ECE 122: Introduction to Programming for ECE- Spring 2021

Project 3- **Part 1- Bonus 15pts**: Drawing with Tkinter Mapping
(OOP, Encapsulation and Graphics examples)

Due Date: **See website, class policy and moodle for submission**

Description

The goal of this project is to practice some OOP using class, instance variables and methods. You will also operate Graphics.

The first part of the project include one file:

1. `Mapping_for_Tkinter.py`: module containing the class `Mapping_for_Tkinter` (that will be used by all other application files later on) and a main function for testing and plotting mathematical functions.

Preliminary

Once you create a simple root window with its associated canvas in Tkinter of size (width,height), the origin is located at the top left corner and the y-axis points down. From now on, we will talk about (i,j) coordinates for the Tkinter canvas such that (i=0,j=0) for the top left corner, (i=0,j=height-1) for the bottom left corner, (i=width-1,j=0) for the top right corner, and (i=width-1,j=height-1) for the bottom right corner. One can then quickly realize that this coordinate system is not that convenient if you want to represent mathematical expressions. Math equations are better represented using a traditional coordinate system where the x-axis extends from x_{min} to x_{max} (left to right) and the y-axis extends from y_{min} to y_{max} (bottom to up). We propose to develop a mapping that will allow us to convert any (x,y) points in the math system to the corresponding points (i,j) in the Tkinter canvas, and vice-versa. The two coordinate systems are represented in Figure 1.

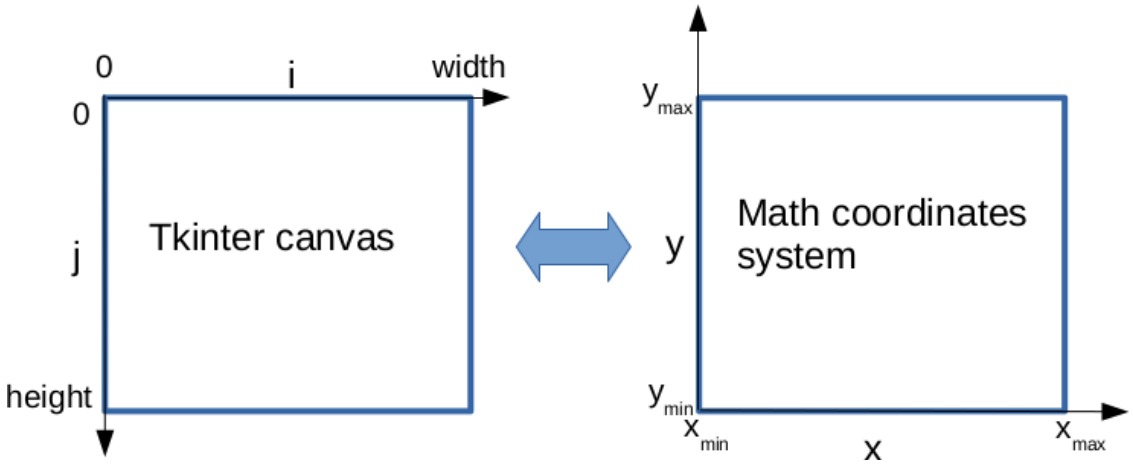

Figure 1: Tkinter canvas (left), Math coordinate systems (right).

To achieve this mapping, we propose to implement a new class `Mapping_for_Tkinter` that will be used to instantiate a new mapping object, i.e.

```
mapping=Mapping_for_Tkinter(-5.0,5.0,-5.0,5.0,500) # instantiate mapping
```

We consider the following arguments for `Mapping_for_Tkinter(xmin,xmax,ymin,ymax,width)` where `xmin`, `xmax`, `ymin`, `ymax` (float numbers) will be used to set the math coordinate system, and `width` (int number) is the width of the Tkinter canvas that we want to create (width is in number of pixels along x). The class will use those arguments to initialize the instance attributes (of the same name). In addition, you will consider the attribute `height` that must be calculated proportionally to the math coordinates i.e.

$$height = width * \frac{y_{max} - y_{min}}{x_{max} - x_{min}}$$

Also `height` will need to be converted to an int number (since it represents a number of pixels).

All attributes should be private and you will be using encapsulation. You will use the traditional OOP way to implement encapsulation using getter and setter methods. You will then need to implement: `set_xmin`,`set_xmax`,`set_ymin`,`set_ymax`,`set_width`, `__set_height` (which is a private method), as well as
`get_xmin`,`get_xmax`,`get_ymin`,`get_ymax`,`get_width` and `get_height`.

Once this is done, you need to define the instance methods that will allow us to make use of the mapping. You will need to implement:

- `get_x` that accepts i (the x coordinate in the Tkinter canvas) as argument and returns the x coordinate in the math format.
- `get_y` that accepts j (the y coordinate in the Tkinter canvas) as argument and returns the y coordinate in the math format.
- `get_i` that accepts x in math format and returns the i (int) coordinate in the Tkinter format.
- `get_j` that accepts y in math format and returns the j (int) coordinate in the Tkinter format.

Hint: This project is about finding linear formula that enable change of coordinates (between i and x, and between j and y). You can use the following mapping for (i,x): if $i = 0$ then $x = x_{min}$ and if $i = \text{w}idth - 1$ then $x = x_{max}$, and for (j,y): if $j = 0$ then $y = y_{max}$ and if $j = \text{h}eight - 1$ then $y = y_{min}$.

Complete the entire class which already contains the following main method:

```
def main():
    """ TESTING MAPPING using FUNCTION PLOTTER """

    #### formula input
    formula=input("Enter math formula (using x variable): ")

    #### coordinate input
    coord=input("Enter xmin,xmax,ymin,ymax (return for default -5,5,-5,5): ")
    if coord=="":
```

```
        xmin,xmax=-5,5
        ymin,ymax=-5,5
    else:
        #split the string/create list of string
        xmin,xmax,ymin,ymax=coord.split()

    #### instantiate a mapping
    width=800
    m=Mapping_for_Tkinter(float(xmin),float(xmax),float(ymin),float(ymax),width)

    #### instantiate a tkinter window
    window = Tk()
    canvas = Canvas(window, width=m.get_width(),height=m.get_height(),bg="white") # create a canvas width*height
    canvas.pack()

    #### create axis
    if m.get_xmin()<0 and m.get_xmax()>0:
        canvas.create_line(m.get_i(0.0),m.get_j(m.get_ymin()),m.get_i(0.0),m.get_j(m.get_ymax()))
    if m.get_ymin()<0 and m.get_ymax()>0:
        canvas.create_line(m.get_i(m.get_xmin()),m.get_j(0.0),m.get_i(m.get_xmax()),m.get_j(0.0))

    #### plot function
    for i in range(width):
        x=m.get_x(i)
        y=eval(formula)
        canvas.create_rectangle((m.get_i(x),m.get_j(y))*2,outline="blue")

    window.mainloop() # wait until the window is closed
```

Here is an example of output after execution:

```
Enter math formula (using x variable): 0.4*x*sqrt(abs(x))
Enter xmin,xmax,ymin,ymax (return for default -5,5,-5,5):
```

where we choose to enter our own formula but use default values for the boundary of the coordinate systems. We also obtain the plot in Figure 2.

Let us see the result of another formula below that gives the output in Figure 3.

```
Enter math formula (using x variable): sin(x)**2
Enter xmin,xmax,ymin,ymax (return for default -5,5,-5,5): -6.28 6.28 -0.01 1.1
```

How does the main program work?

1. the mathematical formula is stored as a string. It is notoriously difficult to parse a mathematical formula to identify and extract the math operations. Luckily, python provides a very powerful function called `eval` that can be used to evaluate a string as if it was a piece of code. For example:

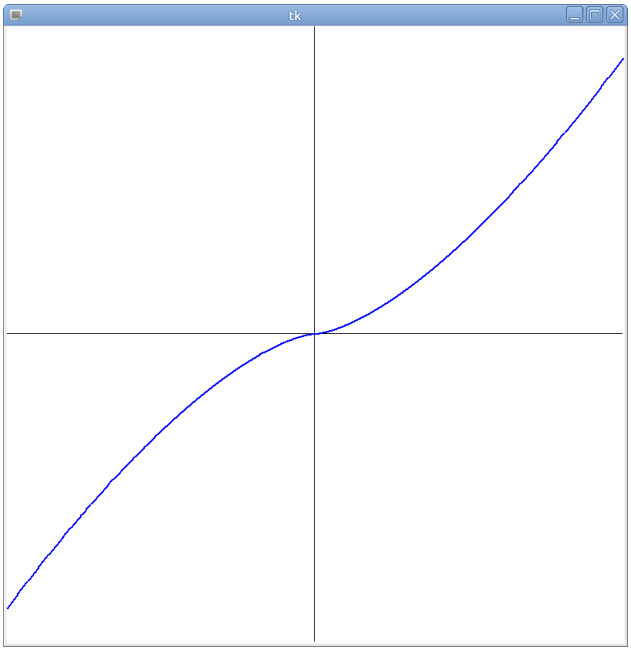

Figure 2: Example output of Task2- $0.4 * x * \sqrt{|x|}$

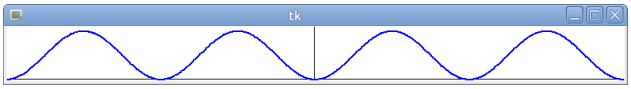

Figure 3: Example output of Task2- $\sin(x)^2$

```
x=3.0
y=eval("3*x")
print(y) # that will display 9!!
```

In general, the function eval could be dangerous to use for security reasons (you can basically enter any codes including malware, virus, etc.). But it is fine for our purpose here.

2. The program is using a width of 800 for the canvas for all situations here. The value for the height would depends on the window ratio of the math coordinates.

3. If 0 is included between `[xmin,xmax]` and/or `[ymin,ymax]`, the program is plotting the x and y axis (in black).

4. The math function is then evaluated and plotted point by point (in blue).

Finally, we would also like to have some safeguards in case of typos while entering the coordinates, for example:

```
Enter math formula (using x variable): sin(x)
Enter xmin,xmax,ymin,ymax (return for default -5,5,-5,5): 3.14 -3.14 3.14 -3.14
Your xmax is invalid (xmax<=xmin), Re-Enter correct [xmin,xmax]: -3.14 3.14
Your ymax is invalid (ymax<=ymin), Re-Enter correct [ymin,ymax]: 1 -1
Your ymax is invalid (ymax<=ymin), Re-Enter correct [ymin,ymax]: -1 1
```

In this case, you will need to modify accordingly the `set_xmax` and `set_ymax` methods from the class.