# Solutions

1. Solution 1

```
    module bitwise_or(a, b, c);
         input[3:0] a,b;
         output[3:0] c;
         reg [3:0] c;   // output is declared as reg and then assigned
using assign
         assign c = a | b;
         // to correct it, either remove line reg [3:0] c; or remove
assign keyword, and make this assignment inside an always block
         /*
            input[3:0] a,b;
            output[3:0] c;
            assign c = a | b;
         */
         /*
            input[3:0] a,b;
            output[3:0] c;
            reg [3:0] c;
            always @(a or b)
                c = a | b;
         */
     endmodule
```

2. Solution 2

```
    module and_gate(ina, inb, out);
        input ina;
        input inb;
        output out; // the out signal is assigned inside an assign
block without being declared as reg, declare it as a reg
        always @(ina or inb)
           begin
                if(ina == 1)
                    out = inb;
                else
                    out = ina;
           end
    endmodule
```

3. Solution 3

```
    module or_gate(ina, inb, out);
        input ina;
```

```
        input inb;
        output out; // declare it as reg
        if(ina == 1)    // put the if-else block inside an always
block
           out = 1;
        else
           out = inb;
    endmodule
```

4. Solution 4

```
    module adder(ina, inb, out);
         input [3:0] ina;
         input [3:0] inb;
         output [3:0] out;
         out = ina + inb;
    endmodule

    module tb_test();
        reg a, b, c; // inputs and output are of 4 bit, passing 1 bit
here
        // declare c as a wire instead of reg
        adder (a, b, c);    // missing a name for module instance
        initial
             begin
                 #0 a = 4'b0000; b = 4'b0001;
                 #5 a = 4'b0010; b = 4'b0001;
                 #10 a = 4'b1111; b = 4'b0001;
                 #15 $finish;
             end

        $monitor("a = %b, b = %b, c = %b", a, b, c);    // put monitor
statement inside an initial block
    endmodule
```