Laboratory Activity No. 11	
The Grid Manager	
Course Code: CPE103	**Program:** BSCPE
Course Title: Object-Oriented Programming	**Date Performed: APRIL 5, 2025**
Section: 1A	**Date Submitted: APRIL 5, 2025**
Name: CATAHAN, JOSHUA A.	**Instructor: ENGR. MARIA RIZETTE SAYO**

1. Objective(s):

This activity aims to familiarize students on how to implement geometry manager

2. Intended Learning Outcomes (ILOs):

The students should be able to:
2.1 Identify the main components in a GUI Application
2.2 Create a simple GUI Application using Grid manager

3. Discussion:

A Graphical User Interface (GUI) application is a program that the user can interact with through graphics (windows, buttons, text fields, checkboxes, images, icons, etc..) such as the Desktop GUI of Windows OS by using a mouse and keyboard unlike with a Command-line program or Terminal program that support keyboard inputs only.

Geometry managers are tools used to place widgets on the screen. There are three geometry managers available in tkinter—grid, pack, and place. The place manager provides complete control in the positioning of widgets, but is complicated to program

Grids

- A grid is an imaginary rectangle containing horizontal and vertical lines that subdivide it into rectangles called cells. The first row of cells is referred to as row 0, the second row is referred to as row1, and so on. Similarly, the first column of cells is referred to as column 0, the second column of cells is referred to as column 1, and so on. Each cell is identified by its row and column numbers.

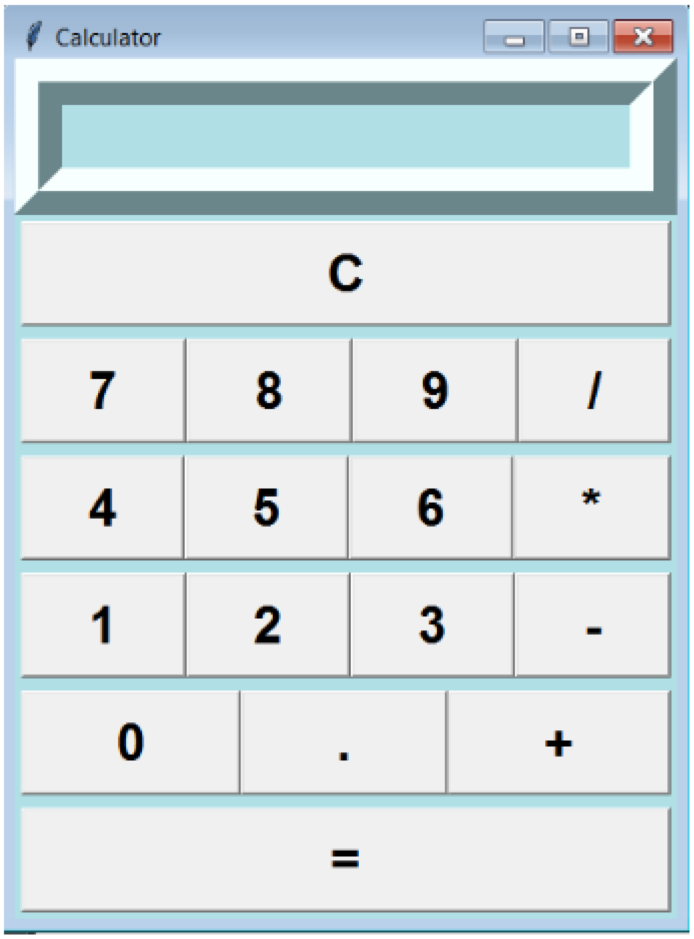
4. Materials and Equipment:

Desktop Computer with Pycharm
Windows Operating System

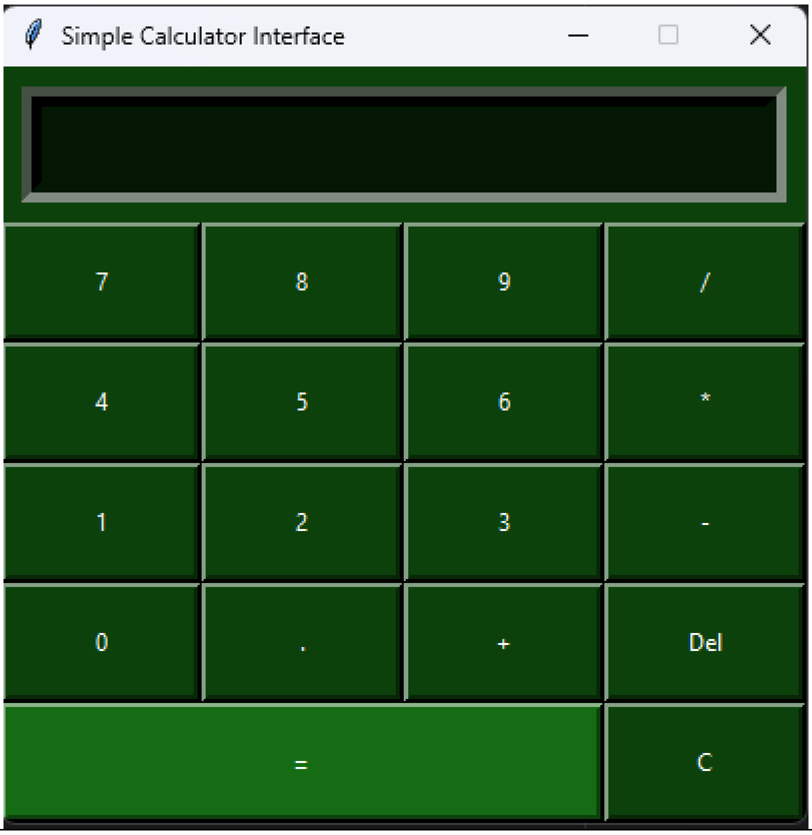
5. Procedure:

General Instruction:

1. Redesign the interface of the standard calculator using grid ( ) method:



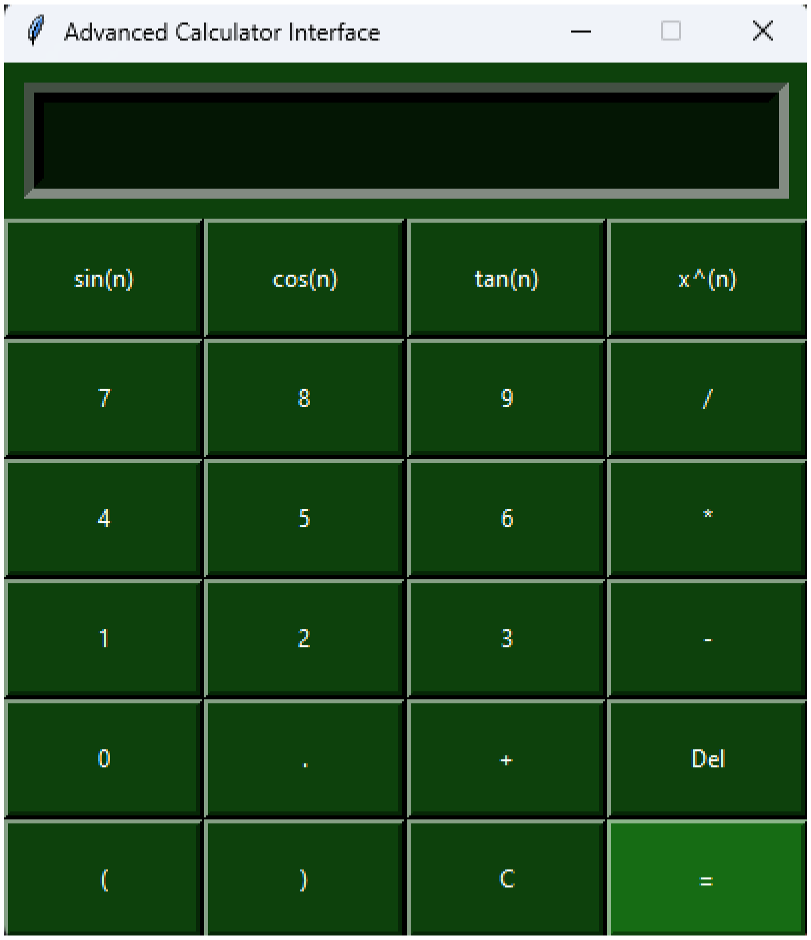
2. Run the program and observe the output when the button is clicked.



6. Supplementary Activity:

1. Make a calculator program that can compute perform the Arithmetic operations as well as exponential operation, sin, cosine math functions as well clearing using the C button and/or clear from a menu bar.
2. Use Geometry manager grid()

3. Use bind () or command parameter in associating event to callback a function.



Note: Don’t click the text box, when you want to use the keyboard. You can type in the keyboard directly because it uses the bind function.

Questions

1. How do you configure rows and columns in PyCharm when using Tkinter's grid() manager?

In PyCharm, when I use Tkinter’s grid() manager, I set each widget to a specific row and column by using the row and column parameters. For example, if I want my entry field in the first row and first column, I write entry.grid(row=0, column=0). I can also use options like columnspan to make a widget span multiple columns, or sticky to control the widget's alignment within the grid cell.

2. Why do widgets sometimes disappear when using grid() in PyCharm, and how can you fix it?

Sometimes, widgets disappear when using the grid() method because they might be placed outside the visible area of the window, or the grid configuration might overlap. To fix it, I make sure the row and column numbers are correct, and I avoid using both pack() and grid() on the same widget. Additionally, I use sticky to align the widget properly within its grid cell and check for any rowspan or columnspan issues to prevent widgets from being hidden.

3. How can message boxes be used to provide a better User Experience or how can message boxes be used to make a GUI Application more user-friendly? How can you align widgets across multiple frames using grid() in PyCharm?

I can use message boxes, like tk.messagebox.showinfo() (from the past laboratories), to improve the user experience by providing helpful feedback or alerts. In my calculator, for example, I could use a message box to inform the user if there’s an error (In which I did not). To align widgets across multiple frames, I use the grid() method with specific row and column values for each widget. For example, I can position widgets in the first frame with grid(row=0, column=0) and in another frame with grid(row=0, column=1) to keep everything aligned neatly.

7. Conclusion:

In conclusion, this laboratory activity helped me expand my knowledge of Tkinter’s grid() manager in Python using PyCharm. Although I initially found it confusing to implement some of the functions for the calculator, revisiting past code examples helped me get started. I learned how to create buttons and associate them with functions like basic arithmetic, trigonometric operations, and handling user inputs. By building the calculator, I also gained hands-on experience with Tkinter's event-driven model and how to manage a simple user interface. The process improved my understanding of organizing the layout using grid(), handling mathematical expressions with Python's eval() function, and ensuring the program could handle errors gracefully. Overall, this activity reinforced my skills in Python and Tkinter, and I look forward to gaining additional knowledge from the next laboratory activities.

8. Assessment Rubric: