# DAY-54
# #100DAYSRTL

## "System Verilog: Enumerations"

## Enumerations:-

- Need for variables that have a limited set of possible values that can be usually referred to by name.
- There's a specific facility, called an enumeration in System Verilog .
- Enumerated data types assign a symbolic name to each legal value taken by the data type.

**Syntax:-**

- enum <data-type> {<elements>} <name>;
- enum {IDLE, SETUP, ACCESS} state; // example 1
- enum bit[1:0] {IDLE, SETUP, ACCESS} state; // example 2

In the above example, the Enum type state has 3 elements, IDLE, SETUP and ACCESS. By default, the first element is Enum gets the value of 0 and the value increments for other following elements. Values can be manually assigned to Enum elements as shown below. If the value is only defined for the first element, the following elements get incremented value unless otherwise stated

Eg:-

- enum {IDLE=6, SETUP, ACCESS} state; // example 3 - IDLE=6, SETUP=7, ACCESS=8
- enum {IDLE=6, SETUP=9, ACCESS} state; // example 3 - IDLE=6, SETUP=9, ACCESS=10
- enum {IDLE=6, SETUP=9, ACCESS=14} state; // example 4 -IDLE=6,SETUP=9, ACCESS=14

**Some other important points:-**

1. Enums can be used as a data type only when the typedef keyword is used to define the enum. Otherwise, it's a simple variable.

2. Elements defined inside Enum can be referenced directly. It will return the value that it holds.
3. Elements having the same name cannot be defined inside 2 different enums. It is because the element of an enum has visibility in the entire block inside which the enum is defined.
   - enum bit[2:0] { RED , BLUE } color_1;
   - enum bit[2:0] { RED , GREEN } color_2; // Will show error
   - enum bit[2:0] { VOILET , GREEN } color_2; // Will work fine

If the value for an element is not mentioned explicitly it will automatically take the incremented value of the previous element for which the value is mentioned.

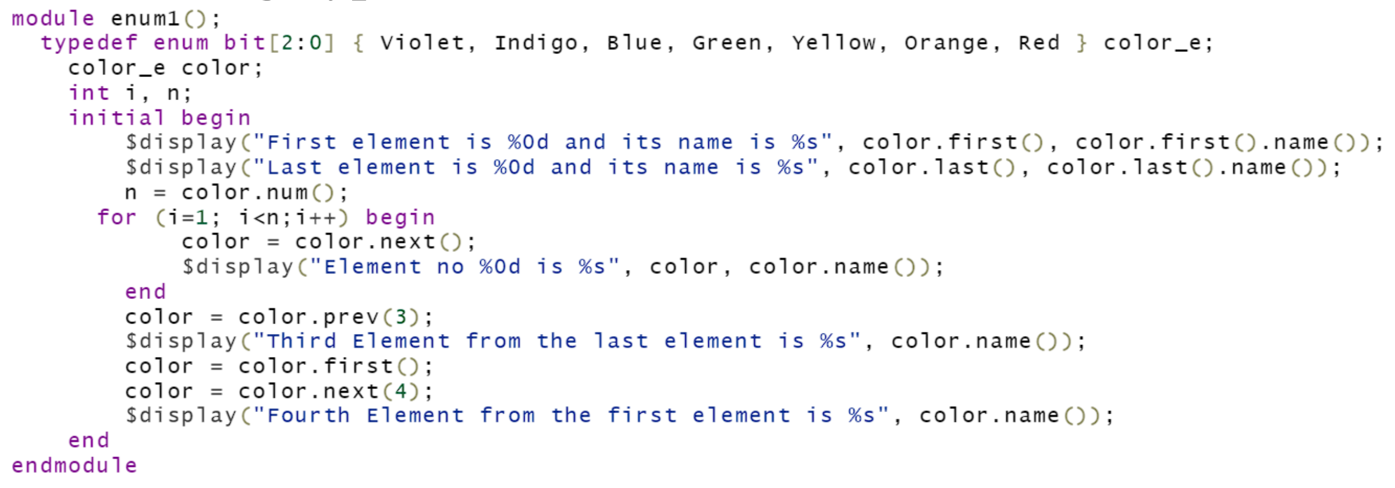
**Methods:-**

Enum provides various in-built system functions that can be used to iterate within elements of the Enum. Let's see these in-built functions in detail.

1. **first()** – return the value of the first element of the Enum
2. **last()** – returns the value of the last element of the Enum
3. **prev(int N)** – returns the value of the Nth element before the current element of Enum
4. **next(int N)** – returns the value of the Nth element after the current element of Enum
5. **num()** – returns the number of element in the Enum
6. **name()** – returns the name of the element

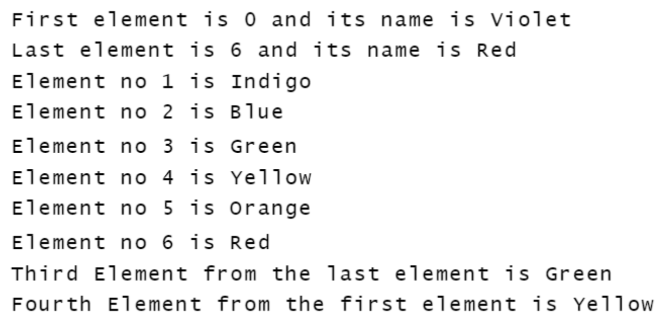
**Code Practising:-**

**Enum using Typedef:-**

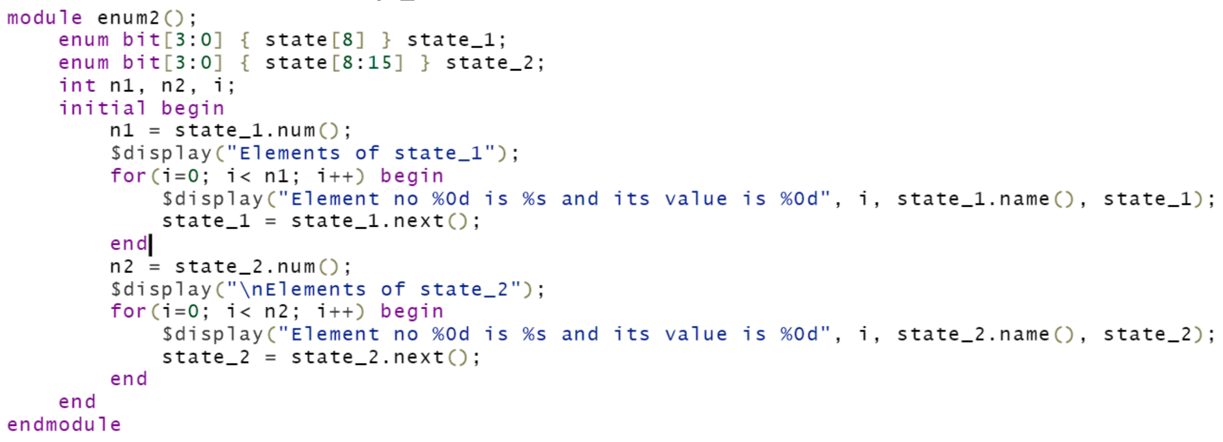


```
module enum1();
  typedef enum bit[2:0] { Violet, Indigo, Blue, Green, Yellow, Orange, Red } color_e;
    color_e color;
    int i, n;
    initial begin
        $display("First element is %0d and its name is %s", color.first(), color.first().name());
        $display("Last element is %0d and its name is %s", color.last(), color.last().name());
        n = color.num();
      for (i=1; i<n;i++) begin
            color = color.next();
            $display("Element no %0d is %s", color, color.name());
        end
        color = color.prev(3);
        $display("Third Element from the last element is %s", color.name());
        color = color.first();
        color = color.next(4);
        $display("Fourth Element from the first element is %s", color.name());
    end
endmodule
```

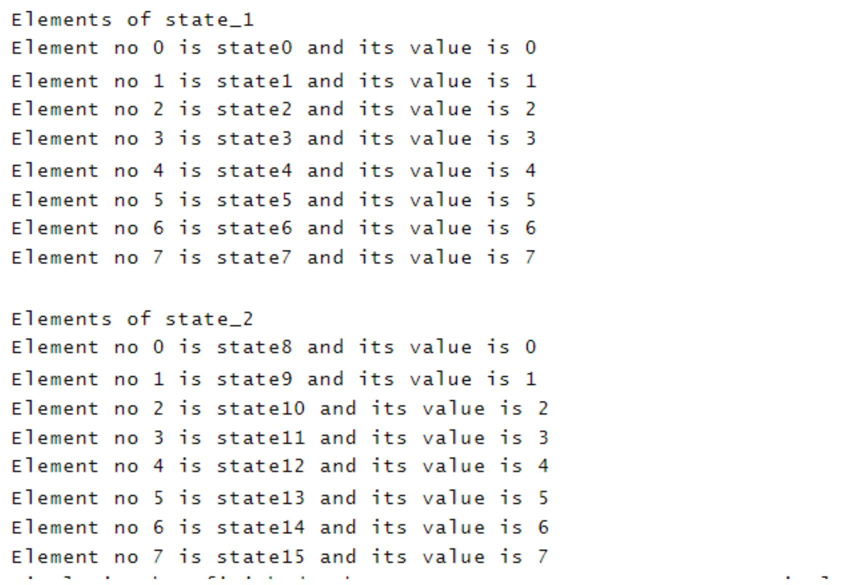
## Result:-


```
First element is 0 and its name is Violet
Last element is 6 and its name is Red
Element no 1 is Indigo
Element no 2 is Blue
Element no 3 is Green
Element no 4 is Yellow
Element no 5 is Orange
Element no 6 is Red
Third Element from the last element is Green
Fourth Element from the first element is Yellow
```

## Enum without Typedef:-


```
module enum2();
    enum bit[3:0] { state[8] } state_1;
    enum bit[3:0] { state[8:15] } state_2;
    int n1, n2, i;
    initial begin
        n1 = state_1.num();
        $display("Elements of state_1");
        for(i=0; i< n1; i++) begin
            $display("Element no %0d is %s and its value is %0d", i, state_1.name(), state_1);
            state_1 = state_1.next();
        end
        n2 = state_2.num();
        $display("\nElements of state_2");
        for(i=0; i< n2; i++) begin
            $display("Element no %0d is %s and its value is %0d", i, state_2.name(), state_2);
            state_2 = state_2.next();
        end
    end
endmodule
```

## Result:-


```
Elements of state_1
Element no 0 is state0 and its value is 0
Element no 1 is state1 and its value is 1
Element no 2 is state2 and its value is 2
Element no 3 is state3 and its value is 3
Element no 4 is state4 and its value is 4
Element no 5 is state5 and its value is 5
Element no 6 is state6 and its value is 6
Element no 7 is state7 and its value is 7

Elements of state_2
Element no 0 is state8 and its value is 0
Element no 1 is state9 and its value is 1
Element no 2 is state10 and its value is 2
Element no 3 is state11 and its value is 3
Element no 4 is state12 and its value is 4
Element no 5 is state13 and its value is 5
Element no 6 is state14 and its value is 6
Element no 7 is state15 and its value is 7
```