Weeks 1-2

1. Numerical representations
   1. Limitations & overflows
   2. Number bases & conversions
2. Logic circuits and hardware
   1. Boolean logic & binary operations
   2. Design a 2-bit binary adder
   3. Add overflow detection to the adder
3. Processors
   1. Op codes (sequential list of numerical instructions)
   2. Conceptualize a trivial/simplistic compiler (add, read from memory, write to memory)
4. Programming languages, compilers, and interpreters
   1. “Compile” some simple addition problems for the simplistic compiler
      1. Output to a volatile “RAM” memory line/grid with 4-bit address space
         1. Define special ram locations that “print” to some display or 7-seg
      2. Program “pointer” follows a program memory line/grid with 4-bit address space
5. Command line interface
   1. File paths ‘/’, home directory, ~, cd, .., ., ls, cat, nano, >, javac, java
   2. Unix: everything is a file
   3. Compiling and running a Java program
      1. Learning the “boilerplate” class and main method template
      2. Simple C-like coding (just lots of variables at this point, but progress toward an example where numerous duplicated variables shows how the concept of an “object” makes sense)

**Lab:**

1. “Compile” an algebraic expression for the “trivial processor”
2. Java program: calculate final distance of a trajectory

**Exercises:**

1. Logic circuit: design a 2-bit adder
2. Add feature to lab program: calculate maximum height of the trajectory

Extra credit: calculate height and distance at four (4) arbitrary points in time