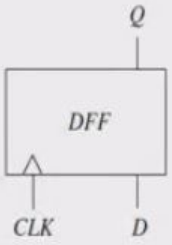
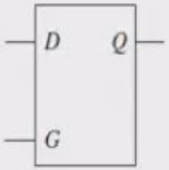
Modeling Flip-Flops Using Always Block

a.Verilog Code for Simple Flip Flop

b. Verilog code for Transparent Latch

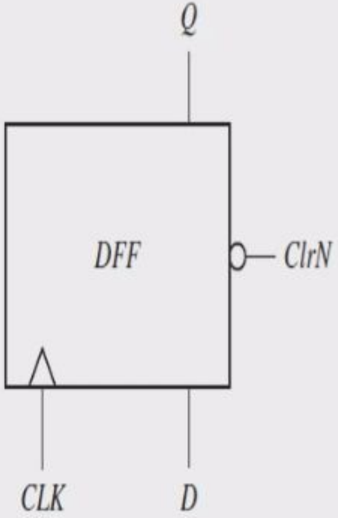


```
always @ (posedge CLK)
begin
  Q <= D;
end
```



```
always @ (G or D)
begin
 if (G)
   Q <= D;
end
```

Verilog code for DFF with Asynchronous clear



```
always @ (posedge CLK or negedge ClrN)
begin
  if (~ClrN)
    Q <= 0;
  else
    Q <= D;
end
```

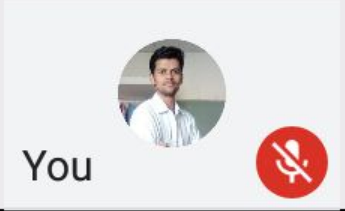
Basic form of if

```
if (condition)
  sequential statements1
else
  sequential statements2
```

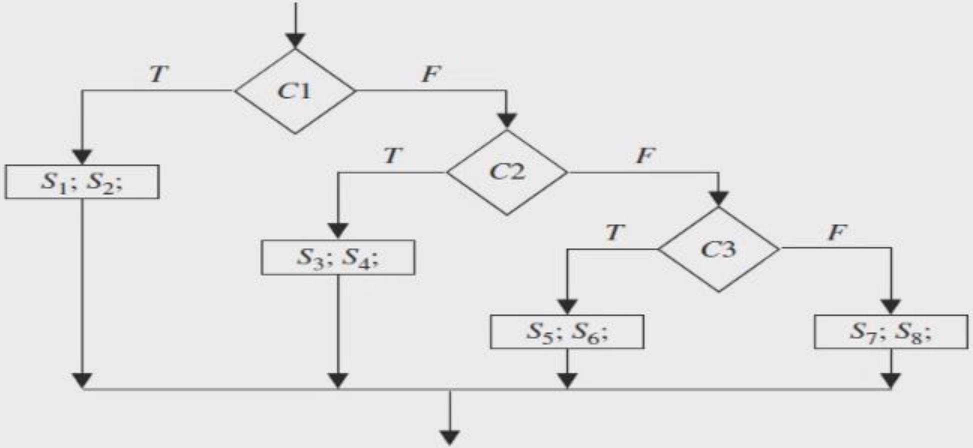
The condition is a Boolean expression that evaluates to TRUE or FALSE. If it is TRUE, `sequential statements1` are executed; otherwise, `sequential statements2` are executed.

Verilog **if** statements are sequential statements that can be used within an always block (or an initial block), but they cannot be used as concurrent statements outside of an always block. The most general form of the **if** statement is

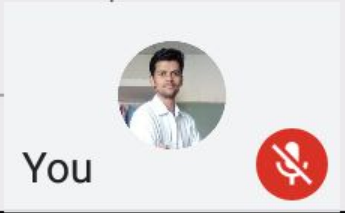
```
if (condition)
  sequential statements
  // 0 or more else if clauses may be included
else if (condition)
  sequential statements}
[else  sequential statements]
```



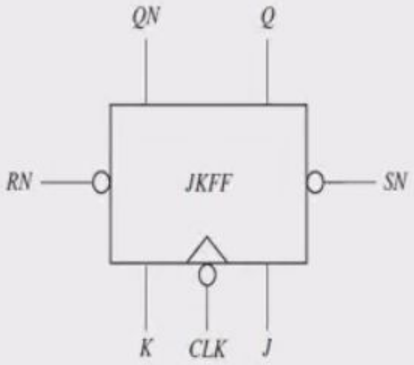
Nested if else loop



```
if (C1)
begin
  S1; S2;
end
else if (C2)
begin
  S3; S4;
end
else if (C3)
begin
  S5; S6;
end
else
begin
  S7; S8;
end
```



JK Flip Flop Verilog Code



```
module JKFF (SN, RN, J, K, CLK, Q, QN);
input  SN, RN, J, K, CLK;
output Q, QN;

reg Qint;

always @(negedge CLK or RN or SN)
begin
  if (~RN)
    #8  Qint <= 0;                                   // statement1
  else if (~SN)
    #8  Qint <= 1;                                   // statement2
  else
    Qint <= #10 ((J && ~Qint) || (~K && Qint)); // statement3
end

assign Q = Qint;                                     // statement4
assign QN = ~Qint;                                   // statement5

endmodule
```