TWO PASS ASSEMBLER DOCUMENTATION

OVERVIEW

This project implements a basic two-pass assembler using Python's **`Tkinter`** library for GUI. The assembler reads assembly code, processes it to build a symbol table in Pass One, and then generates a machine code in Pass Two. Users can input their assembly code into the GUI and view the output after each pass.

COMPONENETS USED

- GUI LAYOUT
  - **Input Area:** A `Text` widget where users input assembly code.
  - **Buttons:**
    - `Run Pass One`: Processes the code to create a symbol table.
    - `Run Pass Two`: Generates machine code using the symbol table.
  - **Output Box:** A `Text` widget that displays the result of each pass.

- PASS ONE
  - Initializes a **location counter**.
  - Processes each line to detect labels and instructions.
  - If an instruction like `DATA` or `LOAD` is encountered, the location counter is updated, and the label is added to the **symbol table**.

- PASS TWO
  - Converts assembly instructions into **machine code** using the symbol table generated in Pass One.
  - `LOAD` and `STORE` instructions reference the symbol table for addresses; unknown labels are marked.

- FUNCTIONS
  - **`update_output(text)`**
    - Updates the output display with results from either pass.
  - **`pass_one()`**
    - Parses the input assembly code.
    - Builds a symbol table with label addresses.
    - Output: Symbol table displayed in the output box.
  - **`pass_two()`**

- Converts the input code into machine code using the symbol table.
- Handles `LOAD` and `STORE` instructions.
- Output: Displays machine code in the output box.