The **goal** of this assignment is to get some more practice in mapping assembly instructions to hardware controls, with machine code as an intermediate step. This process is one we have practiced a few times using pen and paper. This assignment asks for a program that automates that process. We will use the model outlined in Section 4.4 of the textbook[1] using the discussion forum for further details as usual.

The **packaging and submission** requirements are similar to those in the previous homework assignments, using the `hw05` tag instead of `hw04`. Similarly, the rules for using additional **resources** are the same as before.

The `hw05` implementation should use Python 3 and read from standard input and write to standard output, with optional diagnostics to standard error.

The **input** to the program is a sequence of (zero or more) RISC-V assembly language instructions, one per newline-terminated line, using the specific syntax of RARS(M)[2] but limited to instructions (that is, without assembler directives, macros, etc.). Blank lines (those containing only white-space) should be ignored. Further, lines whose first non-blank character is `#` should also be ignored. (The may be used for comments in the input, but additional comment syntax, such as comments following an instruction on the same line need not be supported.)

The **output** of the program is also a sequence of newline-terminated lines, with two lines of output for each line of input (in corresponding order). The first line of output corresponding to a line of input (that is, to an instruction on that input line) is the machine code corresponding to that instruction, printed as a binary string (without any additional prefixes or indicators). The second line of output corresponding to that line of input provides the values of the hardware control signals for that instruction, from Figure 4.11 of the textbook, listed in alphabetical order, with a single space separating adjacent values. The values are provided as binary strings similar to the one used for the machine code.

The `hw05` program should produce the two lines corresponding to each instruction on an input line immediately after that line is read (not, for example, delayed until after reading all or several instructions from the input). There should be no extraneous output such as prompts or diagnostics on the standard output (but such output may optionally appear on standard error).

The program should implement at least the instructions listed near the beginning of Section 4.1 of the textbook (page 254) but support for additional instructions is encouraged (not required). These and other details should be described in the README file included in the submission package as usual.

[1] David A Patterson and John L Hennessy, *Computer Organization and Design RISC-V Edition*, 2nd edition (Morgan Kaufmann, 2020).

[2] Jean Privat and others, RARSM—RISC-V Assembler and Runtime Simulator (iMproved), `https://github.com/rarsm/rars`, 2024.