Group 9 Members:

- Laurenz Luke Gonzales
- Mahooba Hashimi
- Saviour Msopa

GitHub Link: https://github.com/lgonza32/CSCI6461_Group9_Project.git
Guide/Instructions show in GitHub Readme.md

Goal for Part 1:
Implementing GUI and simulator backend, specifically

- Manual entry of values into registers (R0-R3)
- IPL to preload a short program (test_load.txt from loading in test.asm from modified part 0)
- Single Step that fetches and executes instructions

Assumptions/Limitations:

- Minimal opcode support, specifically based on ISA provided figure 3 Sample Assembler Source File
    - HLT, LDX, LDR, LDA, JZ
- Run/Halt may be stubbed
- Printer/Console fields are placeholders and not fully implemented
- Cache Content is used as memory window/MAR-value display

IPL design design

- Start at first instruction
- IPL loads the *_load.txt file into memory (from test.asm)
- Uses the *_listing.txt (if present) to identify the first non-Data/non-LOC line and sets PC to that location
- PC falls back to firstAddress if listing is missing

Switch input design

- Set Target panel supports writing values into PC/MAR/MBR/R0-R3/X1-X3 from operator input
- Auto detects logic
    - Octal input empty => use binary
    - Else if binary contains at least 1 => use binary
    - Else => use octal

Architecture
/src/simulator/ui/GUI.java

- Built with JavaSwing and Java AWT
- Renders the panel to display registers switches, memory operations, control buttons, chache/printer/console
- No machine logic

- Pulls from MachineState.java via refreshFromController() and displays them in octal

/src/simulator/control/Controller.java
- Handles machine code into GUI
- Performs/Handles
    - Memory in 2048 word memory array
    - MachineState
    - CPU
    - IPL loading (parse load file, write to memory, set PC start)
    - Set target logic using binary + octal input
    - Load/store/load+/store+ using MAR/MBR
    - Reset clearing registers + memory
    - Single Step execution via CPU.step()
- Callbacks to GUI
    - log(strring) console input
    - setProgramFilePath(String) for Program File Field
    - setCacheText(String) for Cache Content
    - refreshUI.run() to refresh register fields

/src/simulator/machine/Memory.java
- Implements memory as int[2048]
- Enforces bounds and basks values to 16 bit on write
- toOct6() helper provides consistent 6-digit octal display for UI/logs

/src/simulator/machine/MachineState.java
- Memory
    - Size: 2048 words
    - Size: 16-bit
- MachineState
    - GPR: 16 bit
    - IXR: 16 bit
    - PC, MAR 12 bit
    - MBR, IR: 16 bit
    - CC, MFR: 4 bit
    - Setter and mask values to correct bit width to prevent overflow

/src/simulator/io/ProgramLoader.java
- Parses a load file
- Produces (address, word) records + firstAddress
- Values octal by default

/src/simulator/cpu/CPU.java
- Implements the single step fetch/decode/execute
- Implements EA logic

- Indexing (IXR) + indirect bit (I)