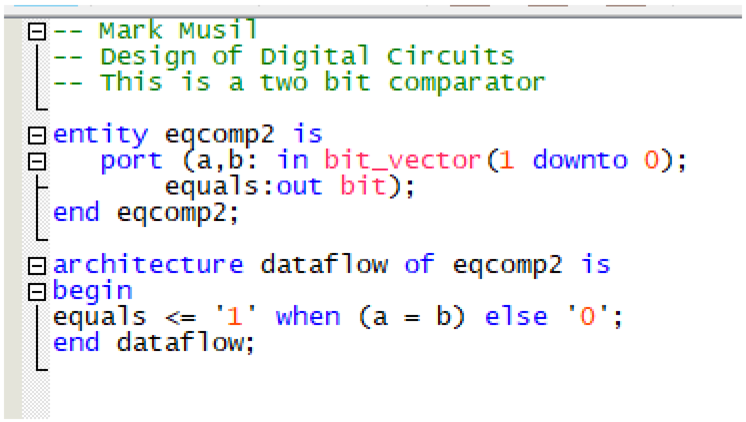
Mark Musil | Design of Digital Circuits | January 2022
Ohio University | Homework 3

Q1. Do question 3.1 (chapter 3).

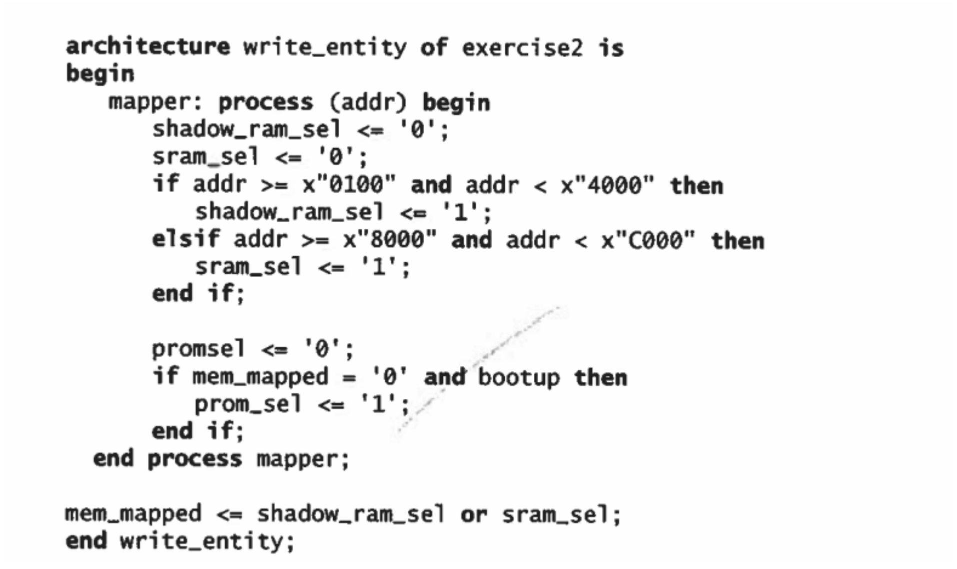
Write the entity declaration for a 2-bit equality comparator.



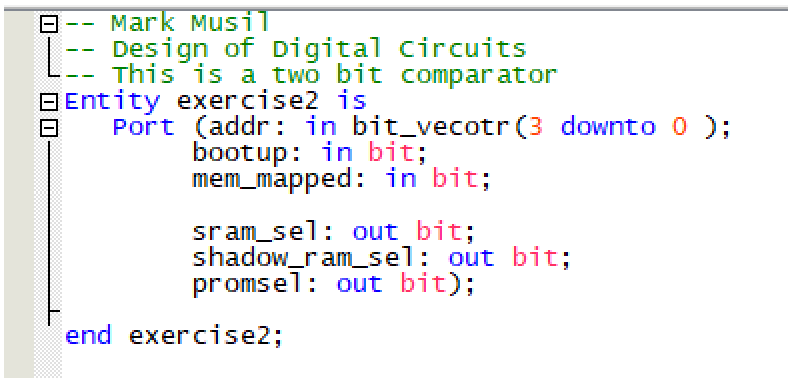
```
-- Mark Musil
-- Design of Digital Circuits
-- This is a two bit comparator

entity eqcomp2 is
   port (a,b: in bit_vector(1 downto 0);
        equals:out bit);
end eqcomp2;

architecture dataflow of eqcomp2 is
begin
equals <= '1' when (a = b) else '0';
end dataflow;
```

Q2. Do question 3.2 (chapter 3).

Write the entity declaration for the following architecture, assuming that all signals in the architecture are ports.



```
architecture write_entity of exercise2 is
begin
   mapper: process (addr) begin
      shadow_ram_sel <= '0';
      sram_sel <= '0';
      if addr >= x"0100" and addr < x"4000" then
         shadow_ram_sel <= '1';
      elsif addr >= x"8000" and addr < x"C000" then
         sram_sel <= '1';
      end if;

      promsel <= '0';
      if mem_mapped = '0' and bootup then
         prom_sel <= '1';
      end if;
  end process mapper;

mem_mapped <= shadow_ram_sel or sram_sel;
end write_entity;
```



```
-- Mark Musil
-- Design of Digital Circuits
-- This is a two bit comparator
Entity exercise2 is
   Port (addr: in bit_vecotr(3 downto 0 );
         bootup: in bit;
         mem_mapped: in bit;

         sram_sel: out bit;
         shadow_ram_sel: out bit;
         promsel: out bit);

end exercise2;
```

Mark Musil — Design of Digital Circuits — January 2022
Ohio University — Homework 3

Q3. Do question 3.3 (chapter 3).

Write an entity declaration for each of the TTL devices in table 2-1.

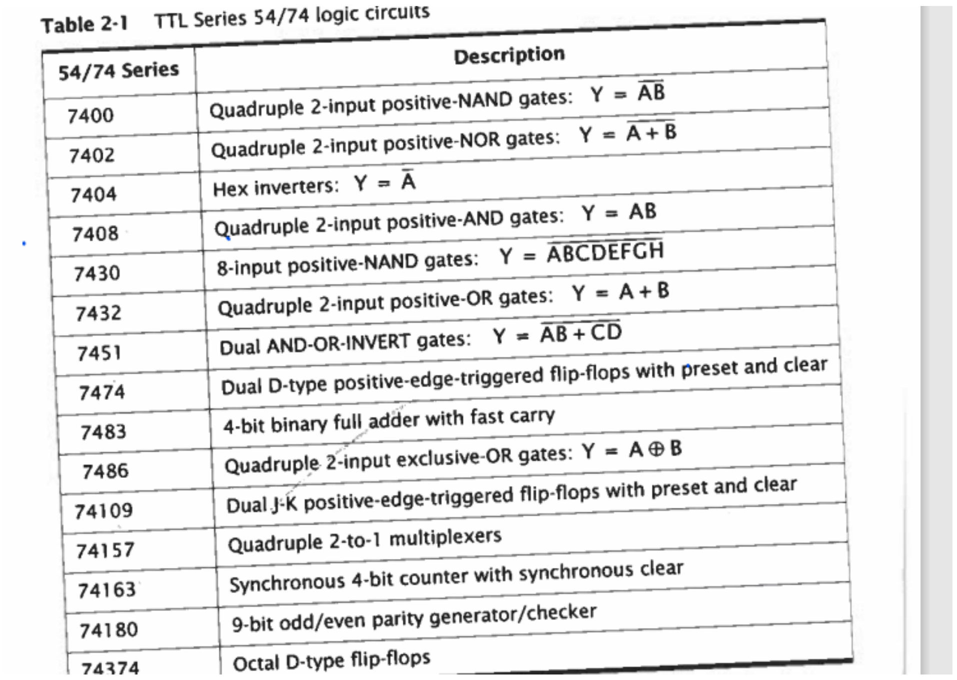
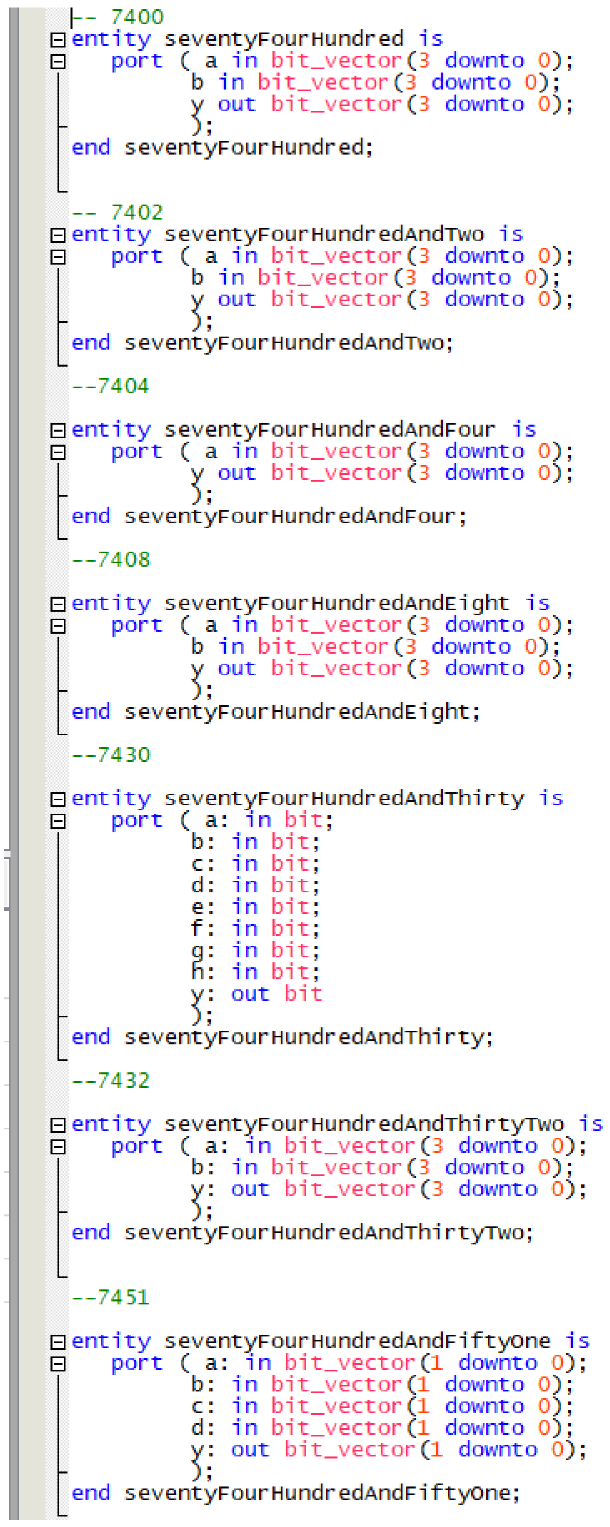


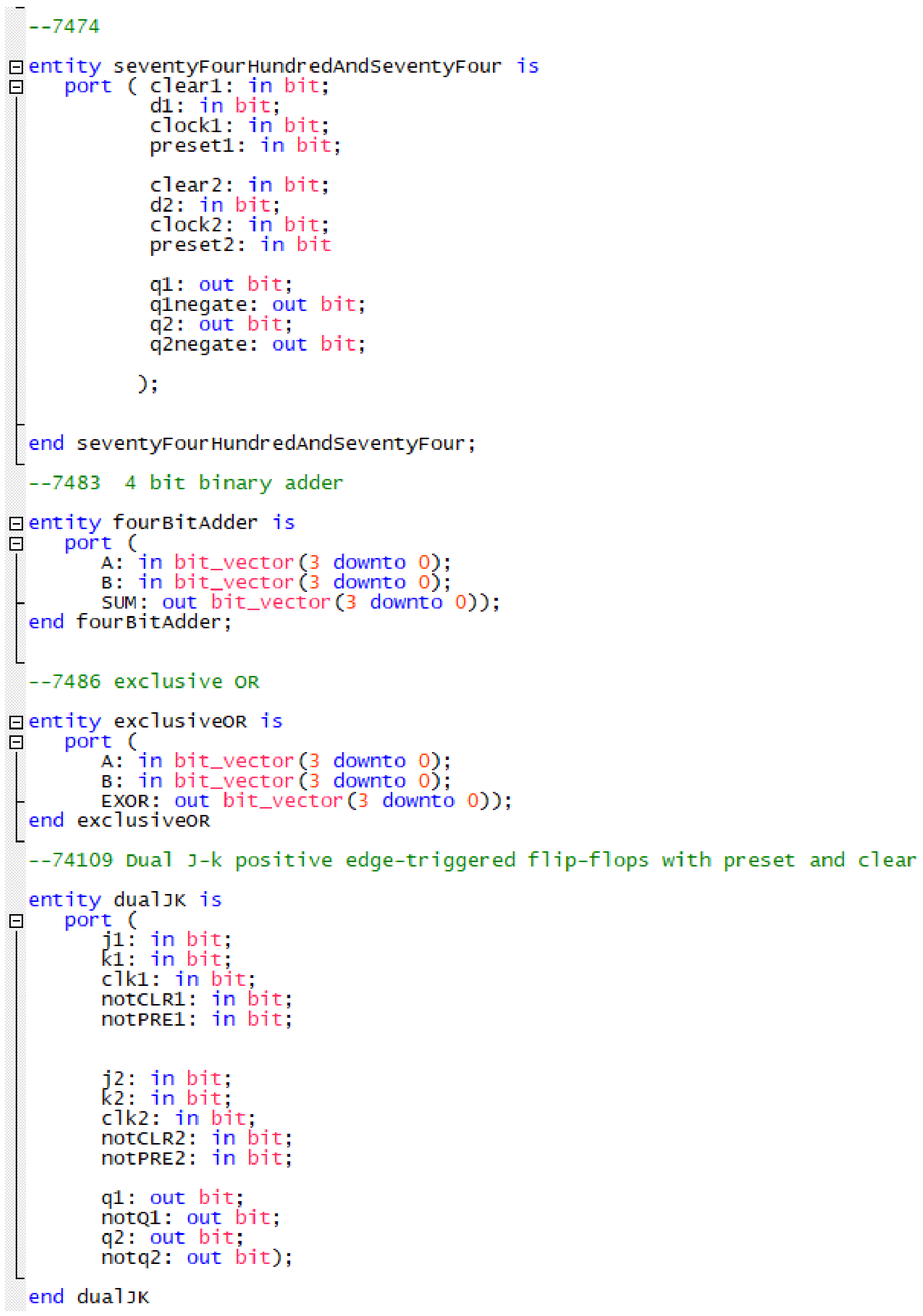
Table 2-1 TTL Series 54/74 logic circuits

54/74 Series	Description
7400	Quadruple 2-input positive-NAND gates: $Y = \overline{AB}$
7402	Quadruple 2-input positive-NOR gates: $Y = \overline{A + B}$
7404	Hex inverters: $Y = \overline{A}$
7408	Quadruple 2-input positive-AND gates: $Y = AB$
7430	8-input positive-NAND gates: $Y = \overline{ABCDEFGH}$
7432	Quadruple 2-input positive-OR gates: $Y = A + B$
7451	Dual AND-OR-INVERT gates: $Y = \overline{AB + CD}$
7474	Dual D-type positive-edge-triggered flip-flops with preset and clear
7483	4-bit binary full adder with fast carry
7486	Quadruple 2-input exclusive-OR gates: $Y = A \oplus B$
74109	Dual J-K positive-edge-triggered flip-flops with preset and clear
74157	Quadruple 2-to-1 multiplexers
74163	Synchronous 4-bit counter with synchronous clear
74180	9-bit odd/even parity generator/checker
74374	Octal D-type flip-flops

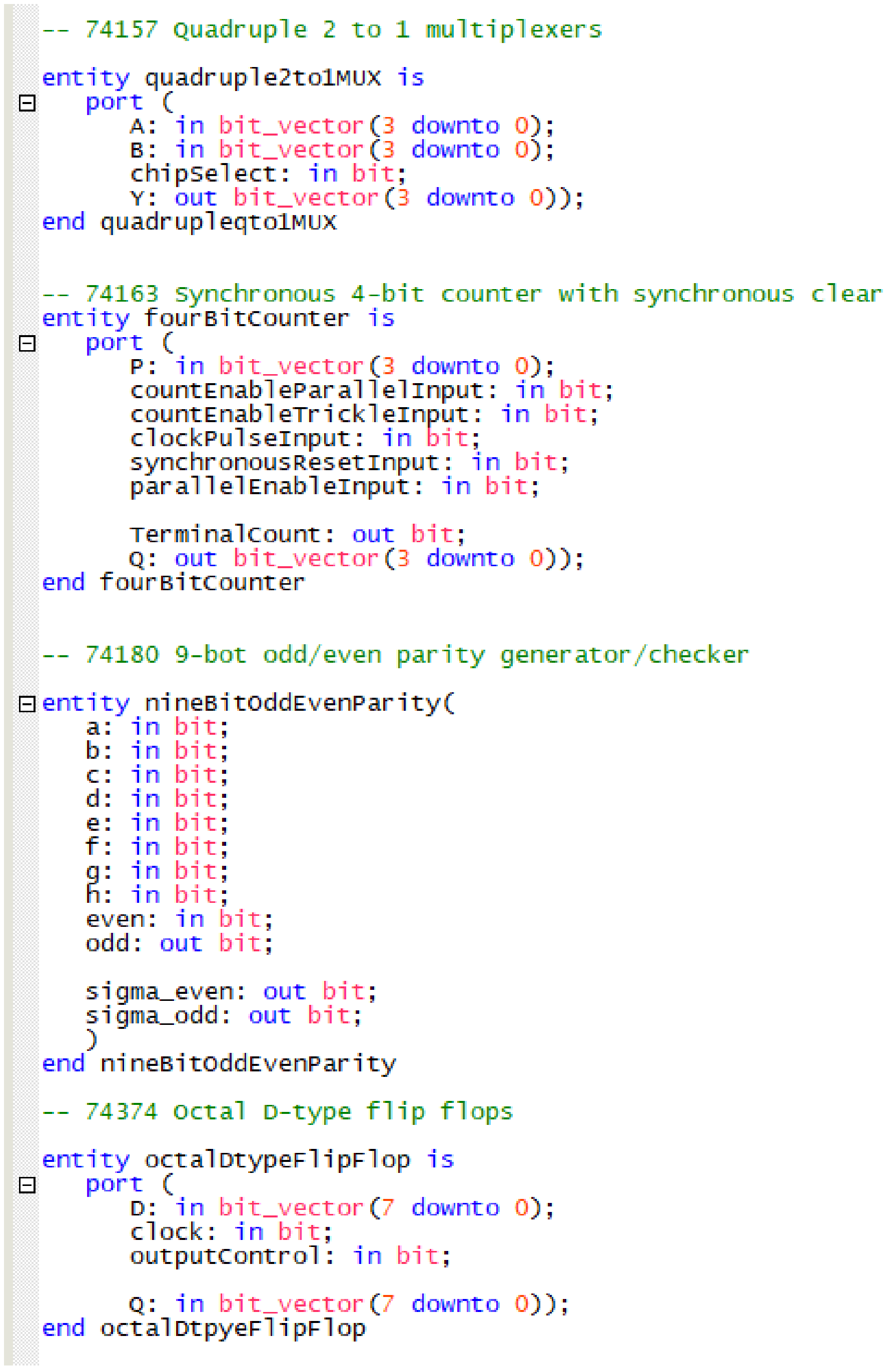


```vhdl
-- 7400
entity seventyFourHundred is
    port ( a in bit_vector(3 downto 0);
           b in bit_vector(3 downto 0);
           y out bit_vector(3 downto 0);
           );
end seventyFourHundred;


-- 7402
entity seventyFourHundredAndTwo is
    port ( a in bit_vector(3 downto 0);
           b in bit_vector(3 downto 0);
           y out bit_vector(3 downto 0);
           );
end seventyFourHundredAndTwo;

--7404

entity seventyFourHundredAndFour is
    port ( a in bit_vector(3 downto 0);
           y out bit_vector(3 downto 0);
           );
end seventyFourHundredAndFour;

--7408

entity seventyFourHundredAndEight is
    port ( a in bit_vector(3 downto 0);
           b in bit_vector(3 downto 0);
           y out bit_vector(3 downto 0);
           );
end seventyFourHundredAndEight;

--7430

entity seventyFourHundredAndThirty is
    port ( a: in bit;
           b: in bit;
           c: in bit;
           d: in bit;
           e: in bit;
           f: in bit;
           g: in bit;
           h: in bit;
           y: out bit
           );
end seventyFourHundredAndThirty;

--7432

entity seventyFourHundredAndThirtyTwo is
    port ( a: in bit_vector(3 downto 0);
           b: in bit_vector(3 downto 0);
           y: out bit_vector(3 downto 0);
           );
end seventyFourHundredAndThirtyTwo;


--7451

entity seventyFourHundredAndFiftyOne is
    port ( a: in bit_vector(1 downto 0);
           b: in bit_vector(1 downto 0);
           c: in bit_vector(1 downto 0);
           d: in bit_vector(1 downto 0);
           y: out bit_vector(1 downto 0);
           );
end seventyFourHundredAndFiftyOne;
```

Mark Musil | Design of Digital Circuits | January 2022
Ohio University | Homework 3



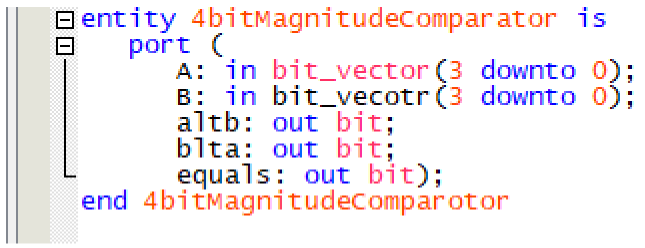
```
--7474

entity seventyFourHundredAndSeventyFour is
   port ( clear1: in bit;
          d1: in bit;
          clock1: in bit;
          preset1: in bit;

          clear2: in bit;
          d2: in bit;
          clock2: in bit;
          preset2: in bit

          q1: out bit;
          q1negate: out bit;
          q2: out bit;
          q2negate: out bit;

         );

end seventyFourHundredAndSeventyFour;
--7483  4 bit binary adder

entity fourBitAdder is
   port (
      A: in bit_vector(3 downto 0);
      B: in bit_vector(3 downto 0);
      SUM: out bit_vector(3 downto 0));
end fourBitAdder;


--7486 exclusive OR

entity exclusiveOR is
   port (
      A: in bit_vector(3 downto 0);
      B: in bit_vector(3 downto 0);
      EXOR: out bit_vector(3 downto 0));
end exclusiveOR
--74109 Dual J-k positive edge-triggered flip-flops with preset and clear

entity dualJK is
   port (
      j1: in bit;
      k1: in bit;
      clk1: in bit;
      notCLR1: in bit;
      notPRE1: in bit;


      j2: in bit;
      k2: in bit;
      clk2: in bit;
      notCLR2: in bit;
      notPRE2: in bit;

      q1: out bit;
      notQ1: out bit;
      q2: out bit;
      notq2: out bit);

end dualJK
```



```
-- 74157 Quadruple 2 to 1 multiplexers

entity quadruple2to1MUX is
    port (
        A: in bit_vector(3 downto 0);
        B: in bit_vector(3 downto 0);
        chipSelect: in bit;
        Y: out bit_vector(3 downto 0));
end quadrupleqto1MUX


-- 74163 Synchronous 4-bit counter with synchronous clear
entity fourBitCounter is
    port (
        P: in bit_vector(3 downto 0);
        countEnableParallelInput: in bit;
        countEnableTrickleInput: in bit;
        clockPulseInput: in bit;
        synchronousResetInput: in bit;
        parallelEnableInput: in bit;

        TerminalCount: out bit;
        Q: out bit_vector(3 downto 0));
end fourBitCounter


-- 74180 9-bot odd/even parity generator/checker

entity nineBitOddEvenParity(
    a: in bit;
    b: in bit;
    c: in bit;
    d: in bit;
    e: in bit;
    f: in bit;
    g: in bit;
    h: in bit;
    even: in bit;
    odd: out bit;

    sigma_even: out bit;
    sigma_odd: out bit;
    )
end nineBitOddEvenParity

-- 74374 Octal D-type flip flops

entity octalDtypeFlipFlop is
    port (
        D: in bit_vector(7 downto 0);
        clock: in bit;
        outputControl: in bit;

        Q: in bit_vector(7 downto 0));
end octalDtpyeFlipFlop
```

Q4. Do question 3.4 (chapter 3).

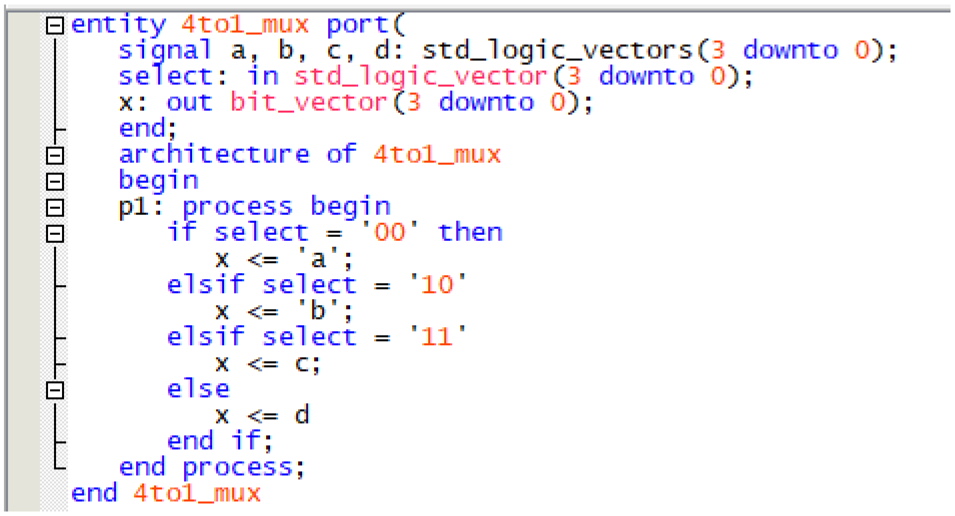
Write an entity declaration for a 4-bit magnitude comparator. Name the output port altb for "a less than b."



```
entity 4bitMagnitudeComparator is
    port (
        A: in bit_vector(3 downto 0);
        B: in bit_vecotr(3 downto 0);
        altb: out bit;
        blta: out bit;
        equals: out bit);
end 4bitMagnitudeComparotor
```

Q5. Do question 3.9 (chapter 3).

Identify the errors in the following code



```
entity 4to1_mux port(
    signal a, b, c, d: std_logic_vectors(3 downto 0);
    select: in std_logic_vector(3 downto 0);
    x: out bit_vector(3 downto 0);
    end;
    architecture of 4to1_mux
    begin
    p1: process begin
        if select = '00' then
            x <= 'a';
        elsif select = '10'
            x <= 'b';
        elsif select = '11'
            x <= c;
        else
            x <= d
        end if;
    end process;
end 4to1_mux
```

The following errors may be enumerated for the above snippet.

1. Port is not "closed" that is to say it is missing a closing parenthesis.
2. "std_logic_vectors" is not a valid VHDL keyword
3. The entity declaration is "closed" in the wrong location. "end 4to1_mux" should be after the port declarations
4. Begin is not matched to an end
5. "x <= d" is missing a terminating semicolon.