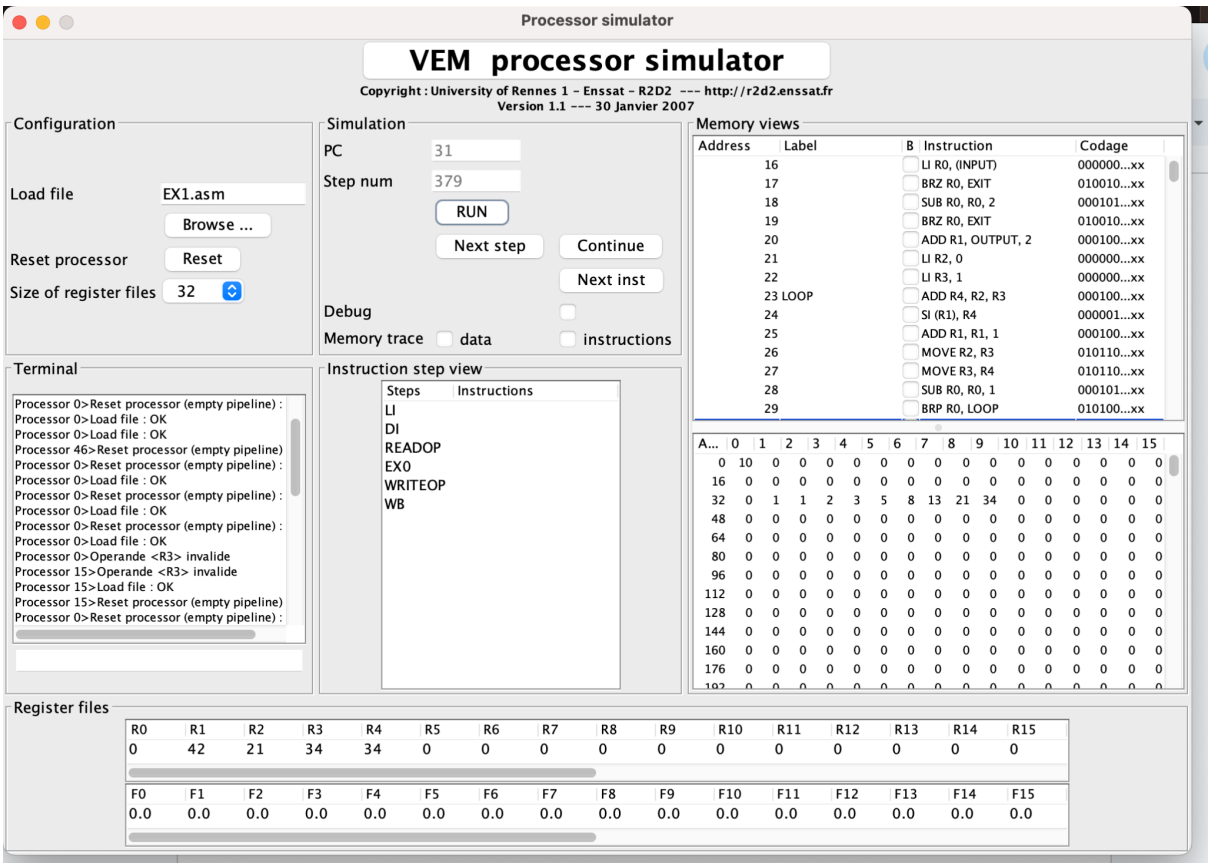

INPUT

```
.data 0                 -- Initialize the data segment
.global INPUT             -- Declare the symbol INPUT as global
    3 8 5 2 9 6             -- Initialize an array named INPUT with values
```

OUTPUT

```
.data 32
.global OUTPUT
    0 1                         -- Declare another global symbol OUTPUT with initial values 0 and 1
```

```
-- Declare a global symbol INPUT with the value 10



-- Declare another global symbol OUTPUT with initial values 0 and 1

-- Set the program counter to 16
-- Load the value of INPUT into register R0
-- If R0 is zero, exit the program
-- Subtract 2 from R0
-- If the result is zero, exit the program
-- Add 2 to the address of OUTPUT and store the result in R1 (points to the second element of
-- Initialize R2 with 0
-- Initialize R3 with 1

-- Begin a loop labeled .loop
-- Calculate the sum of R2 and R3 and store it in R4 (Fibonacci sequence)
-- Store the value of R4 in the memory location pointed to by R1 (store the Fibonacci number)
-- Increment R1 to point to the next memory location
-- Move the value of R3 to R2
-- Move the value of R4 to R3
-- Decrement R0 (loop counter)
-- If R0 is positive, branch back to .loop

-- Label for program exit
-- Exit the program
-- End of the program
```