```
----------------------------------------------------------------------------------
-- Engineer: Olasubomi Borishade
--
-- Create Date: 02/04/2026 01:18:11 PM
-- Design Name: Comparison Logic Block
-- Module Name: Comparison_Logic_Block - Behavioral
-- Project Name: EENG 5560 Homework 2

-- Description: Logic block capable of performing 6 types of comparisons:
-- greater than, less than, equal to, greater than or equal to, less than or equal
to and not equal to
-- on two inputs A and B

-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------



library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use work.OPERATIONS_ARRAY_CUSTOM_PACK.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Comparison_Logic_Block is
    Generic (d_w: integer:= 3);

    Port ( A :  in STD_LOGIC_VECTOR (d_w - 1 downto 0); -- Declaring A as a input
vector
           B :  in STD_LOGIC_VECTOR (d_w - 1 downto 0); -- Declaring B as a input
vector
           OpSelC :  in CLB_Operation; -- Declaring the operation selection signal
as an input vector
           Y : out STD_LOGIC_VECTOR (d_w - 1 downto 0) -- Declaring Y as a vector
            );

end Comparison_Logic_Block;

architecture Behavioral of Comparison_Logic_Block is
```

```vhdl
begin
CLB_proc: process (A, B, OpSelC)
begin
    case OpSelC is
        when cGTH =>
            if A > B then -- Checks if input A is greater than input B
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;



        when cLTH =>
            if A < B then -- Checks if input A is less than input B
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;



        when cEQT =>
            if A = B then -- Checks if input A is equal to input B
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;



        when cGTET =>
            if A >= B then -- Checks if input A is greater than or equal to input B
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;



        when cLTET =>
            if A <= B then -- Checks if input A is less than or equal input B
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;



        when cNET =>
            if A /= B then -- Checks if input A is not equal to input B
```

```
                Y <= (others => '1'); -- If true, output is 1111
            else
                Y <= (others => '0'); -- If false, output is 0000
            end if;


        when NoOp => -- No operation
            Y <= (others => '0'); -- Returns all 0s


        when others =>
            Y <= (others => '0');

    end case;
end process;
end Behavioral;
```