1) Review the following C code segments. For each segment, discuss and write down “what is the state of these programs?” How/when is the state changing?

a)
```c
int variable1 = 3;
int variable2 = 8;

variable2 = variable2 + variable1;

int variable3 = variable2 + 5;
```

b)
```c
if (x >= 0) {
    x = x + 5;
}
x |= 1;
```

```c
void init_array_of_length_12(int array[12]) { 
    for (int i = 0 ; i <= 12 ; i ++) {
        array[i] = 0;
    }
}
```
\[ h(x, y, z) = (x'y'z + xy'z + x'y'z') \]

Draw a circuit for \( h(x,y,z) \):

---

Fill out a truth table for \( h(x, y, z) \). You may use the blank columns for intermediate values

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>h(x,y,z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Write a verilog expression for \( h(x, y, z) \).

```verilog
module h_circuit(h, x, y, z);
    //declare inputs, outputs, and wires

    // combine gates and wires

endmodule // h_circuit
```
Verilog Modules

module sc_block(s, c, a, b);
  output s, c;
  input a, b;
  wire w1, w2, not_a, not_b;

  // the "c" output is just the AND of the two inputs
  and a1(c, a, b);

  // the "s" output is 1 only when exactly one of the inputs is 1
  not n1(not_a, a);
  not n2(not_b, b);
  and a2(w1, a, not_b);
  and a3(w2, b, not_a);
  or o1(s, w1, w2);
endmodule // sc_block

Draw a circuit diagram for the Verilog module above:

![Circuit Diagram for sc_block](image)

Figure 1. AND, OR, and NOT gates.
Hierarchical design: building modules from modules

![Diagram of sc2_block](image)

**Figure 2.** The sc2_block, which is implemented using sc_block, and its partially completed truth table.

Complete the truth table and write the Verilog for the sc2_block module: