SOLAR ENERGY CALCULATION- CUSTOM PROGRAM\_COS10009

NAME: HUYNH QUOC HIEU

STUDENT ID: SWH01668

PESUDOCODE:

1. Declare a class SolarApp that inherits from Gosu::Window

2. Initialize:

- Set window dimensions (800x400)

- Set window title to "Solar Energy Calculation App"

- Create a font object

- Read energy data from "energy\_data.csv"

- Initialize @selected\_hour (stores selected hour) to nil

3. Define the draw method:

- Draw the title "Select Time:" at the top-left corner

- Draw 24 buttons (0-23 hours) with the label "h" (for hour)

- If a button is selected, fill it with green; otherwise, fill it with gray

- If @selected\_hour is not nil (an hour has been selected):

- Find the corresponding data row in @energy\_data

- If data is found, call draw\_energy\_info to display the information

4. Define the button\_down method:

- If the left mouse button is clicked:

- Check if the click is within the bounds of any button

- If so, update @selected\_hour to the corresponding hour

5. Define the draw\_energy\_info(row, x, y) method:

- Extract solar\_power\_generated, battery\_energy, and energy\_consumption from the row

- Calculate total\_energy\_available

- Display the following information in yellow text:

- Time (e.g., "Time: 14:00")

- Solar Power (kWh)

- Battery Energy (kWh)

-Energy consumption (kwh)

- If total\_energy\_available is less than energy\_consumption:

- Display "Warning: Insufficient energy!" in red

- Otherwise:

- Determine battery status (charging, discharging, or neither)

- Display battery status in the appropriate color (green, yellow, or black)

- If the battery is charging or discharging, display the excess energy or energy from the battery

- Display the remaining battery energy

6. Create a new SolarApp object and show the window