4. (10 points) There are four Verilog code snippets in this section. Some of these codes have a problem with the **syntax**. For each code, first state whether or not there is a mistake. If there is a mistake explain how to correct it.
Note: Assume that the behavior as described, is correct

(a)

```
module one (input sel, input [1:0] data, output z);

          always @ (*)
              begin
                assign z= (sel) ? data[1]:data[0];
              end
endmodule
```

Solution: This code has mistakes. sequential assignments do not start with `assign`, these are reserved for combinational statements. In addition, if `z` was assigned in an `always` statement, it should have been declared as `reg`.

(b)

```
module two (input [1:0] sel, output reg [7:0] z);

  always @ (sel)
    if      (sel == 2'b01) z=8'b01010101;
    else if (sel == 2'b10) z=8'hAA;
    else                   z=8'd0;

endmodule
```

Solution: This code is correct, it is OK to write constants using different representations.

(c)

```
1  module three ( input [3:0] data, input sel1, input sel2, output z);
2
3    wire [1:0] m;   // actual exam had (a typo) : wire m;
4
5    module mux2 ( input [1:0] i, input sel, output z);
6      assign z= (sel) ? i[1]:i[0];
7    endmodule
8
9    mux2 i0 (.i(data[1:0]), .sel(sel1), .z(m[0]) );
10   mux2 i1 (.i(data[3:2]), .sel(sel1), .z(m[1]) );
11   mux2 i2 (.i(m), .sel(sel2), .z(z) );
12
13 endmodule
```

Solution: This code has mistakes. The sub module `mux2` should be defined outside the module `three` and not be part of it. Only the instantiations are part of the code. In the actual exam, there was a typo here, and m was declared as a single bit. Students that pointed out this typo got full points. This is the intended version.

(d)

```
1  module four (input [3:0] data, input [1:0] sel, output reg [3:0] z);
2
3    always @ (data, sel)
4        z = data;
5        if (sel[0])       z = ~data;
6        else if (sel[1])  z = 4'b0000;
7
8  endmodule
```

Solution: This code has mistakes. If there is no `begin` following the `always` statement, then only the first statement will be a sequential statement, the rest starting with the `if` will be interpreted as a separate combinational statement.