Assembly Interpreter Handbook

Introduction

This document explains how to use the Assembly Language Interpreter.
It simulates a simple CPU with memory, registers, and instruction logic.

You can run it in three modes:
- Standard: normal execution
- Debug: step-by-step mode
- Set: edit memory/registers interactively

Instruction Set

Available Instructions:

Load R [M] Load memory[M] into register R
Store R [M] Store value from register R into memory[M]
Set R = C Set constant C into register R
Add Rx, Ry, Rz Set Rz = Rx + Ry
Neg R Set R = -R
Jump #L Unconditional jump to line number L
JP R, #L Jump to line L if value in R > 0

Running the Script

Usage:
  python assembly.py [-f FILE] [-d] [-s]

Arguments:
  -f, --file Path to assembly file
  -d, --debug Enable debug mode (step-through)
  -s, --set Start live register/memory editing

If no file is provided, a file chooser appears.

Debug Mode

Assembly Interpreter Handbook

Features:

- See full instruction list
- Current line is highlighted
- Use Arrow Right / Space / Enter to step
- Press ESC to toggle between views
- In overview view, press Enter to quit

Set Mode

Features:

- Lets you manually edit registers and memory
- Format:

    R1 = 5 (sets register R1)

    41 = 99 (sets memory address 41)

- Type 'exit' to quit and save

Only memory is saved between runs (in memory.json).

Notes

- Registers reset every run
- Memory (0-1023) is persistent
- Invalid commands show friendly error messages with line info.

Example Script

```
Load R1 [41]
Load R4 [40]
Set R2 = 2
Neg R2
Add R1, R2, R1
JP R1, #5
Set R3 = 1
Add R1, R3, R1
JP R1, #11
Jump #13
```

Assembly Interpreter Handbook

```
Add R4, R3, R4
Store R4 [40]
```