Overview:
This exercise will give you hands-on experience at how a computer works. That is, you will become part of the computer. Each working group will “portray” a particular piece of the Von Neumann architecture and the class as a whole will work together to execute a simple program.

The Pieces:
Each group will portray one of the 8 parts of the Von Neumann architecture described below. Recall the structure of the architecture:

```
   Bus
  /   \
 Memory  Control Unit  Input/Output
       |         |
       |         |
       ALU     Processor

```

Memory:
1. MAR (Memory Address Register) and Memory Decoder Circuits
   • holds the address involved in a particular operation
   • decodes the binary representation of the address to point to the actual location in memory

2. MDR (Memory Data Register) and Fetch/Store Controller
   • holds the data involved in a particular operation
   • controls flow of data in and out of memory

3. RAM (Random Access Memory)
   • memory cells that contain the data and program instructions
   • for the purpose of this exercise, we will assume that there are 16 memory cells, numbered 0000 to 1111

ALU (Arithmetic Logic Unit):
4. Register R
   • single register for holding operands for mathematical operations

5. ALU Multiplexor
   • determines which operation result to place back in register R

6. Condition Code Register
   • Holds the results of comparison operations
   • GT, LT, EQ – set to 0 or 1 depending on the result of the operation
Control Unit:

7. Program Counter (PC)
   • holds the address of the next instruction to execute

8. Instruction Register (IR) and Instruction Decoder Circuit
   • holds the instruction that is to be executed
   • decodes the instruction and sends the correct signals to the ALU and Memory to execute the operations

For simplicity, we will not represent Input/Output in this exercise. The role of the bus will be played via “verbal communication” among the parts of the computer. That is, if the ALU needs to send something from its register R to a memory location, the group representing the ALU register will tell the group representing RAM to put the value in the location.

Initial Setup:

Memory:

The following is the initial contents of the RAM. We will use decimal representations of numbers and written instructions instead of binary representations for simplicity. However, understand that each of the items in the memory locations can be translated into binary.

We also assume 1-dimensional memory for simplicity.

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>1</td>
</tr>
<tr>
<td>0001</td>
<td>0</td>
</tr>
<tr>
<td>0010</td>
<td>3</td>
</tr>
<tr>
<td>0011</td>
<td></td>
</tr>
<tr>
<td>0100</td>
<td></td>
</tr>
<tr>
<td>0101</td>
<td></td>
</tr>
<tr>
<td>0110</td>
<td></td>
</tr>
<tr>
<td>0111</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>LOAD 0000</td>
</tr>
<tr>
<td>1001</td>
<td>ADD 0001</td>
</tr>
<tr>
<td>1010</td>
<td>COMPARE 0010</td>
</tr>
<tr>
<td>1011</td>
<td>JUMPGT 1000</td>
</tr>
<tr>
<td>1100</td>
<td>HALT</td>
</tr>
<tr>
<td>1101</td>
<td></td>
</tr>
<tr>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>

Program Counter:

The initial value in the program counter PC = 1000

How to Play:

Recall from Chapter 5, the three phases that describe the behavior of the Von Neumann computer:
Phases of the Von Neumann Computer:

1. Fetch Phase
   a) PC -> MAR  send address in PC to MAR
   b) FETCH  get contents of address in MAR and put in MDR
   c) MDR -> IR  move instruction from MDR to IR
   d) PC <- PC + 1  increment PC so that we can execute next instruction

2. Decode Phase
   • IR_{op} -> instruction decoder  decode the instruction in the IR

3. Execution Phase
   • Control unit generates the necessary signals to have the ALU, memory and I/O units perform according to the operations.
   • See attached sheet with details of how each operation is executed.

How to Start:

The exercise starts by executing the fetch phase, and following the above steps from there.

How to Stop:

The exercise is finished when the instruction sent to the instruction decoder is HALT.