File format: First value gives the number of instruction words (the number of instructions plus all the operands; for example `add r1 r2 r3` has size 4 words). The second value begins the code (it is position 0 of the code array).

LOAD Load a literal value into a register. Code 1. Example: `load 10 r2` or `1 10 2`. Meaning: Store value 10 in register 2.

MOVE Move the value in one register into another. Code 2. Example: `move r2 r3` or `2 2 3`. Meaning: Store (copy) the value in register 2 into register 3.

ADD Add the values in two registers together, storing them in a third. Code 3. Example: `add r4 r3 r3` or `3 4 3 3` Meaning: Add the values in registers 4 and 3 together, overwriting register 3 with the answer.

SUB Compare ADD. Code 4.

MUL Compare ADD. Code 5.

DIV Compare ADD. Code 6.

IF Jump to the code position stored in a given register if another register is not equal to zero. Code 7. Example: `if r3 r5` or `7 3 5`. Meaning: Check the value stored in register 3; if it is not 0, then jump to the instruction at the index stored in register 5.

PRNT Print the value in a register to the screen. Code 8. Example: `prnt r3` or `8 3`. Meaning: Print whatever value is contained in register 3 to the screen.

HALT End execution. Code 9.

READ Read a value from main memory and store it in a register. Code 10. Example: `READ r1 r2` or `10 1 2`. Meaning: Read the value in memory at the address stored in register 1 and store the result in register 2.

WRT Write a value to main memory. Code 11. Example: `WRT r1 r2` or `11 1 2`. Meaning: Write the value stored in register 2 to main memory at the address stored in register 1.