Testbenches revisited:

- Logic doesn’t go in initial blocks, delays do not go in always blocks (exception: clock)
- The initial block may contain an assignment to any reg declared in the testbench
- General outline for testbench initial block:
  - Load test vectors (if using test vectors)
  - Assert reset for one or more clock cycles
  - Set reset low
  - Main loop (repeat until tests are done):
    * Set any reg that takes its value from the test vector (i.e., the module’s inputs)
    * Wait for output (if output takes fixed delay or has valid signal)
    * Check output
  - $finish()

Example:
```
reg [3:0] a,ref_out; reg res; wire [3:0] out;
accumulator DUT (clk,a,res,out);
...
initial begin
  ref_out = 4’b0;
a = 4’b0;
res = 1;
#10; // where clock period is 10
res = 0;
#10;
for(i = 0; i < numtests; i = i + 1) begin
  a = {$random} & 4’b1111;
  #10;
  ref_out = ref_out + a; // simulate functionality of module
  if (out != ref_out) $display("Test failed!");
end
$finish();
end
```

Problem 1: LUT programming
You are given the following logic function to implement with a 4-Look-Up Table (4-LUT):

\[ Y = AC + BD + BA \]

What should the contents of the register holding the programming for the LUT be? (Write the MSB as the bit attached to the zero input of the multiplexer). A is attached to \( sel[0] \), B to \( sel[1] \), and so on.

Problem 2: LUT combination
How many 4-LUTs are required to implement any 6-ary logic function when a multiplexer with a 2-bit select is provided? How about if no multiplexer is provided?