JavaScript-JS\JavaScript-SVB\Operators-JS\Operators-JS.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Operators - JS</title>
</head>
<body>
    <script>
        /*
        What are Operators?
        - Let's take an Example i.e. 2 + 5.
          Here,
            Example:-
                2 and 5 == Operands
                +       == Operator
            Here consider,
                a = 10
                b = 20
        ----------------------------------------------------------------------------------------
        - Types of Operators:-
              |_ 1. Arithematic Operators
              |_ 2. Increment & Decrement Operator
              |_ 3. Comparsion Operator
              |_ 4. Logical Operator
              |_ 5. Bitwise Operator
              |_ 6. Assignment Operator
              |_ 7. Miscellaneous Operators:-
                    |_ 1) Conditional Operator (? :)
                    |_ 2) typeof Operator
        ----------------------------------------------------------------------------------------
        1) Arithematic Operators:-
        JavaScript supports the following Arithmetic operators:
        - Addition ( + )
        - Subtraction ( - )
        - Multiplication ( * )
        - Division - ( / )
        - Modulus - (/)
        1) Addition(+)
        - Adds two operands
        - Ex:
            A + B will give 30
        2) Subtraction(-)
        - Subtracts the second operand from the first
        - Ex:
            A - B will give -10
        3) Multiplication(*)
        - Multiply both operands
        - Ex:
            A * B will give 200
        4) Division(/)
        - Divide the numerator by the denominator
        - Ex:
            B / A will give 2
        5) Modulus(%)
        - Outputs the remainder of an integer division
        - Ex:
            B % A will give 0
        */
        console.log("Arithematic Operator: ")
        a = 10
        b = 20
        console.log("Addition a + b: ", a + b)
        console.log("Subtraction a - b: ", a - b)
        console.log("Multiplication a * b: ", a * b)
```

```
        console.log("Division a / b: ", a / b)
        console.log("Modulus a % b: ", a % b)
        // ------------------------------------------------------------------------------------------

        /*
        2) Increment and Decrement Operators:
        JavaScript has an increment and a decrement operator.
        They modify a variable in place./ ++ adds one to a number,
        -- subtracts one from a number.
        1) Increment (++):-
            - Postfix - Increment( a++ )
            - Prefix - Increment( ++a )
        2) Decrement (--):
            - Postfix - Decrement( a++ )
            - Prefix - Decrement( ++a )
        :- Using ++/-- After the Operand
           - When you use the increment/decrement operator after the operand,
           - the value will be returned before the operand is increased/
           decreased.
        :- Using ++/-- Before the Operand
            - If you'd rather make the variable increment/decrement before
            returning, you simply have to use the increment/decrement operator
            before the operand.
        */
        // :- Using ++/-- After the Operand
        let s = 10
        console.log(s++)
        console.log(s)
        let x = 10
        console.log(x--)
        console.log(x)
        // :- Using ++/ -- Before the Operand
        let p = 10
        console.log(++p)
        console.log(++p)
        let r = 10
        console.log(++r)
        console.log(++r)
        // ------------------------------------------------------------------------------------------

        /*
        3) Comparsion Operator:
        - JavaScript comparison operators are essential tools for
        checking conditions and making decisions in your code.
        - They are used to evaluate whether a condition is true or
        false by comparing variables or values.
        - These operators
        play a crucial role in logical expressions, helping to
        determine the equality or differences between values.
        - Types of  Comparsion Operators are:-
        ------------------------------------------------------------------------------
        |   Operator Name        |        Usage          |     Operation              |
        ------------------------------------------------------------------------------|
        |   Equality Operaters   |        a == b         |  Compare the Equality of   |
        |                        |                       |  2 Operators.              |
        ------------------------------------------------------------------------------|
        |   Inequality           |        a != b         |  Compares Inequality of    |
        |   Operator             |                       |   two Operators            |
        ------------------------------------------------------------------------------|
        |   Strict Equality      |        a === b        |  Compares both the value   |
        |   Operator             |                       |  and Data type of          |
        |                        |                       |  the operand               |
        ------------------------------------------------------------------------------|
        |   Stricy Inequality    |        a !== b        | Compares inequality        |
        |   Operatotr            |                       | with type                  |
        |                        |                       |                            |
        ------------------------------------------------------------------------------|
```

```
        |   Greator Than        |        a > b          | Checks if the left         |
        |   Operator            |                       | operator is greater than   |
        |                       |                       | the right operator         |
        ---------------------------------------------------------------------------|
        |   Less Than           |        a < b          | Checks if the left         |
        |   Operator            |                       | operator is smaller than   |
        |                       |                       | the right operator         |
        |--------------------------------------------------------------------------|
        |   Greater than        |        a >= b         |  Checks if the left        |
        |   Or Equal to         |                       |  operator is greater than  |
        |   Operator            |                       |  or equal to the right     |
        |                       |                       |  operator                  |
        |--------------------------------------------------------------------------|
        |   Less than           |        a <= b         |  Checks if the left        |
        |   Or Equal to         |                       |  operator is smaller than  |
        |   Operator            |                       |  or equal to the right     |
        |                       |                       |  operator                  |
        |--------------------------------------------------------------------------|
        */
        let d = 10
        let f = 20
        /*
        1) Equality Operaters
        - Compare the Equality of 2 Operators.
        */
       console.log("Equality Operaters")
       console.log(d == f)

        /*
        2) Inequality Operators:
        - Compares Inequality of  two Operator
        */
        console.log("Inequality Operaters")
        console.log(d != f)

        /*
        3) Strict Equality Operator
        - Compared both the value and the data type of the operand
        */
        console.log("Strict Equality Operators")
        console.log(d === f)

        /*
        4) Strict Inequality Operator
        - Compare Inequality with it's Type
        */
        console.log("Strict Inquality Operators")
        console.log(d === f)

        /*
        5) Greater than Operator:
        - Check if the Left Operator is greater than the Right Operator
        */
        console.log("Strict Inquality Operators")
        console.log(d > f)

        /*
        6) Less than Operator:
        - Check if the Left Operator is smaller than the Right Operator
        */
        console.log("Strict Inquality Operators")
        console.log(d < f)

        /*
        7) Greater than or Equal to Operator:
        - Check if the left operator is greater than or equal to the right operator
        */
```

```
        console.log("Greater than or Equal to Operators")
        console.log(d >= f)

        /*
        8) Less Than or Equal to Operator:-
        - Check if the left operator is smaller than or equal to the right operator
        */
        console.log("Smaller than or Equal to Operators")
        console.log(d >= f)
        // ------------------------------------------------------------------------------------

        /*
        4) Logical Operators:-
        -  Logical operators return a boolean value by evaluating boolean expressions.
        - For example,
        */
        console.log("Logical Operators...")
        console.log("Logical Operators Example")
        const num1 = 3;
        const num2 = 5;
        console.log(num2 < 5 && num1 < 6);
        /*
        Logical Operators:-
        ------------------------------------------------------------------------------
        | Name of the Logical Operator              |                    Symbol          |
        ------------------------------------------------------------------------------
        |  AND Operator                             |            &&                      |
        ------------------------------------------------------------------------------
        |  OR Operator                              |            ||                      |
        ------------------------------------------------------------------------------
        |  NOT Operator                             |             !                      |
        ------------------------------------------------------------------------------
        */
        /*
        1) AND Operator:-
        - The logical AND operator && returns true if both the expressions are true.
        - That is:-
        ------------------------------------------------------------------------------
        |         true              &&         true             =>          true          |
        ------------------------------------------------------------------------------
        |         true              &&         false            =>          false         |
        ------------------------------------------------------------------------------
        |         false             &&         true             =>          false         |
        ------------------------------------------------------------------------------
        |         false             &&         false            =>          false         |
        ------------------------------------------------------------------------------
        - For example,
        */
        console.log("AND Operator");
        console.log(2 < 4 && 4 > 2);
        /*
        2) OR Operator:-
        - The logical OR operator || returns true if at least one expression is true.
        - That is:-
        ------------------------------------------------------------------------------
        |         true              &&          true            =>          true          |
        ------------------------------------------------------------------------------
        |         true              &&          false           =>          true          |
        ------------------------------------------------------------------------------
        |         false             &&          true            =>          true          |
        ------------------------------------------------------------------------------
        |         false             &&          false           =>          false         |
        ------------------------------------------------------------------------------
        - For example,
        */
        console.log("OR Operator");
        console.log(2 < 4 || 4 > 2);
```

```
        /*
        3) NOT Operator:-
        - The logical NOT operator ! returns true if the specified expression is false and
        vice versa.
        - That is:-
        ------------------------------------------------------------------------------------
        |       !(true)                        =               false                       |
        ------------------------------------------------------------------------------------
        |       !(flase)                       =               true                        |
        ------------------------------------------------------------------------------------
        - For example,
        */
        console.log("NOT Operator");
        console.log(!(2 > 4));
        // ------------------------------------------------------------------------------------

        /*
        5) Bitwise Operator:-
        - JavaScript stores numbers as 64 bits floating point numbers, but all bitwise operations
        are performed on 32 bits binary numbers.
        - Before a bitwise operation is performed, JavaScript converts numbers to 32 bits signed
        integers.
        - After the bitwise operation is performed, the result is converted back to 64 bits
        JavaScript numbers.
        - There are 7 Bitwise Operator:-
        1) AND - &
        2) OR  - |
        3) NOT - ~
        4) XOR - ^
        5) Zero Fill Left Shift  -  <<
        6) Zero Fill Right Shift -  >>>
        7) Signed Right Shift    -  >>
        */
        /*
        1) AND - &
        - When a bitwise AND is performed on a pair of bits, it returns 1 if both bits are 1.
        - That is:-
        ------------------------------------------------------------------------------------
        |          1    &     1                       =               1                     |
        ------------------------------------------------------------------------------------
        |          1    &     0                       =               0                     |
        ------------------------------------------------------------------------------------
        |          0    &     1                       =               0                     |
        ------------------------------------------------------------------------------------
        |          0    &     0                       =               0                     |
        ------------------------------------------------------------------------------------
        - Example:-
        */
        console.log("BitWise Operator");
        console.log("BitWise AND Operator");
        console.log(5 & 4);
        /*
        2) BitWise OR:-
        - When a bitwise OR is performed on a pair of bits, it returns 1 if one of the bits is 1.
        - That is:-
        ------------------------------------------------------------------------------------
        |          1    |     1                       =               1                     |
        ------------------------------------------------------------------------------------
        |          1    |     0                       =               1                     |
        ------------------------------------------------------------------------------------
        |          0    |     1                       =               1                     |
        ------------------------------------------------------------------------------------
        |          0    |     0                       =               0                     |
        ------------------------------------------------------------------------------------
        - Example:-
        */
        console.log("BitWise OR Operator");
```

```
        console.log(5 | 4);
        /*
        3) NOT Bitwise Operator:-
        - In NOT Bitwise, the o is converted to 1 and 1 is converted to 0.
        - The NOT operator is often written as a tilde character ("~")
        -
        - That is:-
        ------------------------------------------------------------------------------------
        |           ~ 0                     =                 1                            |
        ------------------------------------------------------------------------------------
        |           ~ 1                     =                 0                            |
        ------------------------------------------------------------------------------------
        - Example:-
        */
        console.log("BitWise NOT Operator");
        console.log(~ 5);
        // Wait of Sir to teach....
        /*
        4) XOR BitWise Operator:-
        - Bitwise XOR ^ returns 1 if the corresponding bits are different and returns 0 if
        the corresponding bits are the same.
        - That is:-
        ------------------------------------------------------------------------------------
        |         1    ^    1                    =               0                     |
        ------------------------------------------------------------------------------------
        |         1    ^    0                    =               1                     |
        ------------------------------------------------------------------------------------
        |         0    ^    1                    =               1                     |
        ------------------------------------------------------------------------------------
        |         0    ^    0                    =               0                     |
        ------------------------------------------------------------------------------------
        - Example:-
        */
        console.log("BitWise XOR Operator");
        console.log(5 ^ 4);
        /*
        5) Zero Fill Left Shift  -  <<
        - In the left shift operator <<, the left operand specifies the number and the right
        operand specifies the number to be shifted left. Zero bits are added to the right and
        excess bits from the left are discarded.
        - Example:-
        */
        console.log("BitWise Zero Fill Left Shift Operator");
        let myVar = 20;
        let myVar1 = 3;
        let result = myVar << myVar1;
        console.log(result);
        /*
        6) Zero Fill Right Shift -  >>>
        - Zero-fill right shift >>> shifts the operand to the right by filling
        the zero bits to the left. Excess bits from the right are discarded.
        - This operator shifts the first operand the specified number of bits to
        the right. Excess bits shifted off to the right are discarded. Zero bits
        are shifted in from the left.
        - Example:-
        */
        console.log("Zero Fill Right Shift Operator");
        console.log(myVar >>> myVar1);
        /*
        7) Signed Right Shift  -  >>
        -  This operator shifts the first operand the specified number of bits
        to the right. Excess bits shifted off to the right are discarded.
        Copies of the leftmost bit are shifted in from the left. Since the new
        leftmost bit has the same value as the previous leftmost bit, the sign
        bit (the leftmost bit) does not change. Hence the name "sign-propagating".
        - Example:-
        */
```

```
        console.log("Signed Right Shift Operator");
        console.log(9 >> 2);
        /*
             9 (base 10): 00000000000000000000000000001001 | (base 2)
                          --------------------------------
        9 >> 2 (base 10): 00000000000000000000000000000010 | (base 2) = 2 (base 10)
        */
        // ------------------------------------------------------------------------------------

        /*
        6) Assignment Operators:-
        - Assignment operators are used to assign values to variables in JavaScript.
        - Syntax:
                data=value
        - Example:-
        */
        console.log("Assignment Operators in JavaScript");
        let box1 = 10;
        let box2 = 20;
        box1 = box2;
        box2 = box1;
        console.log(box1);
        console.log(box2);
        /*
        - Types of Assignment Operators:-
        ------------------------------------------------------------------------------------
        | OPERATOR NAME                |       SHORTHAND OPERATOR        |            MEANING  |
        ------------------------------------------------------------------------------------
        | Addition Assignment          |           a +=  b               |        a = a+b     |
        ------------------------------------------------------------------------------------
        | Subtraction Assignment       |           a -=  b               |        a = a-b     |
        ------------------------------------------------------------------------------------
        | Multiplication Assignment    |           a *=  b               |        a = a*b     |
        ------------------------------------------------------------------------------------
        | Division Assignment          |           a /=  b               |        a = a/b     |
        ------------------------------------------------------------------------------------
        | Remainder Assignment         |           a %=  b               |        a = a%b     |
        ------------------------------------------------------------------------------------
        | Exponentiation Assignment    |           a **= b               |        a = a**b    |
        ------------------------------------------------------------------------------------
        */
        let val1 = 10;
        // 1) Addition Assignment:-
        let val2 = 5;
        console.log("Addition Assignment Operator")
        console.log(val1 += val2);
        // 2) Subtraction Assignment:-
        console.log("Subtraction Assignment Operator")
        console.log(val1 -= val2);
        // 3) Multiplication Assignment:-
        console.log("Multiplication Assignment Operator")
        console.log(val1 *= val2);
        // 4) Division Assignment:-
        console.log("Division Assignment Operator")
        console.log(val1 /= val2);
        // 5) Remainder Assignment:-
        console.log("Remainder Assignment Operator")
        console.log(val1 %= val2);
        // 6) Exponentiation Assignment:-
        console.log("Exponentiation Assignment Operator")
        console.log(val1 **= val2);
        // ------------------------------------------------------------------------------

        /*
        7) Miscellaneous Operators:-
        1) Conditional Operator (? :)
        2) typeof Operator
```

```
*/
/*
1) Conditional Opertor(? :)
- The JavaScript Ternary Operator, also known as the Conditional Operator, offers a better
approach to expressing conditional (if-else) statements. It operates on three operands: a
condition, a value to return if the condition is true, and a value to return if the
condition is false.
- Syntax:-
        ------------------------------------------------
       |  condition ? trueExpression : falseExpression  |
        ------------------------------------------------
- Operands:-
    ------------------------------------------------------------------------------------------
    |    Name                 |                Description                                   |
    ------------------------------------------------------------------------------------------
    |    Condition            |   Expression to be evaluated which returns a boolean value   |
    ------------------------------------------------------------------------------------------
    |   Value if True         |   Value to be executed if the condition results in a true    |
    |                         |   state                                                      |
    ------------------------------------------------------------------------------------------
    |    Value if False       |  Value to be executed if the condition results in a false    |
    |                         |  state                                                       |
    ------------------------------------------------------------------------------------------
- Characteristics of Ternary Operator:-
1) The expression consists of three operands:the condition, value if true, and value if
false.
2) The evaluation of the condition should result in either a true/false or a boolean value.
3) The true value lies between “?” & “:” and is executed if the condition returns true.
Similarly, the false value lies after “:” and is executed if the condition returns false.
*/
console.log("Conditional Opertor/ Ternary Operator")
let age = 13;
age < 18 ? console.log("Minor") : console.log("Adult");
// ------------------------------------------------------------------------------
// ------------------------------------------------------------------------------

/*
2) typeof Operator:-
- The JavaScript typeof operator returns the data type of a variable or expression. It’s
a unary operator placed before its operand and returns a string indicating the data type,
such as “number”, “string”, “boolean”, “object”, “undefined”, “function”, or “symbol”.
It’s commonly used to dynamically determine the type of data being processed, facilitating
conditional logic and type checking in JavaScript programs.
- Syntax:
        typeof operand
            // OR
        typeof (operand)
- Here is a list of the return values for the typeof Operator:-
--------------------------------------------------------------------------------
|    Type                         |                  String Returned by typeof      |
================================================================================
|   Number                        |                        "number"                 |
--------------------------------------------------------------------------------
|    String                       |                         "string"                |
--------------------------------------------------------------------------------
|   Boolean                       |                         "boolean"               |
--------------------------------------------------------------------------------
|   Object                        |                         "object"                |
--------------------------------------------------------------------------------
|   Function                      |                         "function"              |
--------------------------------------------------------------------------------
|   Undefined                     |                         "undefined"             |
--------------------------------------------------------------------------------
|   Null                          |                         "object"                |
--------------------------------------------------------------------------------
- Example:-
*/
console.log("typeof Operator In JavaScript")
```

```
        let str = "Sony";
        let num = 2003;
        let bool = true;
        let dummy = null;
        let undefine = undefined;
        let unInitial;
        // Type of a variable is "undefined" when a variable is only declared...
        let obj = [];
        let obj1 = unInitial;
        let fun = function done(){};
        console.log(typeof(str));
        console.log(typeof(num));
        console.log(typeof(bool));
        console.log(typeof(dummy));
        console.log(typeof(undefine));
        console.log(typeof(obj));
        console.log(typeof(obj1));
        console.log(typeof(fun));
        // ------------------------------------------------------------------------------
    </script>
</body>
</html>
```

BitWise - Operator

Zero Fill Left Shift Operator

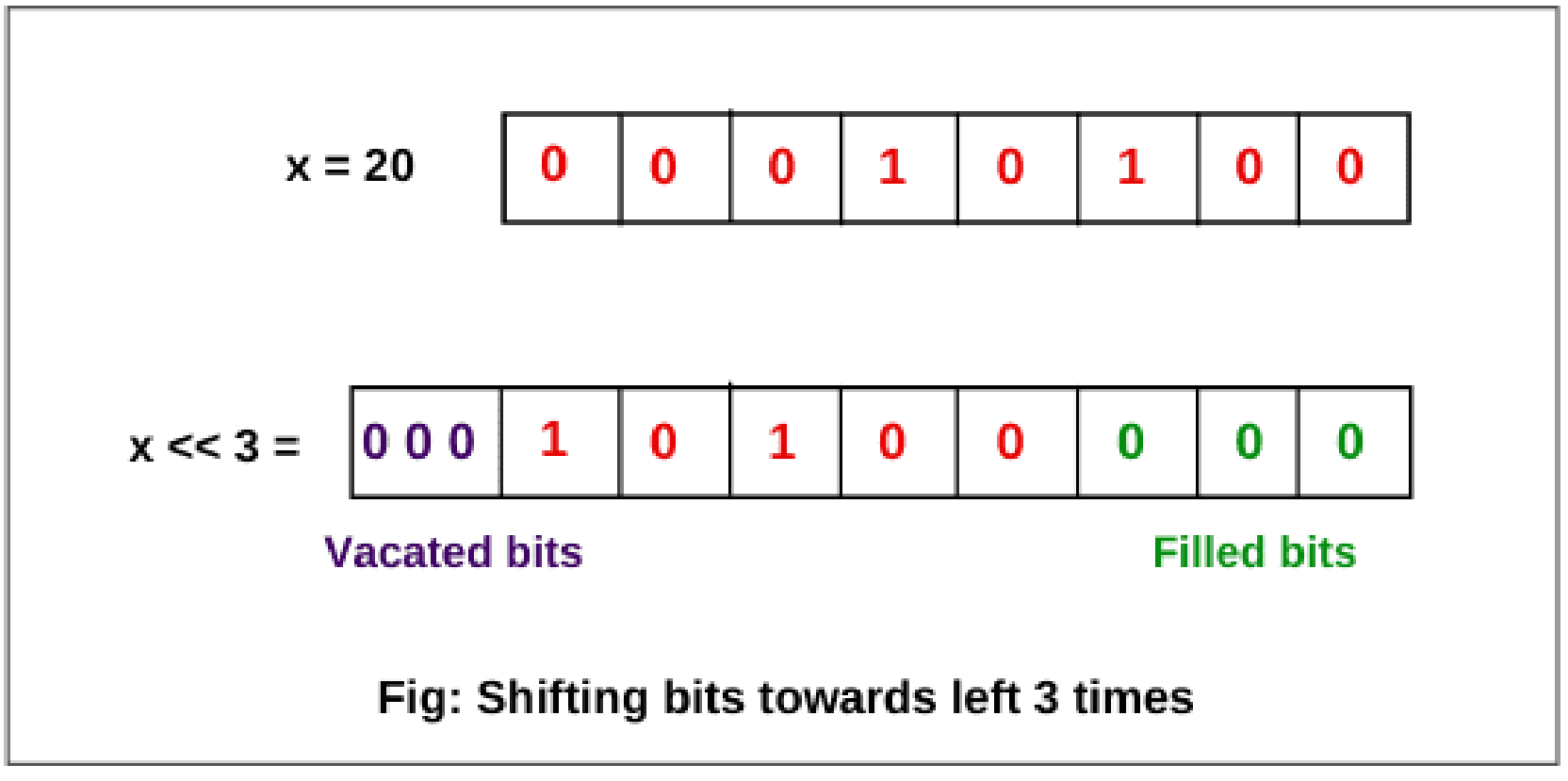

Fig: Shifting bits towards left 3 times