1. For the following example, use list scheduling to find the minimum latency schedule given the following resource constraints (where an ALU can perform addition, subtraction, and logic operations):

   Multiplier: 2 available, 2 unit delay
   Adder: 4 available, 1 unit delay
   ALU: 1 available, 1 unit delay

   \[ g = x + j \]
   \[ h = u \times (2 + x + u \times j) - (8 + y \times j) \]
   \[ i = y + u + j \]
   \[ c = g \land i \]
   \[ k = i - 5 + c \]

2. Convert the following C-like code to a high-level state machine using the following two approaches. For each implementation, including the original software, estimate the number of clock cycles required to complete execution and the overall size of the design. Please state your assumptions used for your estimates.

   a.) Convert the C-like code to a high-level state machine using the approach outlined within the Transmogrifier C Hardware Description Language. Do not convert all operations to bit-level operations. Instead, use the semantics for the identify states but extend the approach to allow multi-bit operations.
   
   *Note: You do not need to create a complete extension to the Transmogrifier C HDL. Keep it simple!

   b.) Convert the C-like code to a high-level state machine using the template based conversion approach. Optimize your design by eliminating unnecessary states whenever possible.

   ```c
   int i, j;
   int psum = 0;
   int thres = 45;

   for(i=0; i<=256; i++)
      for(j=0; j<256; j++)
         if( A[i][j] <= thres ) {
            psum = psum + A[i][j];
         } else {
            psum = psum - 2;
         }
   }
   output = psum / 2;
   ```