus.grid(row=0, column=1)
multiply = Button(frame, text='*', height=4, width=9, font=35, command=lambda: button_press('*'))
multiply.grid(row=0, column=2)
divide = Button(frame, text='/', height=4, width=9, font=35, command=lambda: button_press('/'))
divide.grid(row=0, column=1)
decimal = Button(frame, text='.', height=4, width=9, font=35, command=lambda: button_press('.'))
decimal.grid(row=3, column=0)
modulate = Button(frame, text='%', height=4, width=9, font=35, command=lambda: button_press('%'))
modulate.grid(row=3, column=1)
equal = Button(frame, text='=', height=4, width=9, font=35, command=equals)
equal.grid(row=3, column=3)
clearing = Button(main, text='clear', height=4, width=9, font=35)
clearing.pack()
main.mainloop()
```



Chapter III

Graphical User Interface

(cont.)



E1: Greeting App

```
# Exercise 1: Greeting App

# Develop a GUI to greet the user. The app should have two frames: InputFrame and DisplayFrame.

import tkinter
from tkinter import *
from tkinter import ttk

main = Tk()

def update_greeting():
    name = name_entry.get()
    color = colorvar.get()
    greeting_label.config(text="Hello " + name + "!", fg=color)

main.title('Exercise 1')
main.geometry('500x500')

Finput = Frame(main, bg='skyblue')
Finput.pack(pady=10)

Fdisplay = Frame(main, bg='goldenrod')
Fdisplay.pack(pady=10)

Label = Label(Finput, text="Greeting App", fg="blue", font=("Helvetica", 20), bg="skyblue")
Label.grid(row=0, column=0, columnspan=2, pady=10)

Label1 = Label(Finput, text="State your name", bg="skyblue")
Label1.grid(row=1, column=0, pady=5)
Label1['text'] = "You, last week + Updating Exercise 1.py"

name_entry = Entry(Finput)
name_entry.grid(row=1, column=1, pady=5)

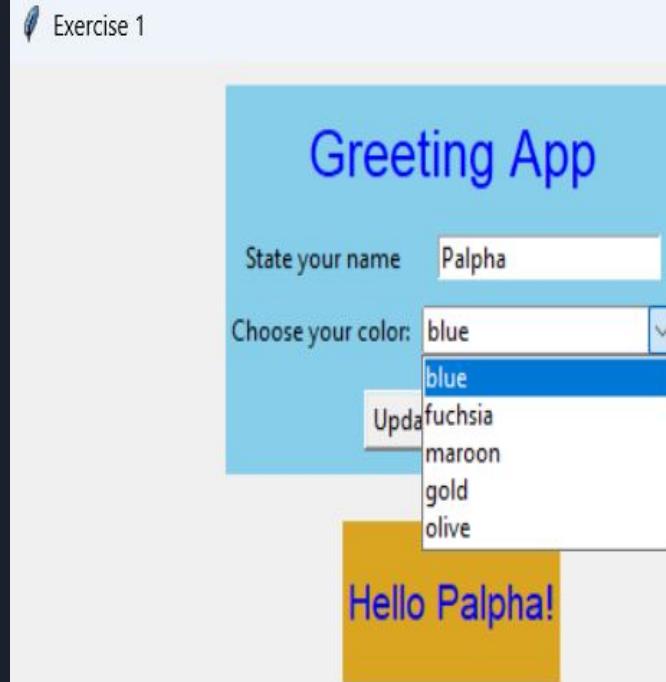
Label2 = Label(Finput, text="Choose your color:", bg="skyblue")
Label2.grid(row=2, column=0, pady=5)
```

```
colors = ['blue', 'fuchsia', 'maroon', 'gold', 'olive']
colorvar = StringVar()
colordropdown = ttk.Combobox(Finput, textvariable=colorvar, values=colors, state="readonly")
colordropdown.current(0)
colordropdown.grid(row=2, column=1, pady=5)

update = Button(Finput, text="Update Greeting", command=update_greeting)
update.grid(row=3, column=0, columnspan=2, pady=10)

greeting_label = Label(Fdisplay, text="", font=("Helvetica", 15), bg='goldenrod')
greeting_label.pack(pady=20)

main.mainloop()
```



E2: Coffee Vending Machine

```
# Exercise 2: Coffee Vending Machine

# Develop a coffee Vending Machine that asks users to select the type of coffee, and also prompts users to select various options like sug
# Extension: Add other features to the previous app such as handling money.

import tkinter
from tkinter import *
from tkinter import messagebox

main = Tk()

main.title('Exercise 2')
main.geometry('500x500')

Greeting = Label(main,text="Welcome to my Coffee Shop!",font=('Helvetica',12,'bold'),bg="#884513",fg="chocolate1",width=25,height=5,pady=5)
Greeting.pack()

coffeetype = ["Cappuccino","Latte","Espresso","Mocha","Americano"]
coffvar = StringVar(main)
coffvar.set(coffeetype[0])

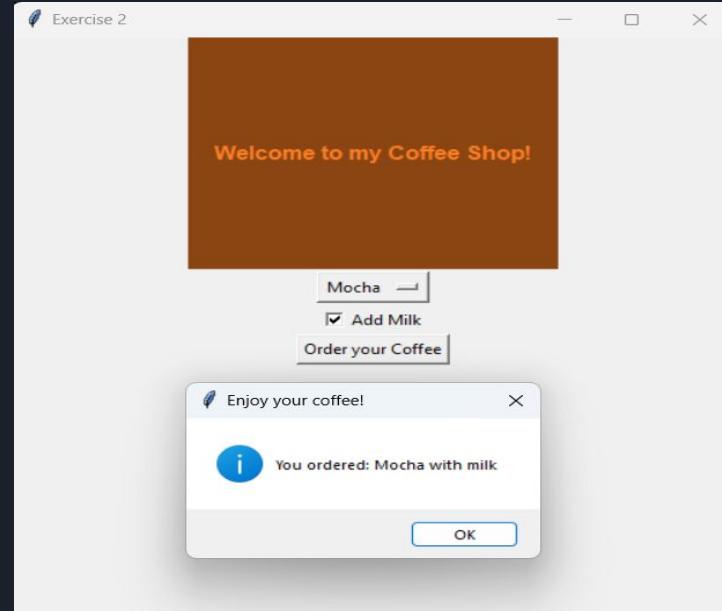
dropdown = OptionMenu(main,coffvar,*coffeetype)
dropdown.pack()

milkar = IntVar(value=0)
checkbox = Checkbutton(main,text="Add Milk",variable=milkar)
checkbox.pack()

def order():
    coffee = coffvar.get()
    havemilk = "with milk" if milkar.get() == 1 else ""
    messagebox.showinfo("Enjoy your coffee!", f"You ordered: {coffee} {havemilk}")

orderup = Button(main,text="Order your coffee",command=order)
orderup.pack()

main.mainloop()
```



E3: Area Function

```
# develop a GUI application using tkinter that allows users to calculate and display the areas of various shapes
# including circles, squares, and rectangles

import tkinter
from tkinter import *
from tkinter import messagebox

def calculate_area():
    try:
        if selected.get() == "Circle":
            radius = float(radiusentry.get())
            area = 3.14 * radius**2
        elif selected.get() == "Square":
            side = float(sideentry.get())
            area = side**2
        elif selected.get() == "Rectangle":
            length = float(lengthentry.get())
            width = float(widthentry.get())
            area = length * width
        else:
            messagebox.showerror("That's not a shape...")
            return
    except ValueError:
        messagebox.showerror("Is that a dimension?")
    resultlabel.config(text=f"Area: {area:.2f} square units")
    resultlabel.pack(pady=10)
    mainloop()

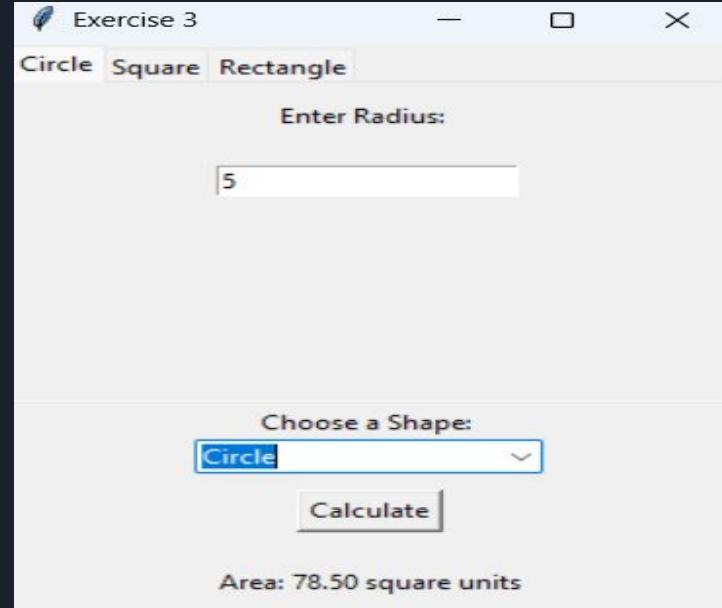
main = Tk()
main.title("Exercise 3")
main.geometry("380x350")
tab = ttk.Notebook(main)

circle = Frame(tab)
square = Frame(tab)
rectangle = Frame(tab)

calculate = Button(main, text="Calculate", command=calculate_area)
calculate.pack(pady=10)

resultlabel = Label(main, text="The Area is: ")
resultlabel.pack(pady=10)

main.mainloop()
```



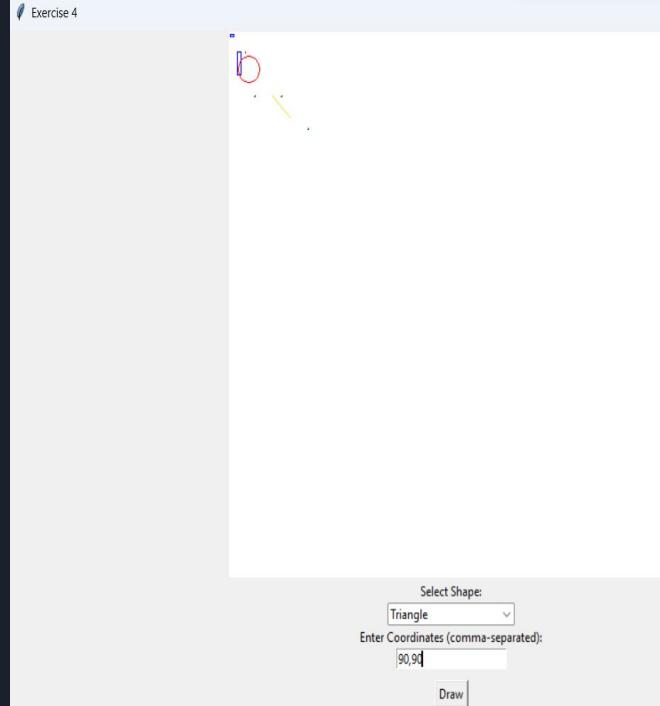
E4: Draw Shapes

```
# Exercise 4: Draw Shape  
  
# Using tkinter module develop GUI to ask the user to select shapes like  
# oval, rectangle, square, triangle  
# and draw the shape using canvas
```

```
import tkinter  
from tkinter import *  
from tkinter import ttk, messagebox  
  
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcey  
def draw_shape():  
    try:  
        selected = shapeselection.get()  
        coordinates = coordinateentry.get()  
        coordinatelist = [int(coord.strip()) for coord in coordinates.split(',')]  
  
        if selected == "Star":  
            canvas.create_polygon(coordinatelist, outline='gold')  
        elif selected == "Square":  
            canvas.create_rectangle(coordinatelist, outline='blue')  
        elif selected == "Circle":  
            canvas.create_oval(coordinatelist, outline='red')  
        elif selected == "Triangle":  
            canvas.create_polygon(coordinatelist, outline='green')  
        else:  
            messagebox.showerror("Nope", "Please choose a valid shape")  
            return  
  
    except ValueError:  
        messagebox.showerror("Where are you going?", "Please enter valid coordinates")  
    main.mainloop()
```

```
main = Tk()  
main.title("Exercise 4")  
main.geometry("1000x1000")  
  
canvas = Canvas(main, width=500, height=500, bg='white')  
canvas.pack()  
You, 8 hours ago • Update Exercise 4.py
```

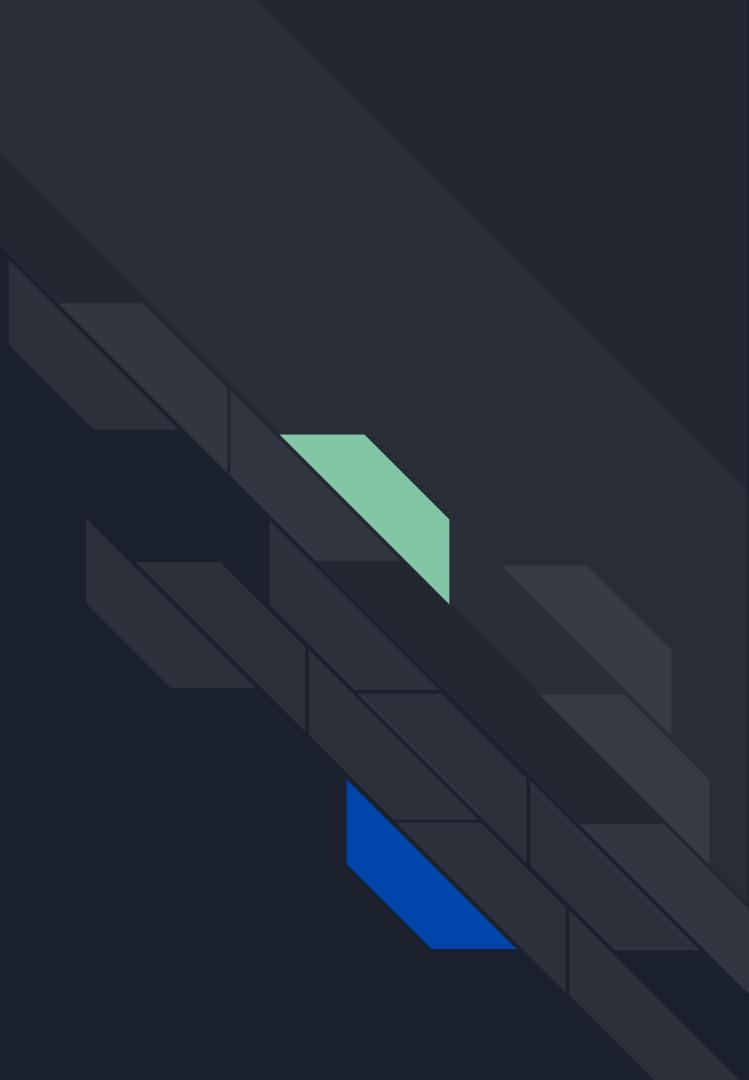
```
shapabel = Label(main, text="Select Shape:")  
shapabel.pack()  
shapeselection = ttk.Combobox(main, values=["Star", "Square", "Circle", "Triangle"])  
shapeselection.pack()  
  
coordinatabel = Label(main, text="Enter Coordinates (comma-separated):")  
coordinatabel.pack()  
coordinateentry = Entry(main)  
coordinateentry.pack()  
  
draw = Button(main, text="Draw", command=draw_shape)  
draw.pack(pady=10)  
return
```



Chapter IV:

File Handling & Regular

Expressions



E1: User Information

Develop a GUI App to create a file called 'bio.txt' and write the following information to the file:
Each piece of data should be on a new line. Once the data has been written to the file, read the file.

```
import tkinter  
from tkinter import *  
from tkinter import messagebox
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
class UserInfoApp:  
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
    def __init__(self, main):  
        self.main = main  
        self.main.title("Exercise 1")
```

```
        self.create_input_fields()  
        self.create_submit_button()
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
def create_input_fields(self):  
    fields = ["Name:", "Age:", "Hometown:"]  
    self.entries = {}  
    for field in fields:  
        label = Label(self.main, text=field + ":").pack()  
        entry = Entry(self.main)  
        entry.pack()  
        self.entries[field] = entry
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
def create_submit_button(self):  
    submit_button = Button(self.main, text="Submit", command=self.write_and_read_info)  
    submit_button.pack()
```

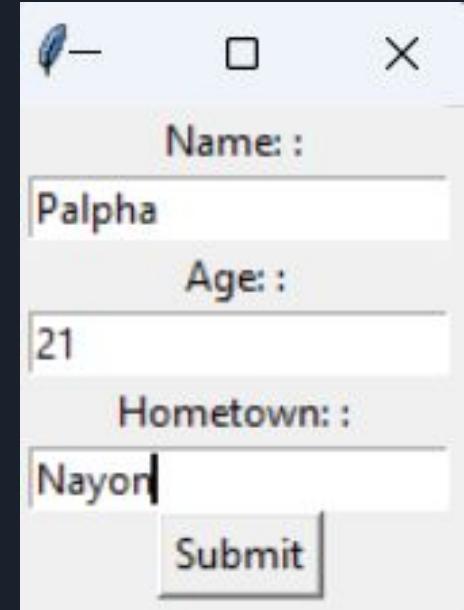
```
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def create_submit_button(self):  
    submit_button = Button(self.main, text="Submit", command=self.write_and_read_info)  
    submit_button.pack()
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
def write_and_read_info(self):  
    user_info = ""  
    for field, entry in self.entries.items():  
        value = entry.get()  
        user_info += f"{field}: {value}"  
  
    with open("Python-Assessments-1-GitHub/Chapter IV/Texts/Bio.txt", 'w') as file:  
        file.write(user_info)  
  
    with open("Python-Assessments-1-GitHub/Chapter IV/Texts/Bio.txt", 'r') as file:  
        user_info = file.read()  
        messagebox.showinfo("User Information", user_info)
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
if __name__ == "__main__":  
    root = Tk()  
    app = UserInfoApp(root)  
    root.mainloop()
```



E2: Countstrings

```
# The file sentences.txt has a list of string data.  
# Develop a GUI app that finds out how many times the following string appears  
  
# Hello my name is Peter Parker  
# I love Python Programming  
# Love  
# Enemy  
  
import tkinter  
from tkinter import *  
You, 2 days ago • IV's exercise 2 is weird  
  
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def count_string_occurrences():  
    target_strings = [  
        "Hello my name is Peter Parker",  
        "I love Python Programming",  
        "Love",  
        "Enemy"  
    ]  
  
    with open("Python-Assessments-1-GitHub/Chapter IV/Texts/Sentences.txt", 'r') as file:  
        data = file.read()  
  
    occurrences = (data.count(string) for string in target_strings)  
    result.config(text="Occurrences: {occurrences}")  
  
    main.mainloop()  
  
main = Tk()  
main.title("Exercise 2")  
  
instructions = Label(main, text="Count the string occurrences in the file")  
instructions.pack()  
  
count = Button(main, text="Count Occurrences", command=count_string_occurrences)  
count.pack()  
  
result = Label(main, text="Occurrences: ")  
  
result.config(text="Occurrences: {occurrences}")
```



E3: Reading to a List

```
# Exercise 3: Reading to a List

# The file numbers.txt has a list of 100 integer numbers each on a newline
# Create a python program that puts this data into a list, then output the values in integer format
```

[Explain Code](#) | [Generate Tests](#) | [Generate Docstrings](#) | [Ask Sourcery](#)

```
def read_numbers(data_path):
    try:
        with open("Python-Assessments-1-GitHub/Chapter IV/Texts/Numbers.txt", "r") as file:
            noombers = [int(line.strip()) for line in file]
        return noombers
    except FileNotFoundError:
        print(f"Error: File '{data_path}' not found.")
        return []
    except ValueError as e:
        print(f"Error: {e}. the file '{data_path}' contains non-integer values.")

if __name__ == "__main__":
    data_path = "Python-Assessments-1-GitHub/Chapter IV/Texts/Numbers.txt"

    noomberlist = read_numbers(data_path)

    for noombers in noomberlist:
        print(noombers)
```



E4: Letter Count

```
# Develop a GUI App that asks the user to enter a character  
# reads the contents of the sentences.txt file  
# and counts the occurrences of the letter.
```

```
import tkinter  
from tkinter import *  
from tkinter import messagebox
```

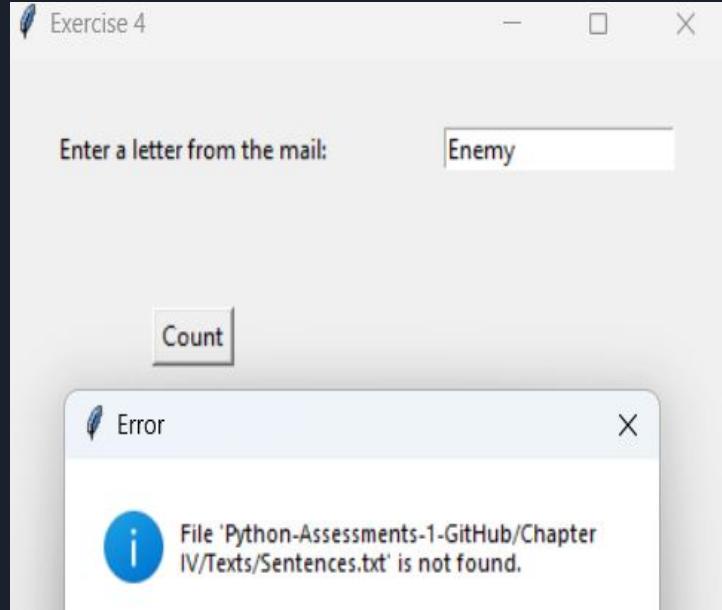
```
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def counting_letter_occurrences(letter,content):  
    return content.lower().count(letter.lower())
```

```
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def read_file(mail_path):  
    try:  
        with open(mail_path,'r') as file:  
            return file.read()  
    except FileNotFoundError:  
        return None
```

```
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def show_letter_count():  
    letter = mail.get()  
    if not letter:  
        messagebox.showinfo("Error", "Where is the letter for your mail?")  
        return
```

```
file_content = read_file('Python-Assessments-1-GitHub/Chapter IV/Texts/Sentences.txt') app.mainloop()
```

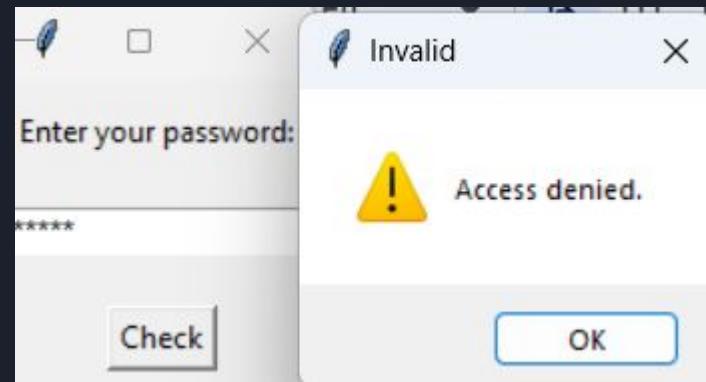
```
if file_content is None:  
    messagebox.showinfo("Error", "File 'Python-Assessments-1-GitHub/Chapter IV/Texts/Sentences.txt'  
occurrences = counting_letter_occurrences(letter,file_content)  
resultlabel.config(text=f"Occurrences of '{letter}': {occurrences}")  
  
app = Tk()  
app.title("Exercise 4")  
  
mailabel = Label(app,text="Enter a letter from the mail:")  
mail = Entry(app)  
submiton = Button(app,text="Count",command=show_letter_count)  
resultlabel = Label(app,text="")  
  
mailabel.grid(row=0,column=0,padx=30,pady=30)  
mail.grid(row=0,column=1,padx=30,pady=30)  
submiton.grid(row=1,column=0,padx=30,pady=30)  
resultlabel.grid(row=2,column=0,padx=30,pady=30)  
  
app.mainloop()
```



E5: Password Check

```
# The password should satisfy the following criteria:  
  
# Contain at least 1 letter between a and z  
# Contain at least 1 number between 0 and 9  
# Contain at least 1 letter between A and Z  
# Contain at least 1 character from $, #, @  
# Minimum length of password: 6  
# Maximum length of password: 12  
  
# Ask user to include a maximum of 5 passcode attempts  
# Each time the user enters an incorrect passcode, they  
# If there are 5 failed passcode attempts the while loop  
  
import tkinter  
from tkinter import *  
from tkinter import messagebox  
import re  
  
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def is_valid_password(password):  
    if (6<=len(password)<=12 and  
        re.search("[a-z]",password)and  
        re.search("[0-9]",password)and  
        re.search("[A-Z]",password)and  
        re.search("[${#@}]",password)):  
            return True  
    return False
```

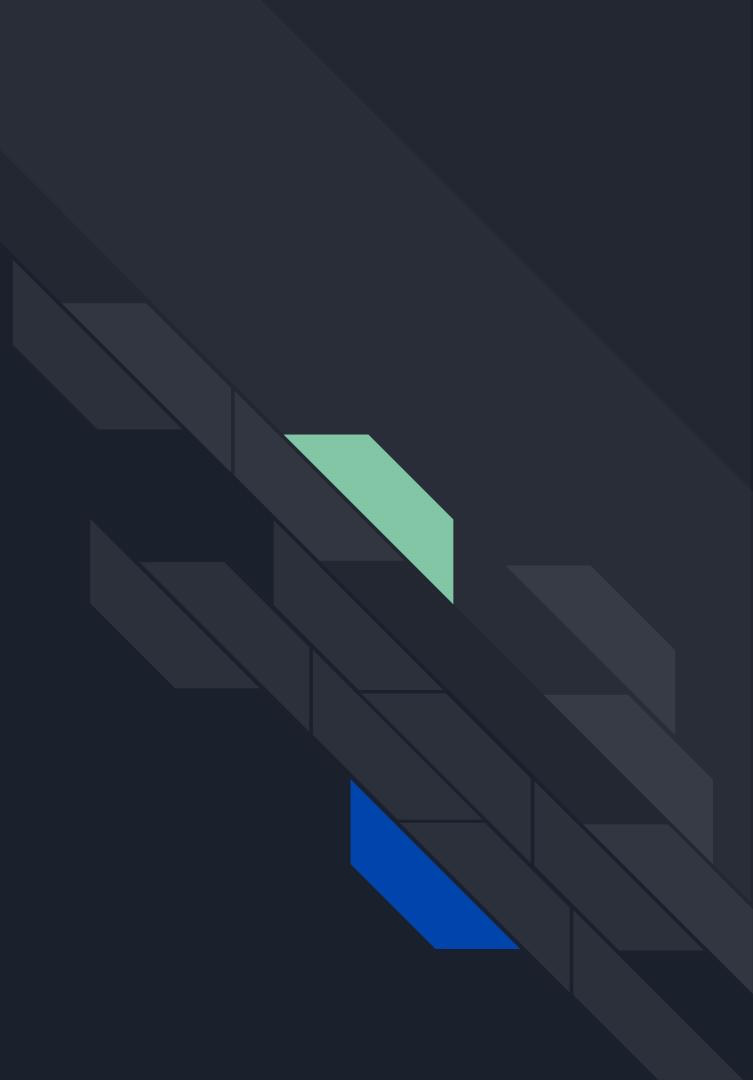
```
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
def check_password():  
    entered_password = pentry.get()  
  
    if is_valid_password(entered_password):  
        messagebox.showinfo("Valid","Access granted!")  
    else:  
        messagebox.showwarning("Invalid","Access denied.")  
  
    app = Tk()  
    app.title("Exercise 5")  
  
    pabel = Label(app,text="Enter your password: ")  
    pentry = Entry(app,show="*")  
    checkon = Button(app,text="Check",command=check_password)  
  
    pabel.pack(pady=10)  
    pentry.pack(pady=10)  
    checkon.pack(pady=10)  
  
    app.mainloop()
```



Chapter V:

Object Oriented

Programming



E1: Woof Woof

```
# Exercise 1: Woof Woof
# Develop a GUI using Tkinter with a class that de

import tkinter as tk

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery | You, 1 sec
class Dog:
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def __init__(Palpha, name, age):
        Palpha.name = name
        Palpha.age = age

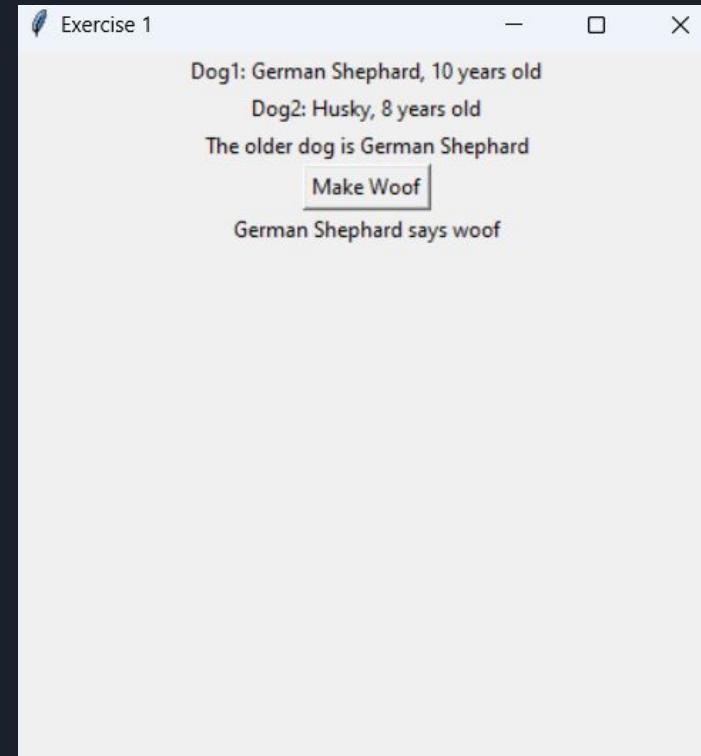
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def woof(Palpha):
        return f"{Palpha.name} says woof"

dog1 = Dog("German Shepherd", 10)
dog2 = Dog("Husky", 8)
older_dog = dog1 if dog1.age > dog2.age else dog2

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery | You, 1 sec
class DogGUI(tk.Tk):
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def __init__(Palpha):
        super().__init__()
        Palpha.title(["Exercise 1"])
        Palpha.geometry("400x400")
        Palpha.display_dog_info()
```

```
def display_dog_info(Palpha):
    label1 = tk.Label(Palpha, text=f"Dog1: {dog1.name}, {dog1.age} years old")
    label1.pack()
    label2 = tk.Label(Palpha, text=f"Dog2: {dog2.name}, {dog2.age} years old")
    label2.pack()
    older_dog_label = tk.Label(Palpha, text=f"The older dog is {older_dog.name}")
    older_dog_label.pack()
    woof_button = tk.Button(Palpha, text="Make Woof", command=Palpha.make_woof)
    woof_button.pack()

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def make_woof(Palpha):
    result = older_dog.woof()
    woof_label = tk.Label(Palpha, text=result)
    woof_label.pack()
```



E2: Student Class

```
import tkinter as tk  
from tkinter import messagebox
```

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery | You, yesterday | 1 author (You)

class Student:

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
def __init__(self, name, mark1, mark2, mark3):
```

self.name = name

self.mark1 = mark1

self.mark2 = mark2

self.mark3 = mark3

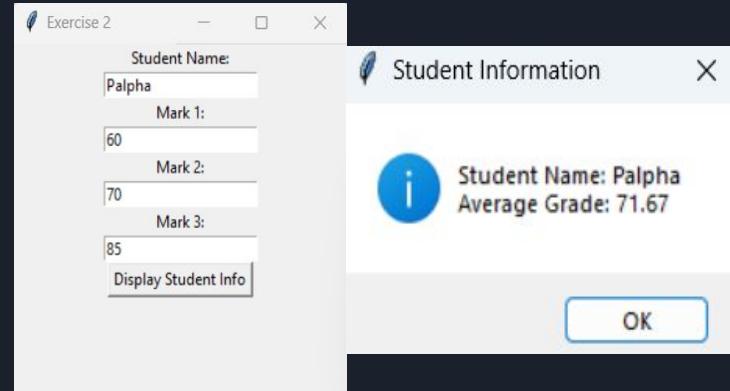
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

```
def calc_grade(self):
```

average = (self.mark1 + self.mark2 + self.mark3) / 3

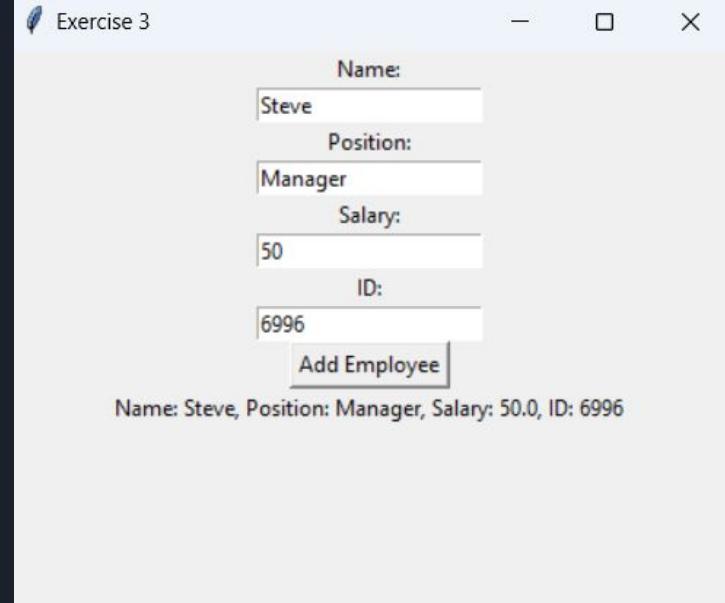
return round(average, 2)

```
def calc_grade(self):  
    average = (self.mark1 + self.mark2 + self.mark3) / 3  
    return round(average, 2)  
  
class StudentGUI:  
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
    def __init__(self, Palpha):  
        Palpha.name_entry = tk.Entry(score)  
        Palpha.name_entry.pack()  
  
        Palpha.score.title("Exercise 2")  
  
        tk.Label(score, text="Student Name:").pack()  
        Palpha.name_entry = tk.Entry(score)  
        Palpha.name_entry.pack()  
  
        tk.Label(score, text="Mark 1:").pack()  
        Palpha.mark1_entry = tk.Entry(score)  
        Palpha.mark1_entry.pack()  
  
        tk.Label(score, text="Mark 2:").pack()  
        Palpha.mark2_entry = tk.Entry(score)  
        Palpha.mark2_entry.pack()  
  
        tk.Label(score, text="Mark 3:").pack()  
        Palpha.mark3_entry = tk.Entry(score)  
        Palpha.mark3_entry.pack()  
  
        Palpha.display_button = tk.Button(score, text="Display Student Info", command=Palpha.display_stu  
        Palpha.display_button.pack()  
  
    explain Code | Generate Tests | Generate Docstrings | Ask Sourcery  
    def display_student_info(Palpha):  
        name = Palpha.name_entry.get()  
        mark1 = int(Palpha.mark1_entry.get())  
        mark2 = int(Palpha.mark2_entry.get())  
        mark3 = int(Palpha.mark3_entry.get())  
  
        if name and mark1 and mark2 and mark3:  
            student = Student(name, mark1, mark2, mark3)  
            average_grade = student.calc_grade()  
            message = f"Student Name: {student.name}\nAverage Grade: {average_grade}"  
            messagebox.showinfo("Student Information", message)  
        else:  
            messagebox.showerror("Error", "Please fill in all the fields.")  
  
    if __name__ == "__main__":  
        main = tk.Tk()  
        app = StudentGUI(main)  
        main.mainloop()
```



E3: Employee Class

```
import tkinter as tk
from tkinter import *
Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
class Employee:
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def __init__(self):
        Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
        self.name = ""
        self.position = ""
        self.salary = 0.0
        self.id = ""
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def set_data(self, name, position, salary, id):
        Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
        self.name = name
        self.position = position
        self.salary = salary
        self.id = id
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def get_data(self):
        return f"Name: {self.name}, Position: {self.position}, Salary: {self.salary}, ID: {self.id}"
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def create_display_button(self):
        Palpha.display_button = tk.Button(Palpha, text="Add Employee", command=Palpha.add_employee)
        Palpha.display_button.pack()
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def add_employee(self):
        name = Palpha.name_entry.get()
        position = Palpha.position_entry.get()
        salary = float(Palpha.salary_entry.get())
        id = Palpha.id_entry.get()
        employee = Employee()
        employee.set_data(name, position, salary, id)
        Palpha.employees.append(employee)
        Palpha.display_employees()
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
    def display_employees(self):
        Palpha.display_frame = tk.Frame(Palpha)
        Palpha.display_frame.pack()
        for employee in Palpha.employees:
            employee_label = tk.Label(Palpha.display_frame, text=employee.get_data())
            employee_label.pack()
app = EmployeeGUI()
app.mainloop()
```



E4: Shapes

```
import tkinter as tk
import math

Explain Code | Generate Tests | Generate Docstrings | Ask Source
You, 2 days ago | author (You)
class Shape:
    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def __init__(self):
        You, 2 days ago + Finished the cursed shapes
        | Palphashaper = shape
        | Palphashaper.pack()

    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def input_sides(self):
        Palphashaderlabel = tk.Label(Palphashaper, text="Enter sides:")
        Palphashaderlabel.pack()

    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def input_radius(self):
        Palpharadiolabel = tk.Label(Palphashaper, text="Enter radius:")
        Palpharadiolabel.pack()

    main = tk.TK()
    main.title("Exercise 4")

    circleframe = tk.Frame(main)
    circleframe.pack(side=tk.LEFT)
    circletlabel = tk.Label(circleframe, text="Circle")
    circletlabel.pack()
    circle = Circle(circleframe)

    rectangleframe = tk.Frame(main)
    rectangleframe.pack(side=tk.LEFT)
    rectanglelabel = tk.Label(rectangleframe, text="Rectangle")
    rectanglelabel.pack()
    rectangle = Rectangle(rectangleframe)

    triangleframe = tk.Frame(main)
    triangleframe.pack(side=tk.LEFT)
    trianglalabel = tk.Label(triangleframe, text="Triangle")
    trianglalabel.pack()
    triangle = Triangle(triangleframe)

main.mainloop()
```

```
class Circle(Shape):
    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def __init__(self):
        super().__init__(shape)
        Palphashaderlabel = tk.Entry(Palphashaper)
        Palphashaderlabel.get()
        Palphacalculatery = tk.Entry(Palphashaper)
        Palphacalculatery.get()
        Palphacalculatebutton = tk.Button(Palphashaper, text="Calculate Area", command=Palphacalculate_area)
        Palphacalculatebutton.pack()

    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def calculate_area(Palphashaderlabel):
        radius = float(Palphashaderlabel.get())
        area = 3.14 * radius * radius
        resultlabel = tk.Label(Palphashaper, text=f"Area of the circle: {area}")
        resultlabel.pack()

    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def calculate_area(Palphashaderlabel):
        base = float(Palphashaderlabel.get())
        height = float(Palphashaderlabel.get())
        area = 0.5 * base * height
        resultlabel = tk.Label(Palphashaper, text=f"Area of the rectangle: {area}")
        resultlabel.pack()

    Explain Code | Generate Tests | Generate Docstrings | Ask Source
    def calculate_area(Palphashaderlabel):
        base = float(Palphashaderlabel.get())
        width = float(Palphashaderlabel.get())
        area = base * width
        resultlabel = tk.Label(Palphashaper, text=f"Area of the triangle: {area}")
        resultlabel.pack()
```

Exercise 4

Circle	Rectangle	Triangle
Enter radius:	Enter sides:	Enter sides:
<input type="text" value="3"/>	<input type="text" value="4"/>	<input type="text" value="3"/>
Calculate Area	Calculate Area	Calculate Area
Area of the circle: 28.27433882308138		
Area of the rectangle: 16.0 Area of the triangle: 4.5		

E5: Playing around in class

Exercise 5: Playing around in class

```
# Use this exercise to play around with creating and accessing class members and methods.
# Develop a GUI using Tkinter to Create a class called Animal

import tkinter as tk

class Animal:
    def __init__(self, animal_type, name, color, age, weight, noise):
        self.animal_type = animal_type
        self.name = name
        self.color = color
        self.age = age
        self.weight = weight
        self.noise = noise

    def say_hello(self):
        return f'Hey guys! Here is {self.name}, a {self.color} {self.animal_type}!'

    def make_noise(self):
        return f'{self.name} says: {self.noise}'

class AnimalUI(tk.Tk):
    def __init__(self):
        super().__init__()
        self.title("Exercise 5")
        self.geometry("400x400")
        self.create_widgets()

    def create_widgets(self):
        labels = ["Animal Type", "Name", "Color", "Age", "Weight", "Noise"]
        entries = []

        for label in labels:
            new_label = tk.Label(self, text=label + ":")
            new_label.pack()
            new_entry = tk.Entry(self)
            new_entry.pack()
            entries.append(new_entry)

        button_create = tk.Button(self, text="Create Animal", command=lambda: self.create_animal(*[entry.get() for entry in entries]))
        button_create.pack()

        Palpha.result_label = tk.Label(self, text="")
        Palpha.result_label.pack()

    def create_animal(self, animal_type, name, color, age, weight, noise):
        try:
            age = int(age)
            weight = float(weight)
            animal = Animal(animal_type, name, color, age, weight, noise)

            result_text = animal.say_hello() + "\n" + animal.make_noise()
            Palpha.result_label.config(text=result_text)
        except ValueError:
            Palpha.result_label.config(text="Error. Please enter a valid age and weight")

    def mainloop(self):
        app = AnimalUI()
        app.mainloop()
```

```
def create_widgets(Palpha):
    labels = ["Animal Type", "Name", "Color", "Age", "Weight", "Noise"]
    entries = []

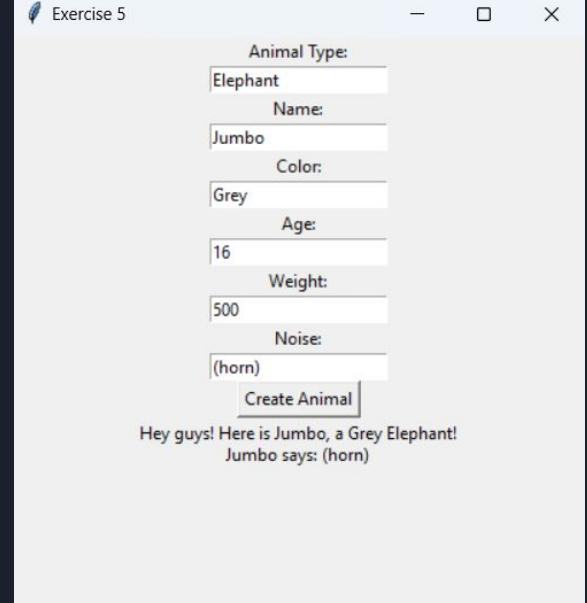
    for label in labels:
        new_label = tk.Label(Palpha, text=label + ":")
        new_label.pack()
        new_entry = tk.Entry(Palpha)
        new_entry.pack()
        entries.append(new_entry)

    button_create = tk.Button(Palpha, text="Create Animal", command=lambda: Palpha.create_animal(*[entry.get() for entry in entries]))
    button_create.pack()

    Palpha.result_label = tk.Label(Palpha, text="")
    Palpha.result_label.pack()

    def create_animal(Palpha, animal_type, name, color, age, weight, noise):
        try:
            age = int(age)
            weight = float(weight)
            animal = Animal(animal_type, name, color, age, weight, noise)

            result_text = animal.say_hello() + "\n" + animal.make_noise()
            Palpha.result_label.config(text=result_text)
        except ValueError:
            Palpha.result_label.config(text="Error. Please enter a valid age and weight")
```



E6: Arithmetic Operations

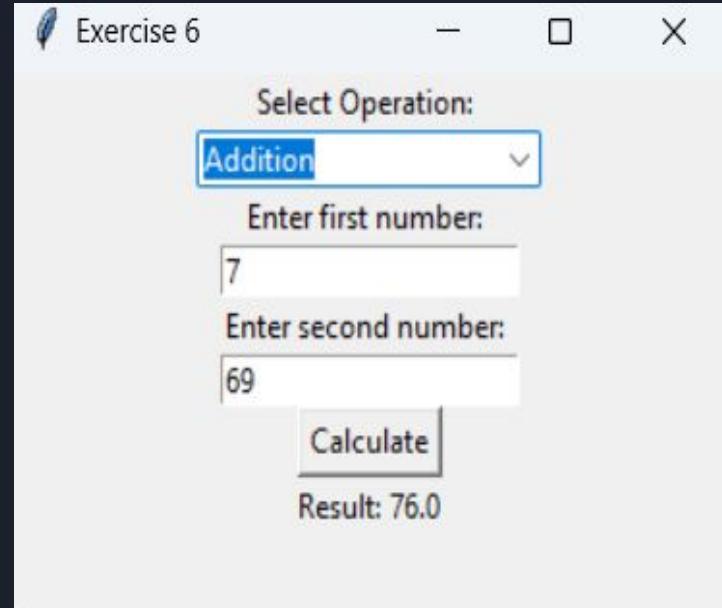
```
# a function calculate() - To perform an arithmetic operation selected by the user
# You can use Combobox to provide users with options to perform selected arithmetic operations
You, 17 hours ago • Updating Chapter V
import tkinter
from tkinter import *
from tkinter import ttk

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcey | You, 2 days ago | 1 author | You!
class ArithmeticOperations(tk):
    Explain Code | Generate Tests | Generate Docstrings | Ask Sourcey
    def __init__(self):
        super().__init__()
        self.title("Exercise 6")
        self.geometry("300x200")
        self.result = 0
        self.create_widgets()

    def create_widgets(self):
        Palpha.operation_label = Label(Palpha, text="Select Operation: ")
        Palpha.operation_label.pack()
        Palpha.operations = ttk.Combobox(Palpha, values=["Addition", "Subtraction", "Multiplication", "Division"])
        Palpha.operations.pack()
        Palpha.num1_label = Label(Palpha, text="Enter first number: ")
        Palpha.num1_label.pack()
        Palpha.num1_entry = Entry(Palpha)
        Palpha.num1_entry.pack()
        Palpha.num2_label = Label(Palpha, text="Enter second number: ")
        Palpha.num2_label.pack()
        Palpha.num2_entry = Entry(Palpha)
        Palpha.num2_entry.pack()
        Palpha.calculate_button = Button(Palpha, text="Calculate", command=Palpha.calculate)
        Palpha.calculate_button.pack()
        Palpha.result_label = Label(Palpha, text="")
        Palpha.result_label.pack()
        self.calculate(Palpha)

    def calculate(self):
        operation = Palpha.operations.get()
        num1 = float(Palpha.num1_entry.get())
        num2 = float(Palpha.num2_entry.get())
        if operation == "Addition":
            Palpha.result = num1 + num2
        elif operation == "Subtraction":
            Palpha.result = num1 - num2
        elif operation == "Multiplication":
            Palpha.result = num1 * num2
        elif operation == "Division":
            if num2 != 0:
                Palpha.result = num1 / num2
            else:
                Palpha.result = "Undefined (Division by zero)"
        Palpha.result_label.config(text=f"Result: {Palpha.result}")

app = ArithmeticOperations()
app.mainloop()
```



Chapter VI: **Python Standard Library**

E1: Basic Math

```
# Exercise 1: Basic Math      You, last week
# Using math module compute the following

# For a =2.3 find the ceil of a
# For a =2.3 find the floor of a
# For a = 5 find the factorial of a
# Find the value of 2^3
# For a=16 find the square root of a

import math

# 1: Ceil
a = 2.3
ceil = math.ceil(a)
print(ceil)

# 2: floor
floor = math.floor(a)
print(floor)

# 3: factorial
a = 5
factorial = math.factorial(a)
print(factorial)

# 4: Value
a = 2**3
print(a)

# 5: SquareRoot
a = 16
sqrt = math.sqrt(a)
print(a)
```

```
3
2
120
8
16
```

E2: Numpy Array

```
import math
import numpy as np

a = [20,23,82,40,32,15,67,52]

# Indicing
print("Before: " + str(a))

odd_i = []
even_i = []
for i in range(0, len(a)):
    if i % 2:
        even_i.append(a[i])
    else:
        odd_i.append(a[i])

a = odd_i + even_i

print("After: " + str(a))

# Sorting
array = np.array(a)

print(np.sort(array))

# slicing 1
s1 = a[3:]

print(s1)

# slicing 2
s2 = a[0:5]

print(s2)

# negative slicing
negs = a[4:7]

print(negs)
```

```
Before: [20, 23, 82, 40, 32, 15, 67, 52]
After: [20, 82, 32, 67, 23, 40, 15, 52]
[15 20 23 32 40 52 67 82]
[67, 23, 40, 15, 52]
[20, 82, 32, 67, 23]
[23, 40, 15]
```

E3: Calculator

```
# Exercise 3: Calculator      You, last week • Updating
# Write a program that will display the calculator menu

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def add(a, b):
|   return a + b

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def subtract(a, b):
|   return a - b

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def multiply(a, b):
|   return a * b

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def divide(a, b):
|   return a / b

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def modulus(a, b):
|   return a % b

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def greatest(a, b):
|   if a>b:
|   |   return a
|   else:
|   |   return b

print("Choose your operation")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
print("5. Modulate")
print("6. Greater than")

while True:
    choice = input("selecting choice(1/2/3/4/5/6): ")

    if choice in ('1','2','3','4','5','6'):
        try:
            a = float(input("Enter your first number: "))
            b = float(input("Enter your second number: "))
        except ValueError:
            print("Sorry, wrong input")
            continue

        if choice == '1':
            |   print(a, "+", b, "=", add(a, b))
        if choice == '2':
            |   print(a, "-", b, "=", subtract(a, b))
        if choice == '3':
            |   print(a, "*", b, "=", multiply(a, b))
        if choice == '4':
            |   print(a, "/", b, "=", divide(a, b))
        if choice == '5':
            |   print(a, "%", b, "=", modulus(a, b))
        if choice == '6':
            |   print("The greatest number is ",greatest(a, b))

    break
```

Choose your operation

1. Add
2. Subtract
3. Multiply
4. Divide
5. Modulate
6. Greater than

Selecting choice(1/2/3/4/5/6): 6

Enter your first number: 69

Enter your second number: 96

The greatest number is 96.0

E4: Line Graph

Exercise 4: Line graph You, last week • Updating Exercise 4.py

```
# Draw a line in a diagram from position (1, 2) to position (6, 8)
# Draw a dotted line in a diagram from position (1, 3) to (2, 8) then to (6, 1) and finally to position (8, 10)

import matplotlib.pyplot as plt

x = [1, 6]
y = [2, 8]

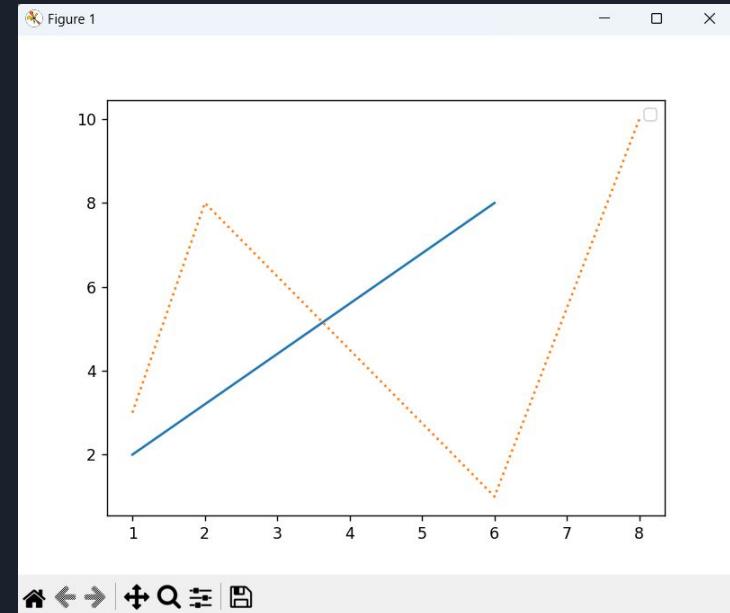
xdot = [1, 2, 6, 8]
ydot = [3, 8, 1, 10]

plt.plot(x, y)

plt.plot(xdot, ydot, linestyle='dotted')

plt.legend()

plt.show()
```



E5: Working with JSON File

```
# Exercise 5: Working with JSON File      You, last week • JSON is weird?

# Create a JSON file named 'StudentJson.json' with the following details

import json

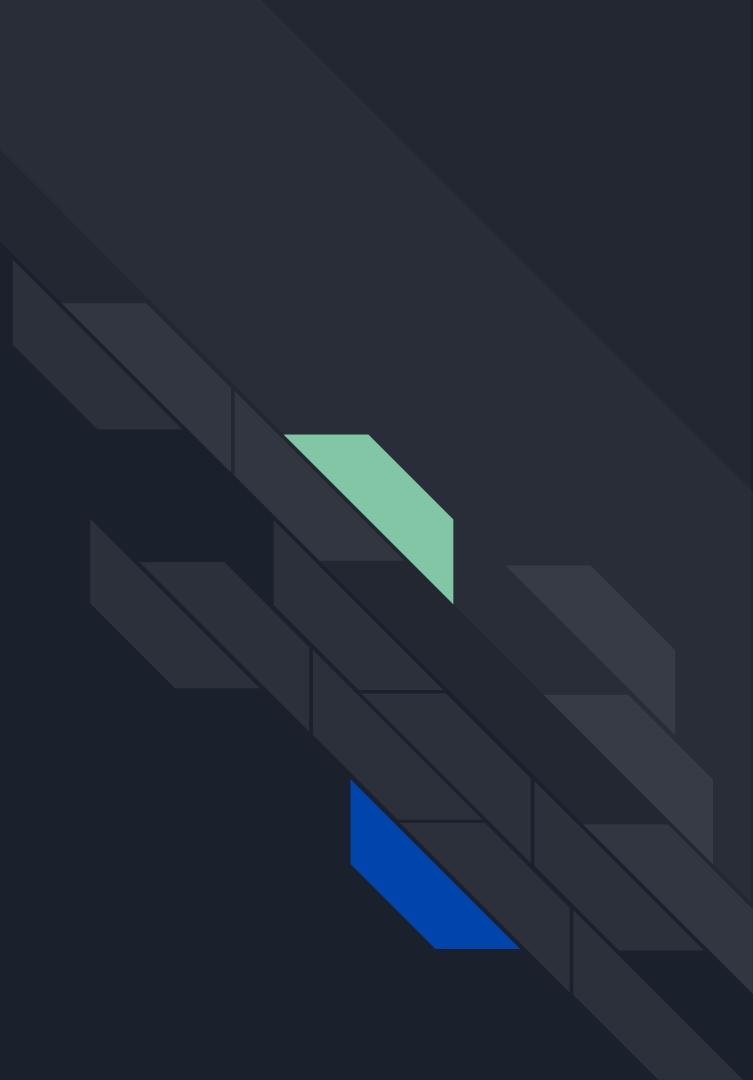
# 1. Ask the user to enter the student name, ID, and course and write these contents to the JSON file
# 2. Read the contents from the JSON file and display the individual values
# 3. Append another dictionary as follows as key value pair for student 1 in StudentDetails dictionary
# 4. Print the individual values of the Student details reading from JSON file

json_string = '''
{
    "Details of the student are": [
        {
            "Name": "Alpha",
            "ID": "1",
            "Course": "BSc CC"
        },
        {
            "Group": "A",
            "Year": 2
        }
    ]
}

data = json.loads(json_string)
print(data)
```

```
{'Details of the student are': [{'Name': 'Alpha', 'ID': '1', 'Course': 'BSc CC'}, {'Group': 'A', 'Year': 2}]} □
```

Bonuses



Bonus: Multiplication Table

```
# Bonus A: Multiplication Tables

# Write a program to print Multiplication tables
# Hint: Use nested loops

# Multiplication table of 1
print ("Multiplication table of 1")
number = int(input("Enter an integer: "))

for count in range (1, 11):
    product = number * count
    print(number, "*", count, "=", product)

# Multiplication table of 10
print ("Multiplication table of 10")
number = int(input("Enter an integer: "))

for count in range (1, 11):
    product = number * count
    print(number, "*", count, "=", product)
```

Multiplication table of 1	Multiplication table of 10
Enter an integer: 4	Enter an integer: 5
4 * 1 = 4	5 * 1 = 5
4 * 2 = 8	5 * 2 = 10
4 * 3 = 12	5 * 3 = 15
4 * 4 = 16	5 * 4 = 20
4 * 5 = 20	5 * 5 = 25
4 * 6 = 24	5 * 6 = 30
4 * 7 = 28	5 * 7 = 35
4 * 8 = 32	5 * 8 = 40
4 * 9 = 36	5 * 9 = 45
4 * 10 = 40	5 * 10 = 50

Bonus: Locations List

```
# Exercise 0: Locations List

# Using the list:
locations = ['Dubai', 'Paris', 'Switzerland', 'London', 'Amsterdam', 'New York']

# Note: I'm pretty sure you could've used 'Bern' instead of 'Switzerland'

# Print the list and find the length of the list
print(locations)
l = len(locations) # You, last week • Submitting my exercises so far

print ("The length of your tour lists is: ", l)

# Use sorted() to print your list in alphabetical order without modifying the actual list.
s = sorted(locations)
print(s)

# Show that your list is still in its original order by printing it.
print(locations)

# Use sorted() to print your list in reverse alphabetical order without changing the order of the original list
print(sorted(locations, reverse=True))

# Show that your list is still in its original order by printing it again.
print(locations)

# Use reverse() to change the order of your list.
locations.reverse()

# Print the list to show that its order has changed.
print(locations)

# Use sort() to change your list so it's stored in alphabetical order.
locations.sort()

# Print the list to show that its order has been changed.
print(locations)

# Use sort() to change your list so it's stored in reverse alphabetical order.
locations.sort(reverse=True)

# Print the list to show that its order has changed.
print(locations)

# Use reverse() to change the order of your list.
locations.reverse()

# Print the list to show that its order has been changed.
print(locations)
```

The length of your tour lists is: 6

```
['Amsterdam', 'Dubai', 'London', 'New York', 'Paris', 'Switzerland']
['Dubai', 'Paris', 'Switzerland', 'London', 'Amsterdam', 'New York']
['Switzerland', 'Paris', 'New York', 'London', 'Dubai', 'Amsterdam']
['Dubai', 'Paris', 'Switzerland', 'London', 'Amsterdam', 'New York']
['New York', 'Amsterdam', 'London', 'Switzerland', 'Paris', 'Dubai']
['Amsterdam', 'Dubai', 'London', 'New York', 'Paris', 'Switzerland']
['Switzerland', 'Paris', 'New York', 'London', 'Dubai', 'Amsterdam']
```

Bonus: Locations List

```
# Exercise C: Calculator Function      You, last week • Subm
# The program should display the following calculator menu:
# Add +, Subtract -, Multiply *, Divide /, Modulus %

# This adds two numbers

def add(a, b):
    return a + b

# This subtracts two numbers

def subtract(a, b):
    return a - b

# This multiplies two numbers

def multiply(a, b):
    return a * b

# This divides two numbers

def divide(a, b):
    return a / b

# This modulates two numbers

def modulus(a, b):
    return a % b
```

```
print("Choose your operation")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
print("5. Modulate")

while True:
    choice = input("Selecting choice(1/2/3/4/5): ")

    if choice in ('1', '2', '3', '4', '5'):
        try:
            a = float(input("Enter your first number: "))
            b = float(input("Enter your second number: "))
        except ValueError:
            print("Sorry, wrong input")
            continue

        if choice == '1':
            print(a, "+", b, "=", add(a, b))
        if choice == '2':
            print(a, "-", b, "=", subtract(a, b))
        if choice == '3':
            print(a, "*", b, "=", multiply(a, b))
        if choice == '4':
            print(a, "/", b, "=", divide(a, b))
        if choice == '5':
            print(a, "%", b, "=", modulus(a, b))

        break
```

Choose your operation

1. Add
2. Subtract
3. Multiply
4. Divide
5. Modulate

Selecting choice(1/2/3/4/5): 5

Enter your first number: 50

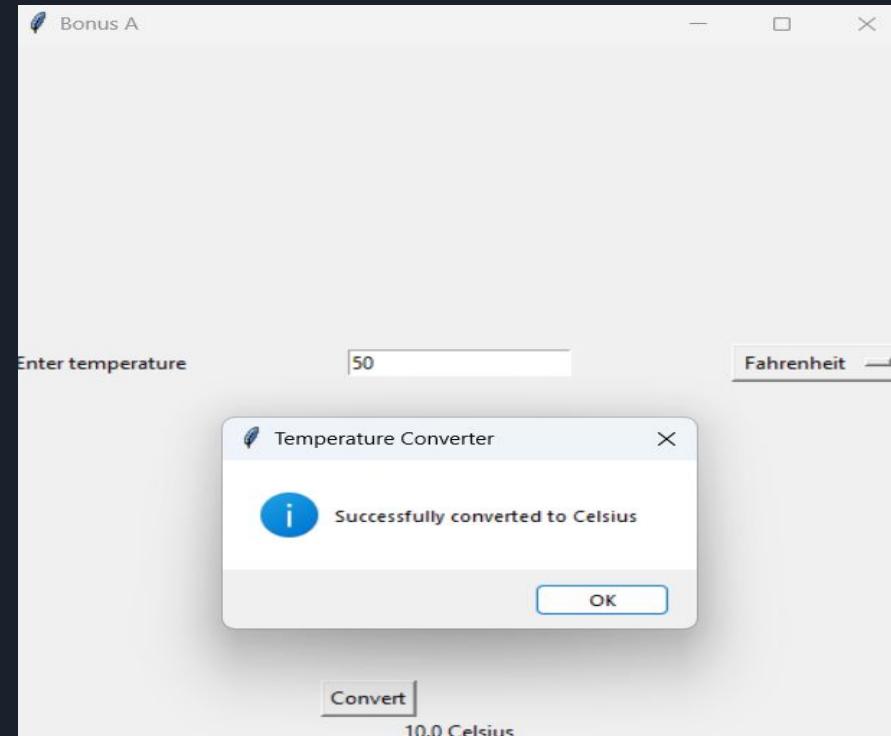
Enter your second number: 75

50.0 % 75.0 = 50.0

Bonus: Temperature Converter

```
import tkinter
from tkinter import *
from tkinter import messagebox
from functools import partial
temp_Val = "Celsius"
def store_temp(set_temp):
    global temp_Val
    temp_Val = set_temp
def call_convert(rlabel1,inputn):
    temp = inputn.get()
    if temp_Val == 'Celsius':
        f = float((float(temp) * 9 / 5) + 32)
        rlabel1.config(text = "%1f Fahrenheit" % f)
        messagebox.showinfo("Temperature Converter",
                           "Successfully converted to Fahrenheit ", )
    if temp_Val == 'Fahrenheit':
        c = float((float(temp) - 32) * 5 / 9)
        rlabel1.config(text = "%1f Celsius" % c)
        messagebox.showinfo("Temperature Converter",
                           "Successfully converted to Celsius " )
return
```

```
main = Tk()
main.geometry('500x500')
main.title('Bonus A')
main.grid_columnconfigure(1, weight = 1)
main.grid_rowconfigure(1, weight = 1)
Ninput = StringVar()
var = StringVar()
Linput = Label(main,text="Enter temperature")
Einput = Entry(main,textvariable=Ninput)
Linput.grid(row=1)
Einput.grid(row=1,column=1)
Rabel = Label(main)
Rabel.grid(row=3,columnspan=4)
dropdownlist = ["Celsius", "Fahrenheit"]
dropw = OptionMenu(main,var,*dropdownlist,
                   command = store_temp)
var.set(dropdownlist[0])
dropw.grid(row = 1, column = 2)
call_convert = partial(call_convert, Rabel,
                      Ninput)
Besult = Button(main, text ="Convert",
                command = call_convert)
Besult.grid(row = 2, columnspan = 2)
main.mainloop()
```



Bonus: The Age Calculator

```
import tkinter
You, last week • Updating Chapter 2's bonuses
from tkinter import *

from datetime import date

today = date.today()

def exit():
    main.destroy()

def get_age():
    d= int(e1.get())
    m=int(e2.get())
    y=int(e3.get())
    age=today.year-y-(today.month, today.day)<(m,d)
    t1.config(state='normal')
    t1.delete('1.0', END)
    t1.insert(END,age)
    t1.config(state='disabled')

main = Tk()
main.geometry("400x300")
main.config(bg="navyblue")
main.resizable(true,true)
main.title('Bonus B')

l1 = Label(main,text="The Age Calculator",font=("Helvetica",20,'bold'),fg="skyblue",bg="navyblue")
l2 = Label(main,font=("Helvetica",12,'bold'),text="Please designate your birthday",fg="skyblue",bg="navyblue")

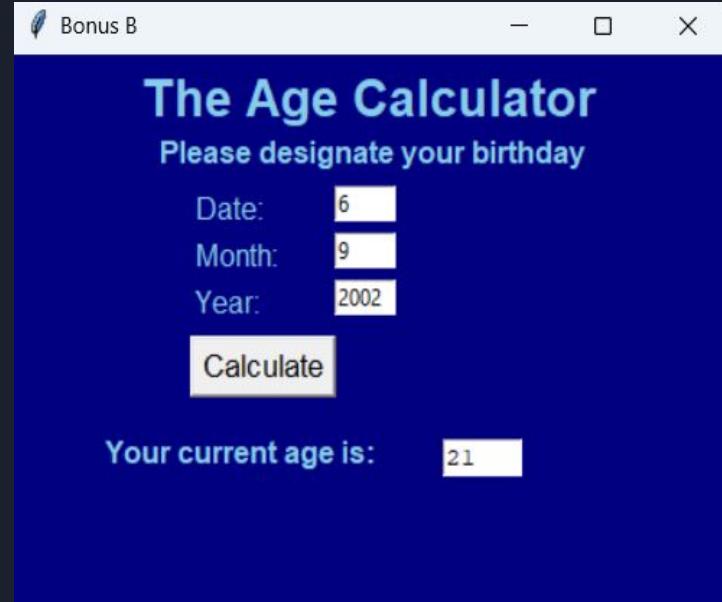
day = Label(main,text="Date: ",font=("Helvetica",12),fg="skyblue",bg="navyblue")
month = Label(main,text="Month: ",font=("Helvetica",12),fg="skyblue",bg="navyblue")
year = Label(main,text="Year: ",font=("Helvetica",12),fg="skyblue",bg="navyblue")
e1 = Entry(main,width=5)
e2 = Entry(main,width=5)
e3 = Entry(main,width=5)

b1 = Button(main,text="Calculate",font=('Helvetica',12),command=get_age)

l3 = Label(main,text="Your current age is: ",font=("Helvetica",12,'bold'),fg="skyblue",bg="navyblue")
t1 = Text(main,width=5,height=0,state="disabled")

l1.place(x=70,y=5)
l2.place(x=80,y=40)
day.place(x=100,y=70)
month.place(x=100,y=95)
year.place(x=100,y=120)
e1.place(x=180,y=70)
e2.place(x=180,y=95)
e3.place(x=180,y=120)
b1.place(x=100,y=200)
l3.place(x=50,y=200)
t1.place(x=240,y=205)

main.mainloop()
```



END