☆ If Statement

* Syntax:

```
if booleanexpression then
    sequential statements
elsif boolean-expression then
    sequential statements
else
    sequential statement
end if;
```

☆☆ Examples:

*
```
if sum <= 100 then
    sum := sum + 10;
end if;
```

* if nickel_in then
    deposited <= total_10;
  elsif dime_in then
    deposited <= total_15;
  else
    deposited <= total_error;
  end if;

* if ctrl1 = '1' then
    if ctrl2 = '0' then
      mux_out <= "0010";
    else
      mux_out <= "0001";
    end if;
  else
    if ctrl2 = '0' then
      mux_out <= "1000";
    else
      mux_out <= "0100";
    end if;
  end if;

☆ Repeat all gates using if else statements

☆ Case Statement (Pg. 100)

* Syntax:

```
case expression is
    when choices => sequential statements
    when choices => sequential statements
    :
    when others => sequential statements
end case;
```

☆ Example: 4:1 Multiplexer

```
library ieee;
use ieee.std_logic_1164.all;
entity MUX_4_1 is
    port(a,b,c,d: in std_logic;
         s: in std_logic_vector(0 to 1);
         z: out std_logic);
end MUX_4_1;
architecture MUX_BEH of MUX_4_1 is
    constant MUX_Delay: time := 10 ns;
begin
    process(a,b,c,d,s)
    variable temp: std_logic;
    begin
        case s is
            when "00" => temp := a;
            when "01" => temp := b;
            when "10" => temp := c;
            when "11" => temp := d;
            when others => temp := 'X';
        end case;
        z <= temp after MUX_Delay;
    end process;
end MUX_BEH;
```

✱ Null Statement, (Pg. 102)

* Syntax: null;
* It's a seq. statement
* Does not cause any action to take place
* Can be used in if/case statement to explicitely specify that no action needs 2 b performed.

✱ Loop Statement

* Syntax:-

```
loop_label iteration_scheme
    loop
        seq_statements ;
        :          ;
    end loop loop_label;
```

* Three types of "iteration_schemes":
  1) for
  2) while
  3) neither of above

1) for

* Example:

```
factorial := 1;
for num in 2 to N
loop
    factorial := factorial * num;
end loop;
```

num → local variable, exists only inside for loop.

* If loop identifier has same name as a variable outside for loop then inside will use the for loop identifier

2) While

Ex:-

```
j:= 0;
sum := 10;
while j<20
loop
    sum := sum*2;
    j := j+3;
end loop;
```

3) No Condition

* loop can only break ~~then~~ using exit, next or return
* Eg:

```
j:= 0;
sum := 1;
loop
    j := j+21;
    sum := sum *10;
    exit when sum >100;
end loop;
```

EXIT Statement (Pg.104)

* Syntax:

```
exit loop_label when condition;
```

* Eg:

```
sum:= 1;
j:=0;
L3: loop
    j := j+21;
    sum := sum *10;
    if sum >100 then
        exit L3;
    end if;
end loop L3;
```

* Eg:
```
loop
    wait on A, B;
    exit when A = B;
end loop;
```
= wait until A = B;

✡ NEXT Statement

* Syntax: next loop_label when condition;
* next statement results in skipping the remaining statements in the current iteration of the specified loop.
* Eg:

```
for j in 10 downto 5
loop
    if sum < totalsum then
        sum := sum + 2;
    elsif sum = totalsum then
        next;
    else
        null;
    end if;
    k := k + 1; ----this statement gets skipped if
end loop;           next is executed.
```

✡ Generate Fibonacci Series
✡ Design a calculator using VHDL
✡ Design a 12 hour digital clock.