# Proposal: Legv8 Simulator

**Team Members: Andy Dang, Sergio Hernandez, and Colin Dutra**

- Data structures:
  - Map of all instruction types. A pair would have a string key and an integer value.
    - Used to identify what instruction format the line will be.
  - Integers to contain flags
  - Integer N to keep track of current instruction
  - Arrays for MEM and RFILES
  - Vectors for PGM and STACK
  - Strings to contain a line of code and segments of the codes. (Possibly will change this to use a custom class for cleaner code).
- Pseudo-codes:
  - Parsing:
    - Will grab all instructions from the text file and store each line onto the vector PGM, initialize all MEM[i#] = v# from user inputs, go through the PGM vector and find the token "main" and return the index of that instruction line.
  - Execution:
    - Main exec function will be called in a loop from main in a while N is within the boundary of the PGM size. In main exec function, it will call the helper exec functions to specifically parse and execute the instruction based on its format type.
- Functions/Methods information:
  - There will be a main, parse, and 11 exec functions.
  - Parse will initialize global variables based on user inputs, store all instructions into PGM vector from input file, and return index of main.
  - There will be a main exec function that get the current instruction line, get the instruction type, then call the function exec for that specific instruction type.
  - There will be 10 exec functions for each format type. Example) R-format without flags, R-format with flags, I-format without flags, I-format with flags, etc.
    - Each of these functions will have inputs of instruction type number from the map and the instruction line.
    - Will parse the rest of the parameters for the instruction.
    - Will be a series of switch statements to determine how to specifically execute the instruction.
    - N will be incremented based on the specific instructions. Example) branch and non-branch instructions.
- Extra credit:
  - GUI for the simulator that allows users to create, edit, save, and execute Legv8.