JavaScript

BAGHEL COMPUTER CENTRE

Index

What is JavaScript?..... 3
Comparison between JavaScript and Java:.. 6
How can apply.....7
Print hello world..... 8
What is variable and data?.....9
Rules to declare a variable..... 25

BAGHEL COMPUTER CENTRE

What is JavaScript?

JavaScript is a popular programming and scripting language used to make web pages interactive.

JavaScript is a scripting language because it is mainly used to control web pages by adding interactive features like buttons, animations, and pop-ups.

It is also a programming language because it can handle complex tasks, store data, and even run on servers using Node.js.

BAGHEL COMPUTER CENTRE

5 Easy Features of JavaScript

1. **Makes Websites Interactive** – Adds animations, buttons, and effects.
2. **Runs in Web Browsers** – No need to install extra software.
3. **Handles User Actions** – Responds to clicks, typing, and mouse movements.
4. **Updates Web Content Dynamically** – Changes text, images, and styles without reloading the page.
5. **Works with HTML & CSS** – Controls webpage layout and design.

BAGHEL COMPUTER CENTRE

5 Important Uses of JavaScript

1. **Web Development** – Powers interactive websites.
2. **Form Validation** – Checks if users enter correct details (e.g., email, password).
3. **Games & Animations** – Creates fun web-based games.
4. **Mobile & Web Apps** – Used in apps like WhatsApp Web and Instagram.
5. **Server-Side Development** – Works with Node.js to build backend systems.

Founder & Date:

- **Founder:** Brendan Eich
- **Year:** 1995

BAGHEL COMPUTER CENTRE

Comparison between JavaScript and Java:

JavaScript	Java
Used to make websites interactive.	Used to create big software and mobile apps.
Runs inside a web browser.	Runs on computers, mobiles, and servers.
Easier to learn and write.	More complex and requires setup.
Interpreted language (executes line by line).	Compiled language (converted into machine code before running).
Mainly used for web development.	Used for many types of software, including Android apps.

BACHEL COMPUTER CENTRE

How can apply

Internal

-inside head tag

-inside body tag

-inside both head and body tag

External

-save file with .js extension.

-For link javascript file with html use.

```
<script src="file.js"></script>
```

BAGHEL COMPUTER CENTRE

Print hello world

Input

```
<!DOCTYPE html>
<html>
<head>
    <title>js</title>
</head>
<body>
<script type="text/javascript">document.write("hello world")
</script>
</body>
</html>
```

Output

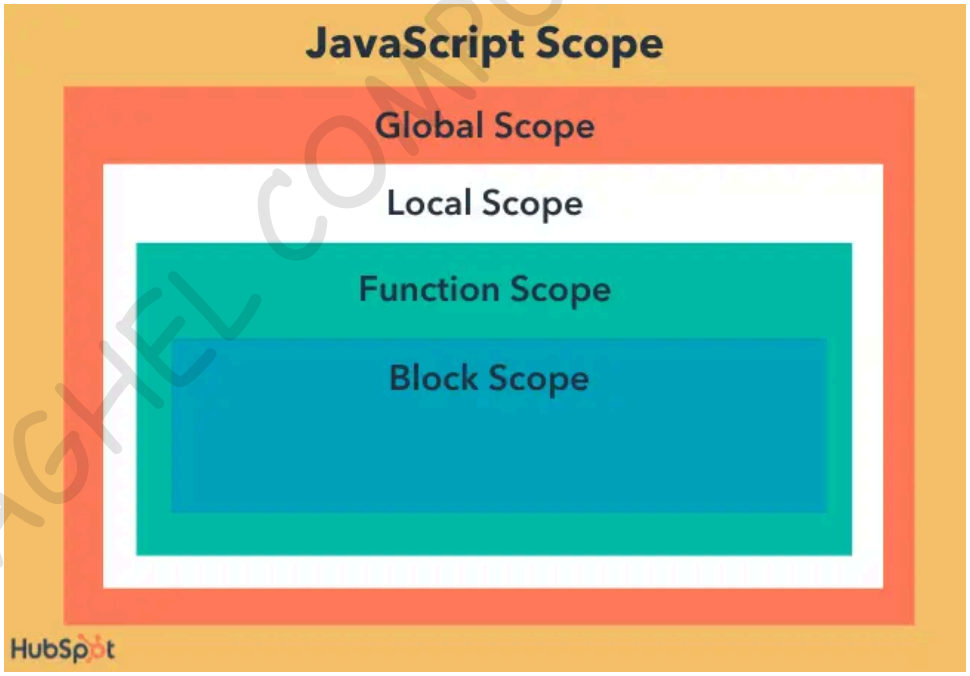
hello world

BAGHEL COMPUTER CENTRE

What is variable and data?

1. Variable

A variable is like a container that holds a value. You can think of it as a labeled box where you store information. We use variables to save data that we want to use or change later in the program.



BAGHEL COMPUTER CENTRE

Understanding Variable Scope

1.Global Scope – A variable that can be used anywhere in the program.

2.Local Scope – A variable that can be used only inside a function.

3.Functional Scope – A variable works inside the whole function, even inside {} blocks.

4.Block Scope – A variable works only inside {} where it is declared.

BAGHEL COMPUTER CENTRE

- To create a variable, we use keywords like **let, const, or var.**

Keyword	let	const	var
Global Scope	N	N	Y
Functional Scope	Y	Y	Y
Block Scope	Y	Y	N
Can be reassigned	Y	N	Y
Can be re-declared	N	N	Y

Syntax for variable declare:-

let/var/const	x	=	10	;
declare a variable	variable name	assignment operator	data or variable assign to "x"	seperator

BAGHEL COMPUTER CENTRE

Types of Variables in JavaScript

var – Old way to declare variables (can be changed and used anywhere).

let – Modern way, can be changed but only inside its block.

const – Stores fixed values that cannot be changed.

BAGHEL COMPUTER CENTRE

Let

Redeclare - no

```
1   let x=10;
2   let x=11; //redeclare not allowed
3   console.log(x);

Console ×                                    ...
The symbol "x" has already been declared     script.js? 2:4
```

Reassign- yes

```
1   let x=10;
2    x=11; //reassign allowed
3   console.log(x);

Console ×

11
```

BAGHEL COMPUTER CENTRE

Blockscope-yes

```
let x=10;
console.log(x);
{
let x=11; //blockscope
console.log(x);
}

console.log(x);
```

Console ×

```
10
11
10
```

BAGHEL COMPUTER CENTRE

Functional scope-yes

```
function sayHello() {
    let message = "Hello!";
    console.log(message); // Works here
}

sayHello();

console.log(message); // ERROR: 'message' is not accessible outside
```

```
Console ×                                   ···   Web View ×
Hello!
▸ ReferenceError: message is not defined
    at <anonymous>:14:13
    at mn (<anonymous>:16:5455)
```

BAGHEL COMPUTER CENTRE

Global scope-no

```
<!DOCTYPE html>
<html>
<head>
    <title>let is not global scope</title>
</head>
<body>
   <script>
        let greeting = "Hi!";
        document.write(greeting);        // "Hi!"
        document.write(window.greeting); // undefined
   </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Const

Redecalre-no

```
1   const a=10;
2   const a=11;
3   console.log(a);
```

```
Console ×                                                    ...
The symbol "a" has already been declared          script.js? 2:6
```

BAGHEL COMPUTER CENTRE

Reassign-no

```
1   const a=10;
2   a=11;
3   console.log(a);
```

```
Console ×                                                    ...
Cannot assign to "a" because it is a constant   script.js? 2:1
```

BAGHEL COMPUTER CENTRE

Global scope-no

```
<!DOCTYPE html>
<html>
<head>
    <title>const is not global scope reassign</title>
</head>
<body>
    <script>
        const greeting = "Hi!";
        document.write(greeting);        // "Hi!"
        document.write(window.greeting); // undefined
    </script>
</body>
</html>
```

Blockscope-yes

```
1  const a=10;
2   console.log(a);
3   {
4    const a=11;
5  console.log(a);}
6
7   console.log(a);
```

Console ×

```
10
11
10
```

BAGHEL COMPUTER CENTRE

Functional scope- yes

```
1  function sayhello(){
2    let message="hello";
3    console.log(message);//work here
4  }
5  sayhello();
6  console.log(message);//error
```

Console ×

```
hello
null
```

BAGHEL COMPUTER CENTRE

Var

Functional scope-yes

```
1  function sayhello(){
2    var message="hello";
3    console.log(message);//work here
4  }
5  sayhello();
6  console.log(message);//error
```

```
Console ×
hello
null
```

BAGHEL COMPUTER CENTRE

Block scope-no

```
1  var a=10;
2   console.log(a);
3   {
4   var a=11;
5  console.log(a);}
6
7   console.log(a);
```

Console ×

```
10
11
11
```

BAGHEL COMPUTER CENTRE

Redeclare - yes

```
1   var a=10;
2   var a=11;
3   console.log(a);
```

```
Console ×
11
```

Reassign - yes

```
1   var a=10;
2    a=11;
3   console.log(a);
```

```
Console ×
11
```

BAGHEL COMPUTER CENTRE

Global scope-yes

```
<!DOCTYPE html>
<html>
<head>
    <title>var is global scope </title>
</head>
<body>
   <script>
        var greeting = "Hi!";
        document.write(greeting);          // "Hi!"
        document.write(window.greeting); // "Hi!"
   </script>
</body>
</html>
```

Output

Hi!Hi!

BAGHEL COMPUTER CENTRE

2. Data

Data refers to the **actual information or values stored in variables**. In JavaScript, data can come in different types, including:

- **Numbers:** Like 5, 3.14
- **Strings:** Text values, like "Hello, World!"
- **Booleans:** True or false values, like true or false
- **Objects:** Complex data with multiple properties which contain key and values, like {name: "Alice", age: 25}
- **Arrays:** Lists of values, like [1, 2, 3]
- **Undefined :** Variable declared but not assigned a value ,like(let x;)
- **Null :** Represents "nothing" or empty, let y = null;

BAGHEL COMPUTER CENTRE

Rules to declare a variable

- Variable names must start with a letter, _ (underscore), or $ (dollar sign).

```
name = "John";  // ✅ Valid
_age = 25;      // ✅ Valid
$price = 100;   // ✅ Valid
```

- Variable names cannot start with a number.

```
1name = "Hello"; // ❌ Invalid
```

- Variable names can only contain letters, numbers, _, and $.

```
user_name = "Alex";  // ✅ Valid
user$name = "Sam";   // ✅ Valid
user-name = "Lily";  // ❌ Invalid (hyphen not allowed)
```

BASEL COMPUTER CENTRE

- **JavaScript is case-sensitive** (uppercase and lowercase are different).

```
age = 30;
Age = 40;
console.log(age); // 30
console.log(Age); // 40
```

- **Variable names cannot be** JavaScript reserved words (like var, if, function).

```
var = 50;  // ❌ Invalid
function = "Hello"; // ❌ Invalid
```

BAGHEL COMPUTER CENTRE

JavaScript

BAGHEL COMPUTER CENTRE

Index

Comments
Use /* ... */ to add comments that span multiple lines
JavaScript Operator Types
1. Arithmetic Operators
2. JavaScript Assignment Operators
3. JavaScript Comparison Operators
4. JavaScript Logical Operators
5.JavaScript Bitwise Operators
    - JavaScript Bitwise AND
    - JavaScript Bitwise OR
    - JavaScript Bitwise XOR
    - JavaScript Bitwise NOT
    - JavaScript Left shift (<<)
    - JavaScript Right shift (>>)
6.Special operator
    - typeof
    - instanceof

BAGHEL COMPUTER CENTRE

Comments

Single-line Comments

Use // for comments that fit on one line.

Multi-line Comments

Use /* ... */ to add comments that span multiple lines

BAGHEL COMPUTER CENTRE

JavaScript Operator Types

list of different JavaScript operators:-

1. Arithmetic Operators
2. Assignment Operators
3. Comparison Operators
4. Logical Operators
5. Bitwise Operators
6. Special Operators

BAGHEL COMPUTER CENTRE

1. Arithmetic Operators

Operator	Name	Example
+	Addition	3 + 4 // 7
-	Subtraction	5 - 3 // 2
*	Multiplication	2 * 3 // 6
/	Division	4 / 2 // 2
%	Remainder	5 % 2 // 1
++	Increment (increments by 1)	++5 // 6
--	Decrement (decrements by 1)	--4 // 3
**	Exponentiation (Power)	4 ** 2 // 16

BAGHEL COMPUTER CENTRE

INPUT

With document.getElementById

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Operators Example</title>
</head>
<body>
    <h1>JavaScript Operators</h1>
    <p id="addition"></p>
    <p id="subtraction"></p>
    <p id="multiplication"></p>
    <p id="division"></p>
    <p id="remainder"></p>
    <p id="increment"></p>
    <p id="decrement"></p>
    <p id="exponentiation"></p>
    <script>
        let x = 5;
        let Y = 5;

        // Displaying each operation result directly in HTML
        document.getElementById("addition").innerHTML = "Addition: x + 3 = " + (x + 3);
        document.getElementById("subtraction").innerHTML = "Subtraction: x - 3 = " + (x - 3);
        document.getElementById("multiplication").innerHTML = "Multiplication: x * 3 = " + (x * 3);
        document.getElementById("division").innerHTML = "Division: x / 3 = " + (x / 3);
        document.getElementById("remainder").innerHTML = "Remainder: x % 3 = " + (x % 3);

        // Increment and display updated x value
        document.getElementById("increment").innerHTML = "Increment: ++x = " + (++x);

        // Decrement and display updated x value
        document.getElementById("decrement").innerHTML = "Decrement: --Y = " + (--Y);

        // Exponentiation
        document.getElementById("exponentiation").innerHTML = "Exponentiation: x ** 3 = " + (x ** 3);
    </script>

</body>
</html>
```

BAGHEL COMPUTER CENTRE

OUTPUT

JavaScript Operators

Addition: x + 3 = 8

Subtraction: x - 3 = 2

Multiplication: x * 3 = 15

Division: x / 3 = 1.6666666666666667

Remainder: x % 3 = 2

Increment: ++x = 6

Decrement: --Y = 4

Exponentiation: x ** 3 = 216

BAGHEL COMPUTER CENTRE

Input

With document.write()

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Operators Example</title>
</head>
<body>
    <h1>JavaScript Operators</h1>
    <script>
        let x = 5;
        let Y = 5;

        // Displaying each operation result using document.write
        document.write("Addition: x + 3 = " + (x + 3) + "<br>");
        document.write("Subtraction: x - 3 = " + (x - 3) + "<br>");
        document.write("Multiplication: x * 3 = " + (x * 3) + "<br>");
        document.write("Division: x / 3 = " + (x / 3) + "<br>");
        document.write("Remainder: x % 3 = " + (x % 3) + "<br>");

        // Increment and display updated x value
        document.write("Increment: ++x = " + (++x) + "<br>");

        // Decrement and display updated Y value
        document.write("Decrement: --Y = " + (--Y) + "<br>");

        // Exponentiation
        document.write("Exponentiation: x ** 3 = " + (x ** 3) + "<br>");
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

2. JavaScript Assignment Operators

Operator	Name	Example
=	Assignment Operator	a = 7;
+=	Addition Assignment	a += 5; // a = a + 5
-=	Subtraction Assignment	a -= 2; // a = a - 2
*=	Multiplication Assignment	a *= 3; // a = a * 3
/=	Division Assignment	a /= 2; // a = a / 2
%=	Remainder Assignment	a %= 2; // a = a % 2
**=	Exponentiation Assignment	a **= 2; // a = a**2

BAGHEL COMPUTER CENTRE

Input

With document.getElementById

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Assignment Operators</title>
</head>
<body>
    <h1>Assignment Operators Example</h1>
    <p id="assign"></p>
    <p id="addAssign"></p>
    <p id="subAssign"></p>
    <p id="mulAssign"></p>
    <p id="divAssign"></p>
    <p id="remAssign"></p>
    <p id="expAssign"></p>
    <script>
        let a = 7;

        // Assignment
        document.getElementById("assign").innerHTML = "Assignment: a = " + a;

        // Addition Assignment
        a += 5;
        document.getElementById("addAssign").innerHTML = "Addition Assignment: a += 5;
        Result: " + a;

        // Subtraction Assignment
        a -= 2;
        document.getElementById("subAssign").innerHTML = "Subtraction Assignment: a -=
        2; Result: " + a;

        // Multiplication Assignment
        a *= 3;
        document.getElementById("mulAssign").innerHTML = "Multiplication Assignment: a
        *= 3; Result: " + a;

        // Division Assignment
        a /= 2;
        document.getElementById("divAssign").innerHTML = "Division Assignment: a /= 2;
        Result: " + a;

        // Remainder Assignment
        a %= 2;
        document.getElementById("remAssign").innerHTML = "Remainder Assignment: a %= 2;
        Result: " + a;

        // Exponentiation Assignment
        a = 3; // Reset a
        a **= 2;
        document.getElementById("expAssign").innerHTML = "Exponentiation Assignment: a
        **= 2; Result: " + a;
    </script>
</body>
</html>
```

RAGHEL COMPUTER CENTRE

Output

Assignment Operators Example

Assignment: a = 7

Addition Assignment: a += 5; Result: 12

Subtraction Assignment: a -= 2; Result: 10

Multiplication Assignment: a *= 3; Result: 30

Division Assignment: a /= 2; Result: 15

Remainder Assignment: a %= 2; Result: 1

Exponentiation Assignment: a **= 2; Result: 9

BAGHEL COMPUTER CENTRE

Input

With document.write()

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Assignment Operators</title>
</head>
<body>
    <h1>Assignment Operators Example</h1>
    <script>
        let a = 7;

        // Assignment
        document.write("Assignment: a = " + a + "<br>");

        // Addition Assignment
        a += 5;
        document.write("Addition Assignment: a += 5; Result: " + a + "<br>");

        // Subtraction Assignment
        a -= 2;
        document.write("Subtraction Assignment: a -= 2; Result: " + a + "<br>");

        // Multiplication Assignment
        a *= 3;
        document.write("Multiplication Assignment: a *= 3; Result: " + a + "<br>");

        // Division Assignment
        a /= 2;
        document.write("Division Assignment: a /= 2; Result: " + a + "<br>");

        // Remainder Assignment
        a %= 2;
        document.write("Remainder Assignment: a %= 2; Result: " + a + "<br>");

        // Exponentiation Assignment
        a = 3; // Reset a
        a **= 2;
        document.write("Exponentiation Assignment: a **= 2; Result: " + a + "<br>");
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Output

Assignment Operators Example

Assignment: a = 7
Addition Assignment: a += 5; Result: 12
Subtraction Assignment: a -= 2; Result: 10
Multiplication Assignment: a *= 3; Result: 30
Division Assignment: a /= 2; Result: 15
Remainder Assignment: a %= 2; Result: 1
Exponentiation Assignment: a **= 2; Result: 9

BAGHEL COMPUTER CENTRE

3. JavaScript Comparison Operators

Operator	Meaning	Example
==	Equal to	3 == 5 gives us false
!=	Not equal to	3 != 4 gives us true
>	Greater than	4 > 4 gives us false
<	Less than	3 < 3 gives us false
>=	Greater than or equal to	4 >= 4 gives us true
<=	Less than or equal to	3 <= 3 gives us true
===	Strictly equal to	3 === "3" gives us false

BAGHEL COMPUTER CENTRE

!==	Strictly not equal to	3 !== "3" gives us true

Difference between == and === in JavaScript:

1. == (Equality Operator)

Checks value only.

2. === (Strict Equality Operator)

Checks value and type.

BAGHEL COMPUTER CENTRE

Input

With document.getElementById

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Comparison Operators</title>
</head>
<body>
    <h1>Comparison Operators Example</h1>
    <p id="equalTo"></p>
    <p id="notEqualTo"></p>
    <p id="greaterThan"></p>
    <p id="lessThan"></p>
    <p id="greaterThanOrEqualTo"></p>
    <p id="lessThanOrEqualTo"></p>
    <p id="strictEqualTo"></p>
    <p id="strictNotEqualTo"></p>

    <script>
        let a = 3;
        let b = 5;
        let c = 4;
        let d = "3";

        // Equal to
        document.getElementById("equalTo").innerHTML = "3 == 5 gives us " + (a == b);

        // Not equal to
        document.getElementById("notEqualTo").innerHTML = "3 != 4 gives us " + (a != c);

        // Greater than
        document.getElementById("greaterThan").innerHTML = "4 > 4 gives us " + (c > b);

        // Less than
        document.getElementById("lessThan").innerHTML = "3 < 3 gives us " + (a < a);

        // Greater than or equal to
        document.getElementById("greaterThanOrEqualTo").innerHTML = "4 >= 4 gives us " +
        (c >= c);

        // Less than or equal to
        document.getElementById("lessThanOrEqualTo").innerHTML = "3 <= 3 gives us " + (a
        <= a);

        // Strictly equal to
        document.getElementById("strictEqualTo").innerHTML = "3 === '3' gives us " + (a
        === d);

        // Strictly not equal to
        document.getElementById("strictNotEqualTo").innerHTML = "3 !== '3' gives us " + (
        a !== d);
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Output

Comparison Operators Example

3 == 5 gives us false

3 != 4 gives us true

4 > 4 gives us false

3 < 3 gives us false

4 >= 4 gives us true

3 <= 3 gives us true

3 === '3' gives us false

3 !== '3' gives us true

BAGHEL COMPUTER CENTRE

Input

With document.write()

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Comparison Operators</title>
</head>
<body>
    <h1>Comparison Operators Example</h1>
    <script>
        let a = 3;
        let b = 5;
        let c = 4;
        let d = "3";

        // Equal to
        document.write("3 == 5 gives us " + (a == b) + "<br>");

        // Not equal to
        document.write("3 != 4 gives us " + (a != c) + "<br>");

        // Greater than
        document.write("4 > 4 gives us " + (c > b) + "<br>");

        // Less than
        document.write("3 < 3 gives us " + (a < a) + "<br>");

        // Greater than or equal to
        document.write("4 >= 4 gives us " + (c >= c) + "<br>");

        // Less than or equal to
        document.write("3 <= 3 gives us " + (a <= a) + "<br>");

        // Strictly equal to
        document.write("3 === '3' gives us " + (a === d) + "<br>");

        // Strictly not equal to
        document.write("3 !== '3' gives us " + (a !== d) + "<br>");
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Output

Comparison Operators Example

3 == 5 gives us false
3 != 4 gives us true
4 > 4 gives us false
3 < 3 gives us false
4 >= 4 gives us true
3 <= 3 gives us true
3 === '3' gives us false
3 !== '3' gives us true

BAGHEL COMPUTER CENTRE

4. JavaScript Logical Operators

Operator	Syntax	Description
&& (Logical AND)	&&	true only if both expression1 and expression2 are true
|| (Logical OR)	||	true if either expression1 or expression2 is true
! (Logical NOT)	!	false if expression is true and vice versa

BAGHEL COMPUTER CENTRE

Input

With document.getElementById

```
<!DOCTYPE html>
<html>
<head>
    <title>Logical Operators with Numbers</title>
</head>
<body>
    <h1>Logical Operators Example</h1>

    <p id="and1"></p>
    <p id="and2"></p>
    <p id="or1"></p>
    <p id="or2"></p>
    <p id="not1"></p>
    <p id="not2"></p>

    <script>
        let x = 3;

        // Logical AND
        document.getElementById("and1").innerHTML = "(x < 5) && (x > 0): " + ((x < 5) && (x > 0));  // true
        document.getElementById("and2").innerHTML = "(x < 5) && (x > 6): " + ((x < 5) && (x > 6));  // false

        // Logical OR
        document.getElementById("or1").innerHTML = "(x > 2) || (x > 5): " + ((x > 2) || (x > 5));  // true
        document.getElementById("or2").innerHTML = "(x > 3) || (x < 0): " + ((x > 3) || (x < 0));  // false

        // Logical NOT
        document.getElementById("not1").innerHTML = "!(x == 3): " + (!(x == 3));  // false
        document.getElementById("not2").innerHTML = "!(x < 2): " + (!(x < 2));  // true
    </script>
</body>
</html>
```

Output

Logical Operators Example

(x < 5) && (x > 0): true

(x < 5) && (x > 6): false

(x > 2) || (x > 5): true

(x > 3) || (x < 0): false

!(x == 3): false

!(x < 2): true

BAGHEL COMPUTER CENTRE

Input

With document.write()

```
<!DOCTYPE html>
<html>
<head>
    <title>Logical Operators with Numbers</title>
</head>
<body>
    <h1>Logical Operators Example</h1>

    <script>
        let x = 3;

        // Logical AND
        document.write("(x < 5) && (x > 0): " + ((x < 5) && (x > 0)) + "<br>");  // true
        document.write("(x < 5) && (x > 6): " + ((x < 5) && (x > 6)) + "<br>");  // false

        // Logical OR
        document.write("(x > 2) || (x > 5): " + ((x > 2) || (x > 5)) + "<br>");  // true
        document.write("(x > 3) || (x < 0): " + ((x > 3) || (x < 0)) + "<br>");  // false

        // Logical NOT
        document.write("!(x == 3): " + (!(x == 3)) + "<br>");  // false
        document.write("!(x < 2): " + (!(x < 2)) + "<br>");  // true
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Output

Logical Operators Example

(x < 5) && (x > 0): true
(x < 5) && (x > 6): false
(x > 2) || (x > 5): true
(x > 3) || (x < 0): false
!(x == 3): false
!(x < 2): true

BAGHEL COMPUTER CENTRE

5.JavaScript Bitwise Operators

Operators	Name	Example
&	Bitwise AND	x & y
|	Bitwise OR	x | y
^	Bitwise XOR	x ^ y
~	Bitwise NOT	~x
<<	Left shift	x << y
>>	right shift	x >> y

BAGHEL COMPUTER CENTRE

JavaScript Bitwise AND

Bitwise AND & returns 1 if the corresponding bits of both operands are 1 else it returns 0.

Operand 1	Operand 2	AND Operation
0	0	0 & 0 is 0
0	1	0 & 1 is 0
1	0	1 & 0 is 0
1	1	1 & 1 is 1

BAGHEL COMPUTER CENTRE

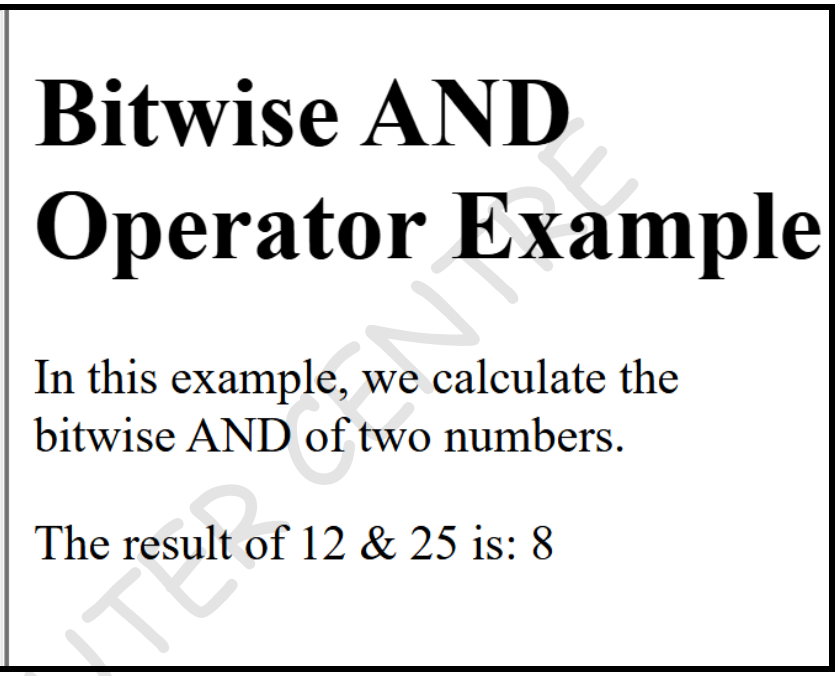
Let's take a look at the bitwise AND operation of two integers 12 and 25.

```
In binary,

12 = 01100
25 = 11001

// Bitwise AND Operation of 12 and 25

    00001100
&   00011001
    ---------
    00001000  = 8 (In decimal)
```

BAGHEL COMPUTER CENTRE

Example 1: Bitwise AND Operator

```
<!DOCTYPE html>
<html>
<head>
    <title>Bitwise AND Operator</title>
</head>
<body>
    <h1>Bitwise AND Operator Example</h1>
    <p>In this example, we calculate the
    bitwise AND of two numbers.</p>

    <script>
        let a = 12;
        let b = 25;

        let result = a & b; // Perform
        bitwise AND operation
        document.write("The result of " + a
        + " & " + b + " is: " + result);
    </script>
</body>
</html>
```



BAGHEL COMPUTER CENTRE

JavaScript Bitwise OR

Bitwise OR | returns 1 if either of the corresponding bits of one operand is 1 else returns 0.

Operand 1	Operand 2	OR Operation
0	0	0 | 0 is 0
0	1	0 | 1 is 1
1	0	1 | 0 is 1
1	1	1 | 1 is 1

BAGHEL COMPUTER CENTRE

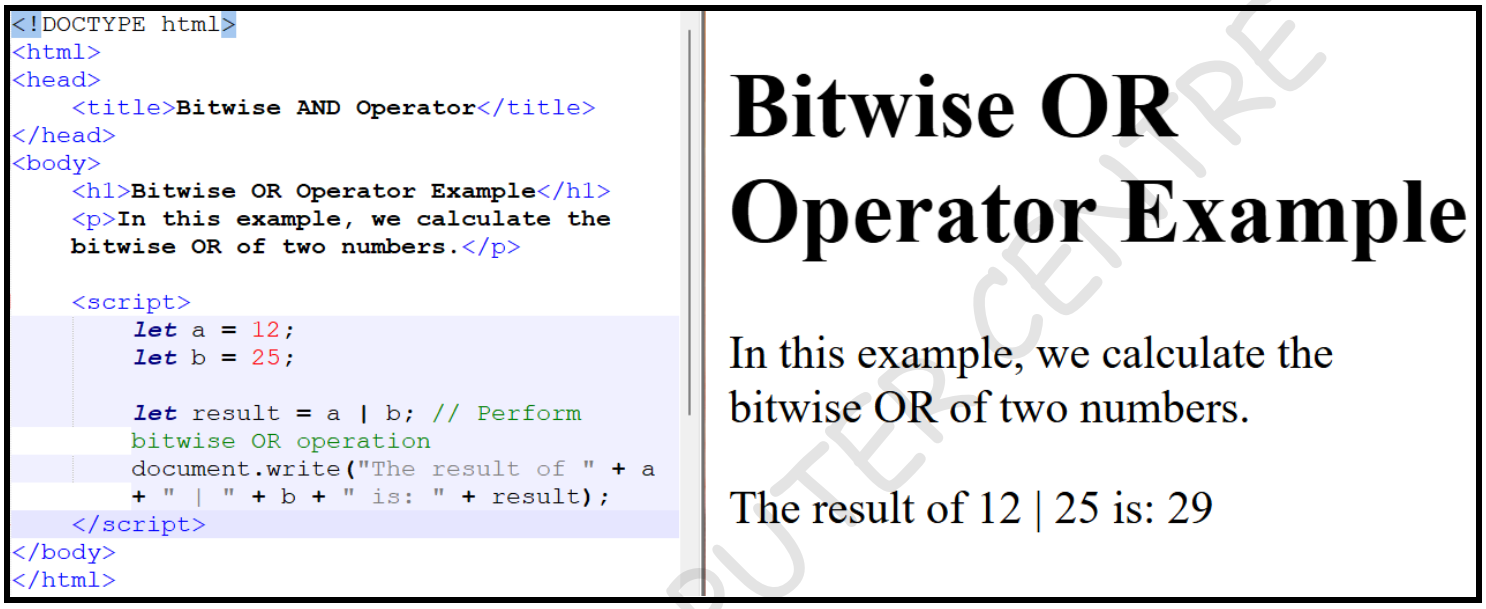
Let's take a look at the bitwise OR operation of two integers 12 and 25.

```
In binary,

12 = 01100
25 = 11001

// Bitwise OR Operation of 12 and 25

    00001100
|   00011001
    --------
    00011101  = 29 (In decimal)
```

BAGHEL COMPUTER CENTRE

Example 2: Bitwise OR Operator

```html
<!DOCTYPE html>
<html>
<head>
    <title>Bitwise AND Operator</title>
</head>
<body>
    <h1>Bitwise OR Operator Example</h1>
    <p>In this example, we calculate the
    bitwise OR of two numbers.</p>

    <script>
        let a = 12;
        let b = 25;

        let result = a | b; // Perform
        bitwise OR operation
        document.write("The result of " + a
        + " | " + b + " is: " + result);
    </script>
</body>
</html>
```



BAGHEL COMPUTER CENTRE

JavaScript Bitwise XOR

Bitwise XOR ^ returns 1 if the corresponding bits are different and returns 0 if the corresponding bits are the same.

Operand 1	Operand 2	XOR Operation
0	0	0 ^ 0 is 0
0	1	0 ^ 1 is 1
1	0	1 ^ 0 is 1
1	1	1 ^ 1 is 0

BAGHEL COMPUTER CENTRE

```
In binary,

12 = 01100
25 = 11001

// Bitwise XOR Operation of 12 and 25

    00001100
^   00011001
    --------
    00010101  = 21 (In decimal)
```

Example 3: Bitwise XOR Operator

```
<!DOCTYPE html>
<html>
<head>
    <title>Bitwise AND Operator</title>
</head>
<body>
    <h1>Bitwise XOR Operator Example</h1>
    <p>In this example, we calculate the
    bitwise XOR of two numbers.</p>

    <script>
        let a = 12;
        let b = 25;

        let result = a ^ b; // Perform
        bitwise XOR operation
        document.write("The result of " + a
        + " ^ " + b + " is: " + result);
    </script>
</body>
</html>
```

Bitwise XOR Operator Example

In this example, we calculate the bitwise XOR of two numbers.

The result of 12 ^ 25 is: 21

BAGHEL COMPUTER CENTRE

JavaScript Bitwise NOT

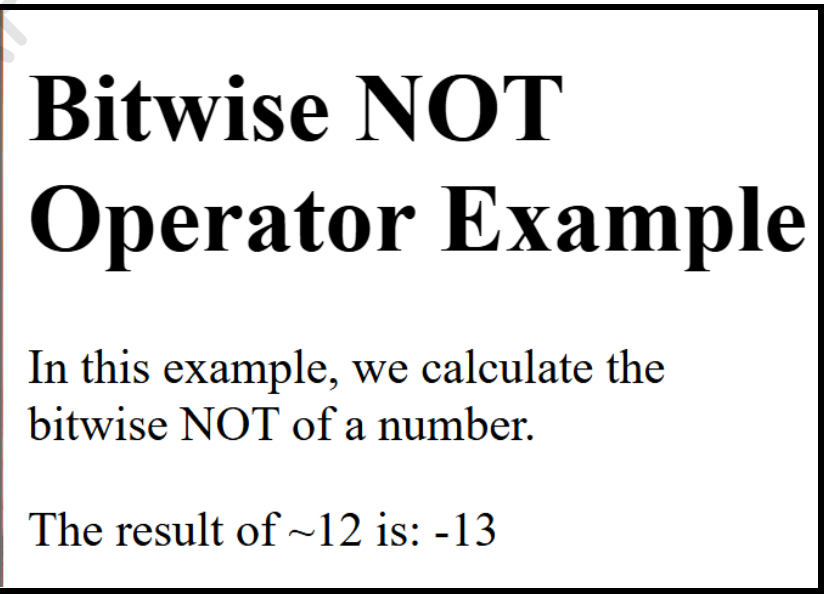
Bitwise NOT ~ inverts the bit(0 becomes 1, 1 becomes 0).

Formula:

~a = -(a + 1)

Example 4: Bitwise NOT Operator

```
<!DOCTYPE html>
<html>
<head>
    <title>Bitwise AND Operator</title>
</head>
<body>
    <h1>Bitwise NOT Operator Example</h1>
    <p>In this example, we calculate the
    bitwise NOT of a number.</p>

    <script>
        let a = 12;  // Example number

        let result = ~a;  // Perform
        bitwise NOT operation
        document.write("The result of ~" + a
         + " is: " + result);
    </script>
</body>
</html>
```



BAGHEE COMPUTER CENTRE

JavaScript Left shift (<<)

To shift 3 left by 2 positions:

1. Initial value: 3 in binary = 00000011.
2. Left shift by 2: Move bits left by 2 and add 0s on the right → 00001100.
3. Result: 00001100 = 12 in decimal.

Shortcut:

1- 3 << 2 = 3 * (2^2) = 3 * 4 = 12.

2- 5 << 3 = 5 * (2^3) = 5 * 8 = 40.

BAGHEL COMPUTER CENTRE

JavaScript Right shift (>>)

To shift 12 right by 2 positions:

1. Initial value: 12 in binary = 00001100.
2. Right shift by 2: Move bits right by 2 and discard the rightmost bits → 00000011.
3. Result: 00000011 = 3 in decimal.

Shortcut:

1. 12 >> 2 = 12 / (2^2) = 12 / 4 = 3.
2. 40 >> 3 = 40 / (2^3) = 40 / 8 = 5.

BAGHEL COMPUTER CENTRE

```
<!DOCTYPE html>
<html>
<head>
    <title>Bitwise AND Operator</title>
</head>
<body>
   <h1>Bitwise Left and Right Shift Operators</h1>
    <p>In this example, we demonstrate the left and right
    shift operators.</p>

    <script>
        document.write("12 << 2 = " + (12 << 2) + "<br>");
        document.write("12 >> 2 = " + (12 >> 2));
    </script>
</body>
</html>
```

Bitwise Left and Right Shift Operators

In this example, we demonstrate the left and right shift operators.

12 << 2 = 48
12 >> 2 = 3

BAGHEL COMPUTER CENTRE

6.Special operator

typeof

The typeof operator in JavaScript is used to determine the data type of a value or variable. It returns a string indicating the type.

It helps identify the type of a value, which is especially useful in debugging and writing dynamic code.

BAGHEL COMPUTER CENTRE

Input

```
<!DOCTYPE html>
<html>
<head>
  <title>JavaScript typeof</title>
</head>
<body>
  <script>
    document.write(typeof "Hello" + "<br>");
    document.write(typeof 42 + "<br>");
    document.write(typeof {name: 'John'} + "<br>");
    document.write(typeof function(){} + "<br>");
    document.write(typeof undefined + "<br>");
  </script>
</body>
</html>
```

Output

string
number
object
function
undefined

BAGHEL COMPUTER CENTRE

What is object

An object is a data structure that stores key-value pairs. Each key is a unique identifier, and the value can be any type of data.

Note:- array is also a type of object.

Syntax Example:

```
let person = {
  name: "John",  // key: "name", value: "John"
  age: 30        // key: "age", value: 30
};
```

BAGHEL COMPUTER CENTRE

Explanation:

- person is an object with two properties: name and age.
- The key "name" has the value "John", and the key "age" has the value 30.

Objects are written inside {} and each key-value pair is separated by a comma.

BAGHEL COMPUTER CENTRE

instanceof

Definition: instanceof checks if an object is made from a specific type or class.

It is true if the object is of that type, otherwise false.

Input

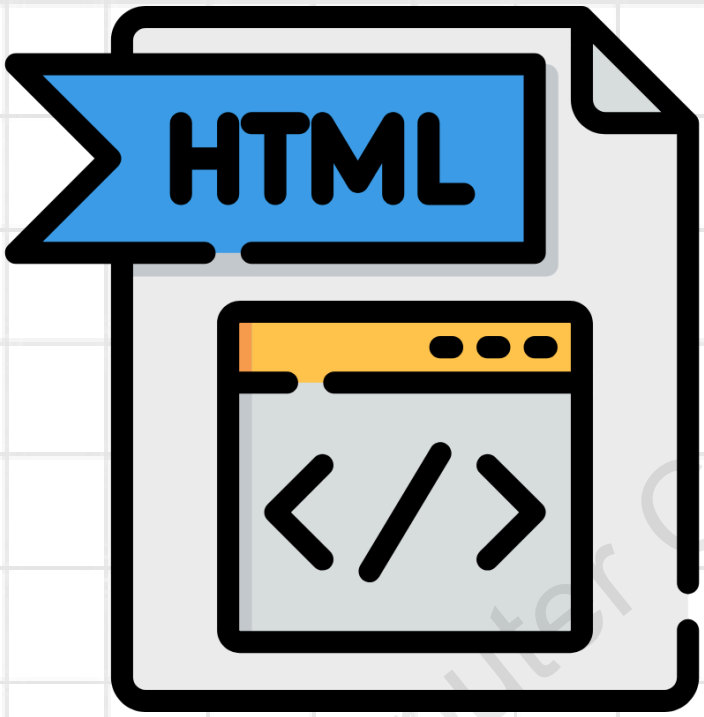
```
<!DOCTYPE html>
<html>
<head>
  <title>JavaScript instanceof</title>
</head>
<body>
  <script>
    let obj = { name: "John", age: 30 };
let arr = [1, 2, 3];   // Array is also an object in JS
let str = "Hello";
let num = 42;

document.write(obj instanceof Object);   // true, because obj is an object
document.write("<br>");
document.write(arr instanceof Object);   // true, because arrays are objects
document.write("<br>");
document.write(str instanceof Object);   // false, because string is not an object
document.write("<br>");
document.write(num instanceof Object);   // false, because number is not an object

  </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Output

```
true
true
false
false
```

BAGHEL COMPUTER CENTRE



"Conditional Statement "

Bagul Computer Centre

Conditional Statement

- A conditional statement allows a program to make decisions by running different code depending on whether a condition is true or false.
- **Types of Conditional Statements:**
  1. `if Statement`
  2. `if...else Statement`
  3. `if...else if...else Statement`
  4. `switch Statement`

Baghel Computer Centre

1- parseInt()

Meaning:

`parseInt()` converts a string to an integer by parsing it and stopping at the first non-numeric character.

2- parseFloat()

Meaning:

`parseFloat()` converts a string to a floating-point number, allowing decimals.

3- prompt()

Meaning:

`prompt()` displays a dialog box that asks the user for input and returns the input as a string.

Baghel Computer Centre

1-

if Statement

The if statement runs a block of code only if a specified condition is true.

Syntax:

```
if (condition) {
    // Code to run if the condition is true
}
```

Baghel Computer Centre

2-

if...else Statement

The if...else statement runs one block of code if the condition is true and another block if it is false.

Syntax:

```
if (condition) {
    // Code to run if the condition is true
} else {
    // Code to run if the condition is false
}
```

Baghel Computer Centre

3-

if...else if...else Statement

The if...else if...else statement tests multiple conditions in sequence and runs the block of code for the first condition that evaluates to true.

Syntax:

```
if (condition1) {
    // Code to run if condition1 is true
} else if (condition2) {
    // Code to run if condition2 is true
} else {
    // Code to run if none of the conditions are true
}
```

Baghel Computer Centre

4-

switch Statement

The `switch` statement evaluates an expression and runs the matching case's block of code. If no cases match, the `default` block is executed..

Syntax:

```
switch (expression) {
    case value1:
        // Code to run if expression equals value1
        break;
    case value2:
        // Code to run if expression equals value2
        break;
    default:
        // Code to run if no cases match
}
```

Baghel Computer Centre

JavaScript

BAGHEL COMPUTER CENTRE

Index

Function in JavaScript

A function is a block of reusable code that performs a specific task. You can use functions to organize and reuse code without having to write the same code repeatedly.

Syntax of a Function:

```
function functionName() {
  // Code to be executed
}
```

BAGHEL COMPUTER CENTRE

explanation:

- **function:** This keyword tells JavaScript that you're defining a function.
- **functionName:** The name of the function. You can call this function by its name.
- **{ }:** Curly braces contain the code that the function will execute when it's called.

BAGHEL COMPUTER CENTRE

🧩 Question 1: How to create a function that prints "Hello, World!"?

```html
<!DOCTYPE html>
<html>
<head>
    <title>Function Example 1</title>
</head>
<body>
    <h2>Question 1: Print "Hello, World!"</h2>
    <script>
        // Creating the function
        function sayHello() {
            document.write("Hello, World!<br>");
        }

        // Calling the function
        sayHello();
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

🧩 Question 2: How to pass a name to a function and print it?

```
<!DOCTYPE html>
<html>
<head>
    <title>Function Example 2</title>
</head>
<body>
     <h2>Question 2: Pass a name to a function and print it</h2>
    <script>
        // Creating the function
        function greet(name) {
            document.write("Hello, " + name + "!<br>");
        }

        // Calling the function
        greet("Alice");
        greet("Bob");
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

🧩 Question 3: How to create a function that adds two numbers and prints the result?

```html
<!DOCTYPE html>
<html>
<head>
    <title>Function Example 3</title>
</head>
<body>
    <h2>Question 3: Add two numbers and print the result</h2>
    <script>
        // Creating the function
        function addNumbers(a, b) {
            let sum = a + b;
            document.write("The sum is: " + sum + "<br>");
        }

        // Calling the function
        addNumbers(5, 10);
        addNumbers(20, 30);
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

Now user define

```html
<!DOCTYPE html>
<html>
<head>
    <title>Question 3</title>
</head>
<body>
    <input type="number" id="num1" placeholder="Enter first number">
    <input type="number" id="num2" placeholder="Enter second number">
    <button onclick="addNumbers()">Add</button>

    <script>
        // Step 1: Create a function
        function addNumbers() {
            // Step 2: Get the values of the two numbers
            let num1 = parseFloat(document.getElementById("num1").value);
            let num2 = parseFloat(document.getElementById("num2").value);

            // Step 3: Calculate the sum and print the result
            let sum = num1 + num2;
            document.write("The result is: " + sum);
        }
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

🧩 Question 4: How to create a function that checks if a number is even or odd?

```html
<!DOCTYPE html>
<html>
<head>
    <title>Question 4</title>
</head>
<body>

    <input type="number" id="number" placeholder="Enter a number">
    <button onclick="checkEvenOdd()">Check</button>

    <script>
        // Step 1: Create a function
        function checkEvenOdd() {
            // Step 2: Get the number from the input
            let number = parseInt(document.getElementById("number").value);

            // Step 3: Check if the number is even or odd
            if (number % 2 === 0) {
                document.write(number + " is Even");
            } else {
                document.write(number + " is Odd");
            }
        }
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

🧩 Question 5: Which is the largest of the two numbers?

```html
<!DOCTYPE html>
<html>
<head>
    <title>Largest Number</title>
</head>
<body>
    <h2>Find the Largest of Two Numbers</h2>
    <input type="number" id="num1" placeholder="Enter first number">
    <input type="number" id="num2" placeholder="Enter second number">
    <button onclick="findLargest()">Check</button>

    <script>
        function findLargest() {
            let num1 = parseFloat(document.getElementById("num1").value);
            let num2 = parseFloat(document.getElementById("num2").value);

            if (num1 > num2) {
                document.write(num1 + " is largest");
            } else {
                document.write(num2 + " is largest");
            }
        }
    </script>
</body>
</html>
```

BAGHEL COMPUTER CENTRE

JavaScript

Baghel Computer Centre

What is a JavaScript Event?

A JavaScript event is an action or something that happens on a webpage.

1. onclick: Happens when you click on an element (like a button).
2. onmouseover: Activates when the mouse pointer moves over an element (like hovering over a button).
3. onmouseout: Happens when the mouse pointer leaves an element (like moving the cursor off a button).
4. onkeydown: Happens when a key is pressed on the keyboard.
5. onload:Happens when a webpage or element finishes loading (like when the page fully loads).
6. oninput: Happens when the value in an input field changes in real-time (e.g., as you type).

Bagri Computer Centre

7. onchange: Happens when the value in an input field changes (e.g., typing in a text box).
8. onfocus: Happens when an input element (like a text box) is selected or clicked on.
9. onblur: Happens when you move away from an input element (like clicking out of a text box).
10. onsubmit: Happens when a form is submitted, often used to validate the form before submission.

Baghel Computer Centre

Onclick

JavaScript Popup Boxes

1 Alert Box

It shows a message to the user with an OK button.

Syntax: alert();

Input

```
<!doctype html>
<html>
<head>
  <title>Alert Box Example</title>
</head>
<body>

<h2>Alert Box Example</h2>
<button onclick="showAlert()">Click me to show Alert</button>

<script>
function showAlert() {
  alert('This is an alert message!');
}
</script>

</body>
</html>
```

Baghel Computer Centre

Output

Baghel Computer Centre

2 Confirm Box

It shows a message with OK and Cancel buttons. It returns true if the user clicks OK, and false if they click Cancel.

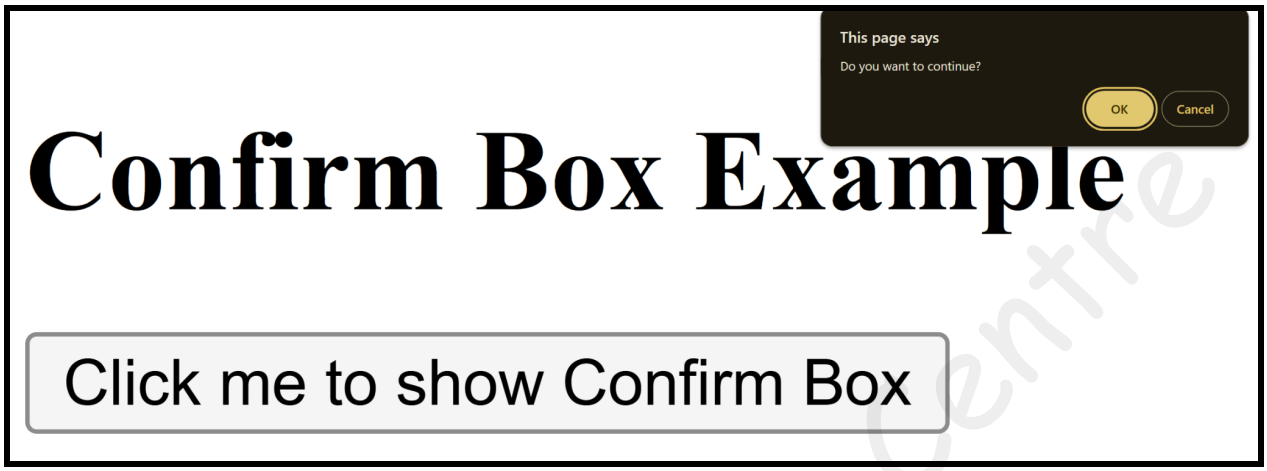
Syntax: confirm();

Input

```
<!doctype html>
<html>
<head>
  <title>Confirm Box Example</title>
</head>
<body>

<h2>Confirm Box Example</h2>
<button onclick="showConfirm()">Click me to show Confirm Box</button>

<script>
function showConfirm() {
  let result = confirm('Do you want to continue?');
  if (result) {
    alert('You clicked OK!');
  } else {
    alert('You clicked Cancel!');
  }
}
</script>

</body>
</html>
```

Baghel Computer Centre

Output



Baghel Computer Centre

3 Prompt Box

It asks the user to enter something. It returns the entered text or null if the user clicks Cancel.

Syntax: prompt();

Input

```
<!doctype html>
<html>
<head>
  <title>Prompt Box Example</title>
</head>
<body>

<button onclick="greetUser()">Click me</button>

<script>
function greetUser() {
  let name = prompt('What is your name?');
  if (name) {
    alert('Hello, ' + name + '!');
  } else {
    alert('You did not enter a name!');
  }
}
</script>

</body>
</html>
```

Bagul Computer Centre

Output

Baghel Computer Centre

ONMOUSEOVER / ONMOUSEOUT

Input

```
<!DOCTYPE html>
<html>
<head>

    <title>onmouseout Event Example</title>
</head>
<body>

<h2>onmouseout Event Example</h2>
<p id="mouseoutParagraph" onmouseover="onMouseOver()" onmouseout="onMouseOut()">Hover over and move out of this
text</p>

<script>
    function onMouseOver() {
        document.getElementById("mouseoutParagraph").style.color = "red";
    }

    function onMouseOut() {
        document.getElementById("mouseoutParagraph").style.color = "black";
    }
</script>

</body>
</html>
```

Output

Bagnel Computer Centre

ONLOAD

Input

```
<!DOCTYPE html>
<html>
<head>

    <title>onload Event Example</title>
</head>
<body onload="pageLoaded()">

<h2>onload Event Example</h2>
<p>The page has loaded.</p>

<script>
    function pageLoaded() {
        alert("Page has fully loaded!");
    }
</script>

</body>
</html>
```

Output

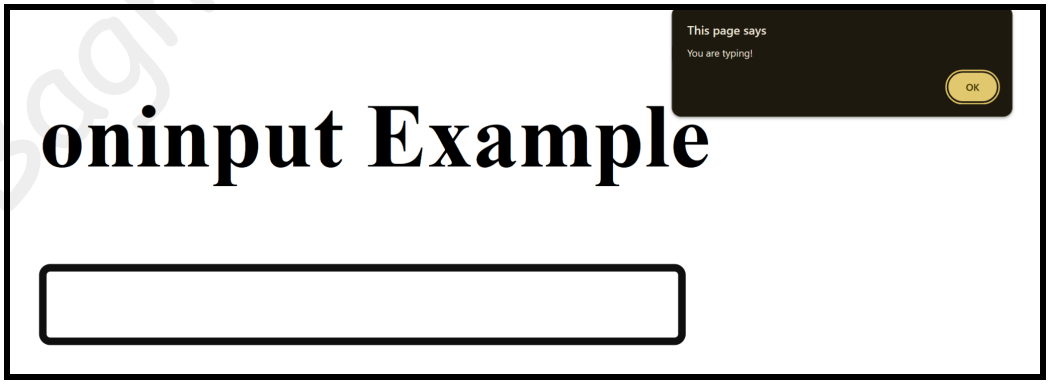
Baghel Computer Centre

Oninput

Input

```
<!doctype html>
<html>
<head>
  <title>oninput Event Example</title>
</head>
<body>

<h2>oninput Example</h2>
<input type="text" oninput="showInputMessage()">

<script>
function showInputMessage() {
  alert('You are typing!');
}
</script>

</body>
</html>
```

Output



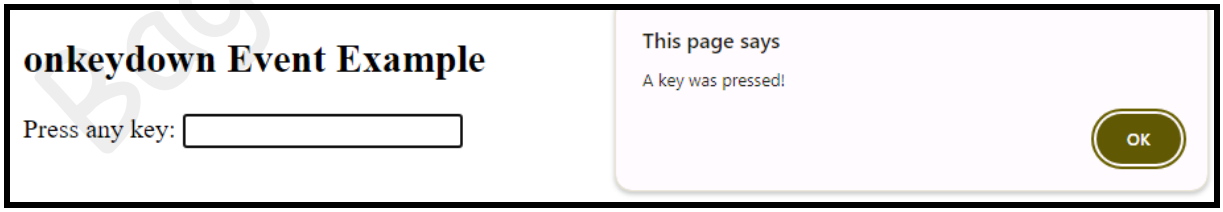
Baghel Computer Centre

ONKEYDOWN

Input

```
<!DOCTYPE html>
<html>
<head>

    <title>onkeydown Event Example</title>
</head>
<body>

<h2>onkeydown Event Example</h2>
<label for="keyInput">Press any key: </label>
<input type="text" id="keyInput" onkeydown="keyDownFunction()">

<script>
    function keyDownFunction() {
        alert("A key was pressed!");
    }
</script>

</body>
</html>
```

Output



Baghel Computer Centre

Conclusion

- Use onkeydown if you want to detect key presses, even when the value doesn't change.

- Use oninput if you want to detect actual content changes in the input field.

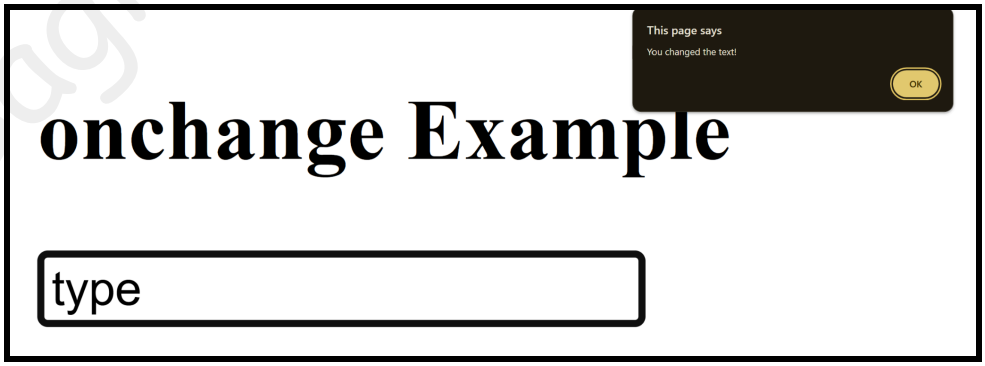
Baghel Computer Centre

Onchange

Input

```
<!doctype html>
<html>
<head>
  <title>onchange Event Example</title>
</head>
<body>

<h2>onchange Example</h2>
<input type="text" onchange="showChangeMessage()">

<script>
function showChangeMessage() {
  alert('You changed the text!');
}
</script>

</body>
</html>
```

Output



Baghel Computer Centre

Onfocus /onblur

Input

```
<!doctype html>
<html>
<head>

    <title>onfocus/onblur</title>
   </head>
<body>
    <input type="text" onfocus="showFocus()" onblur="showBlur()"  id="a">
    <script>
    function showFocus(){
    document.getElementById("a").style.backgroundColor="red";
    }
    function showBlur(){
    document.getElementById("a").style.backgroundColor="yellow";
    }
    </script>
</body>
</html>
```

Output

By click on input

By click outside input

Baghel Computer Centre

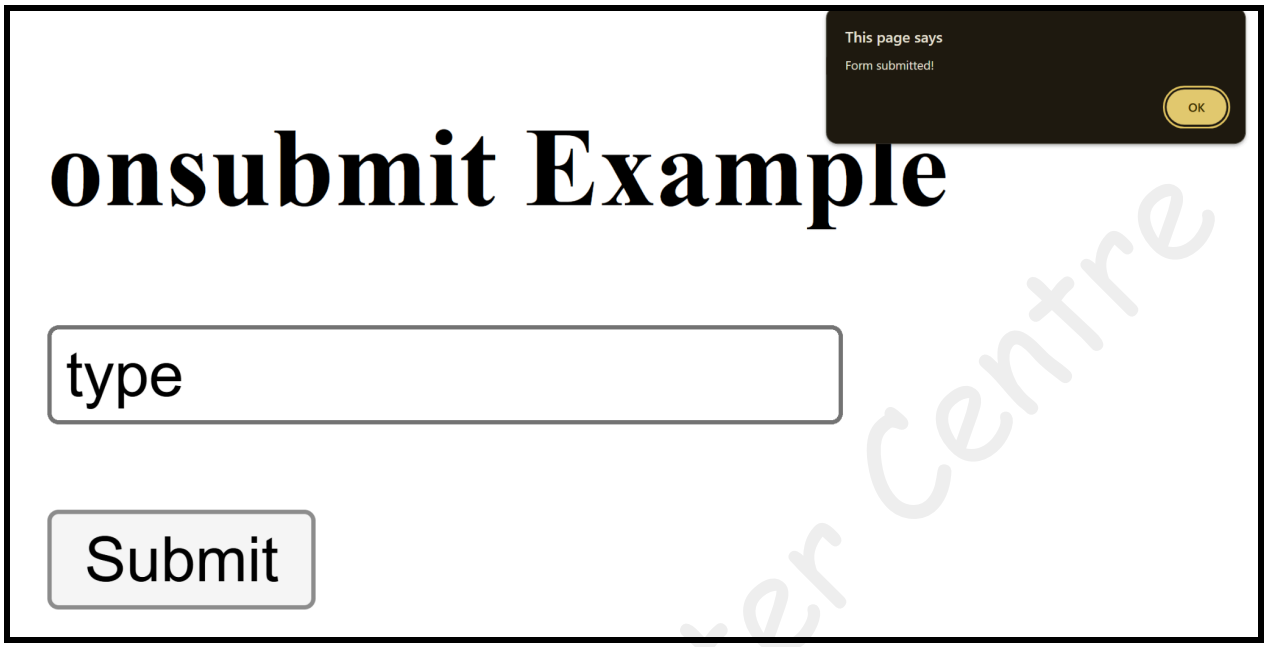
Onsubmit

Input

```
<!doctype html>
<html>
<head>
  <title>onsubmit Event Example</title>
</head>
<body>

<h2>onsubmit Example</h2>
<form onsubmit="showSubmitMessage()">
  <input type="text" placeholder="Type something">
  <br><br>
  <input type="submit" value="Submit">
</form>

<script>
function showSubmitMessage() {
  alert('Form submitted!');
}
</script>

</body>
</html>
```

Baghel Computer Centre

Output



Baghel Computer Centre