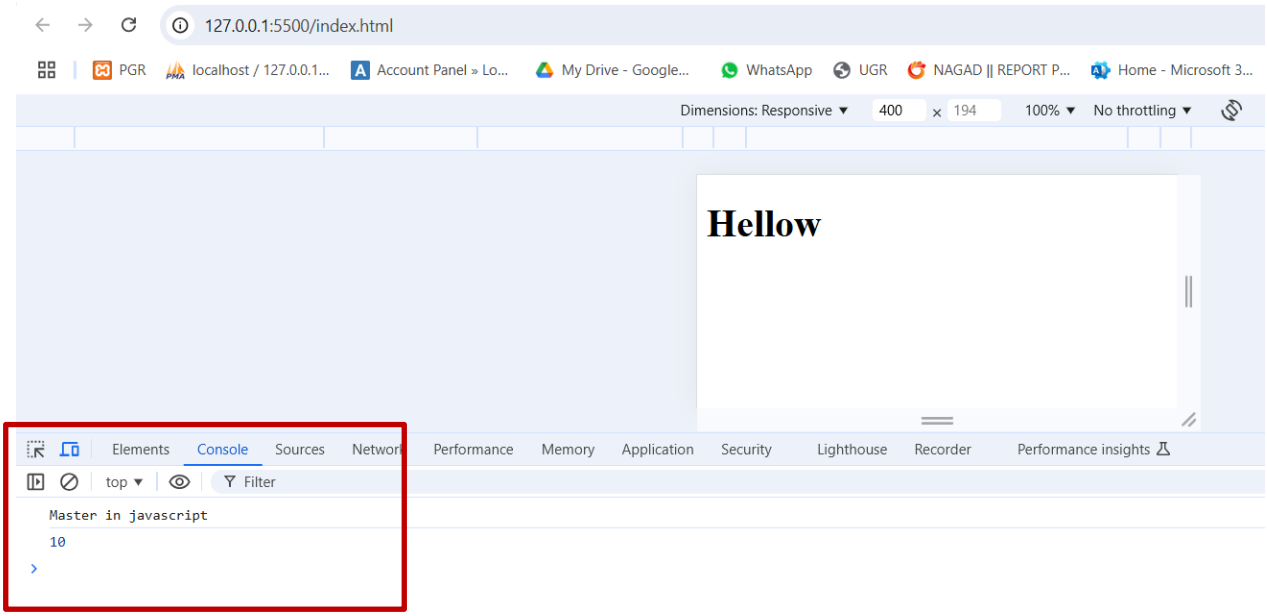
1. JAVASCRIPT Where To Write

(হেড এর মধ্যে লেখা যায়।)

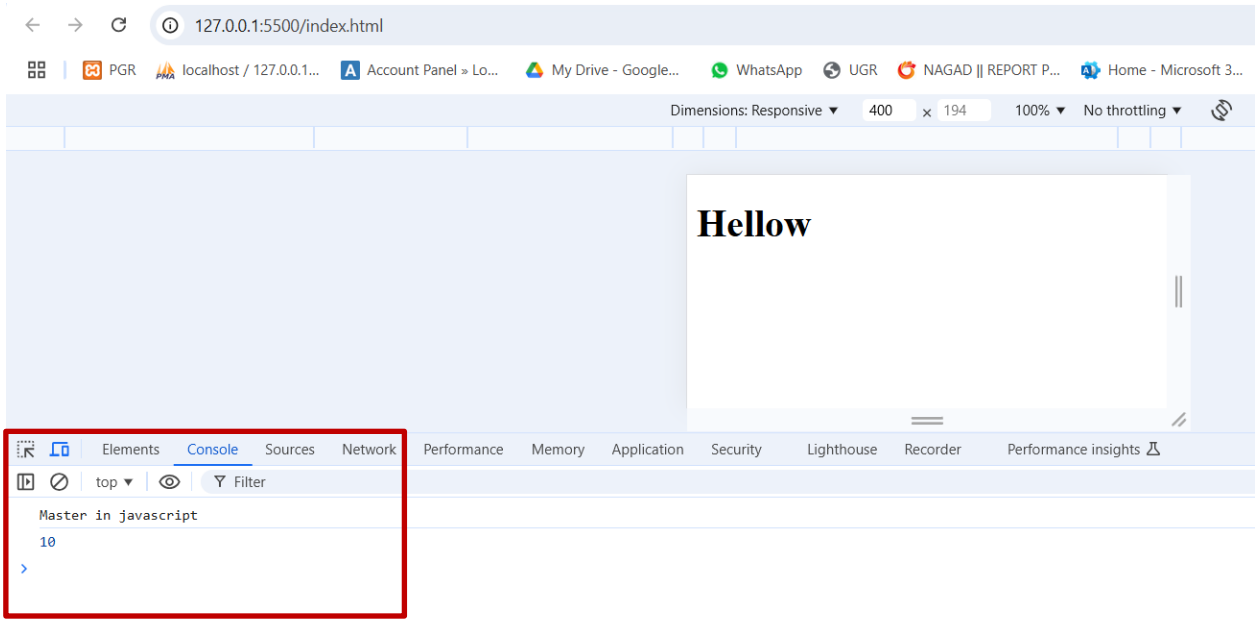
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        let course="Master in javascript";
        let duration=10;

        console.log(course);
        console.log(duration);
    </script>
</head>
<body>
    <h1>Hellow</h1>
</body>
</html>
```

ctrl+shift+i



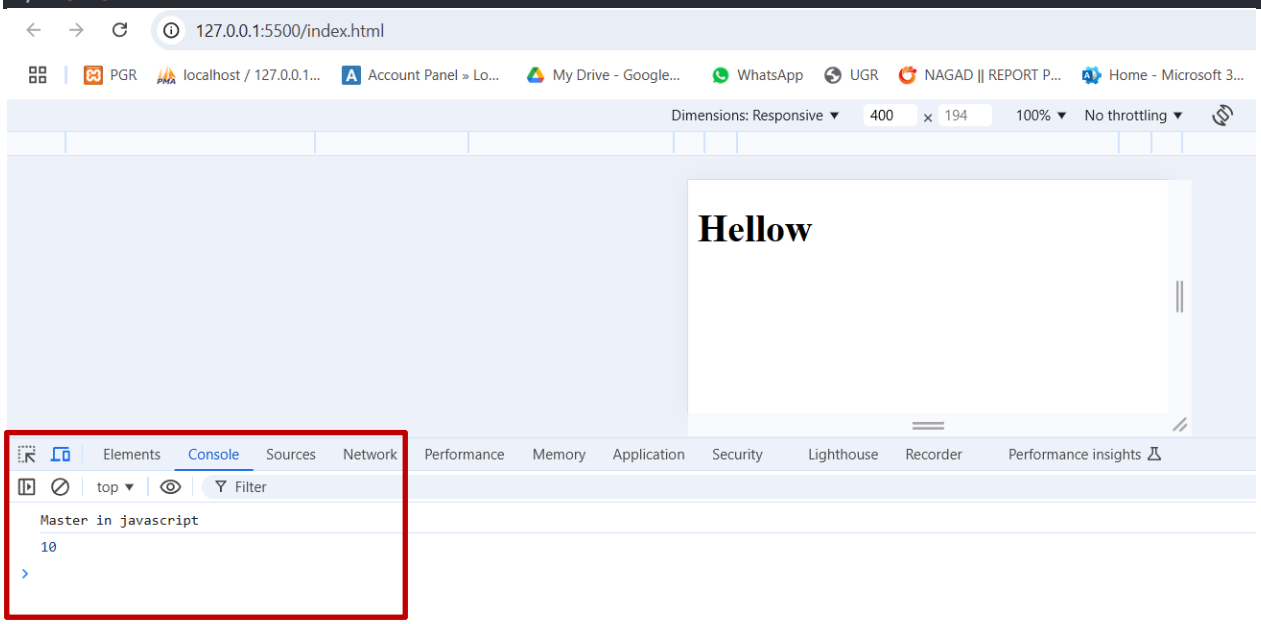
(বডি(body) এর মধ্যে লেখা যায়।)

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>

</head>
<body>
    <script>
        let course="Master in javascript";
        let duration=10;

        console.log(course);
        console.log(duration);
    </script>
    <h1>Hellow</h1>
</body>
</html>
```



(বডি(body) এর বাহিরে লেখা যায়। এটাই বেস্ট উপায়। কারণ এক্ষেত্রে বডি লোড হওয়ার পরে javascript লোড হয়।)

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>

</head>
<body>

    <h1>Hellow</h1>
</body>
<script>
    let course="Master in javascript";
    let duration=10;

    console.log(course);
    console.log(duration);
</script>
</html>
```



2.Variable Declaration

`var` (পুরো Function এর মধ্যে access পাবে। reassign করা যায় একই ব্লক এ।লুপ এ ব্যবহার)

- **Scope**: Function-scoped (accessible within the function or globally if declared outside a function).

`Let` (ব্লক এর মধ্যে access পাবে। reassign করা যায় না একই ব্লক এ।লুপ এ ব্যবহার করার জন্য উপযুক্ত।)

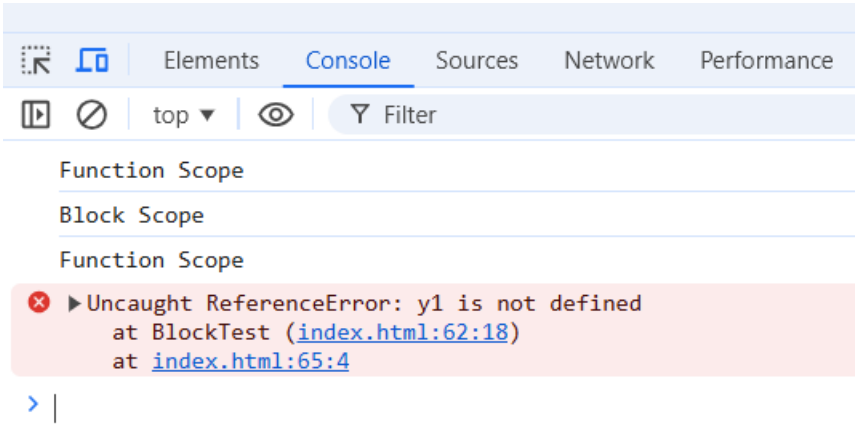
- **Scope**: Block-scoped (limited to the block, statement, or expression where it's declared).

`Const`(একবার assign হয়ে গেলে আর reassign করা যাবে না।)

- **Scope**: Block-scoped, similar to `let`.
- **Immutability**: The value can't be reassigned after initial assignment (though objects and arrays can still be modified).

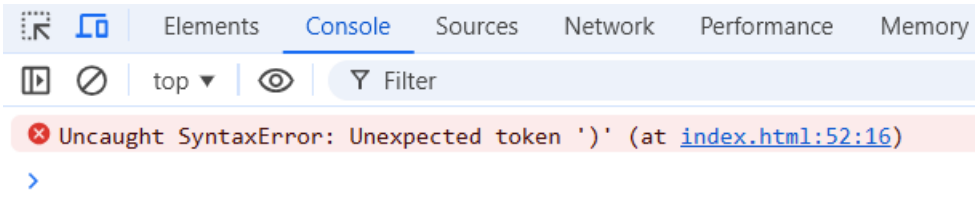
১. var এর scope হচ্ছে Function scope(block এর মধ্যে declare করলেও সম্পূর্ণ function এ এ্যাক্সেস পাবে) আর let এর scope হচ্ছে Block scope(যে Block declare করা হবে শুধু সে ব্লক এ এ্যাক্সেস করা যাবে।).

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   function BlockTest(){
     var x="Function Scope";// Function scope
     let y="Block Scope";// Block scope (limited to the function block)
     if(true){
        var x1="Function Scope";
        let y1="Block Scope";
     }
     console.log(x);//Accessible here Because Function Scoped
     console.log(y);//Accessible here (block-scoped in the function block)
     console.log(x1);//Accessible here Because Function Scope
     console.log(y1);//Error:y1 is accessible in only block only

   }
   BlockTest();
</script>
</html>
```



২.ReDeclaration করা যাবে var এ কিন্তু let এ ReDeclaration করা যাবে না।

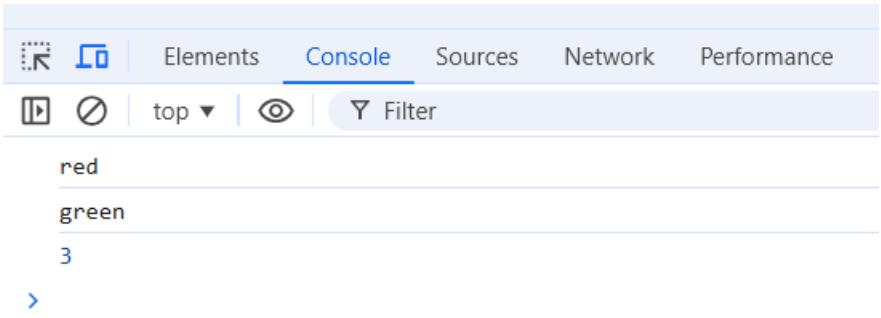
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>

</head>
<body>

</body>
<script>
   function BlockTest(){
     var x="hi";
     var x="hellow";
     let y=1;
     let y=2;//Error here

   }
   BlockTest();
</script>
</html>
```



3.ARRAY

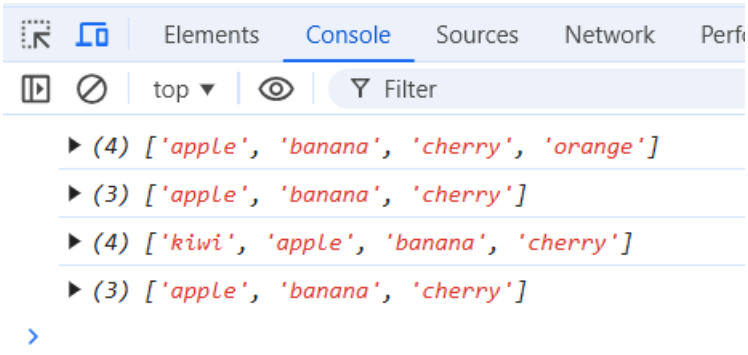
Array তৈরি(create) করা :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>

</head>
<body>


</body>
<script>
    //Using Array Literals:
    let fruits = ["apple", "banana", "cherry"];
    //Using the Array Constructor:
    let numbers = new Array(1, 2, 3, 4, 5);
    //Creating an Empty Array:
    let emptyArray = [];
    //Use indices to access elements. Remember, indexing starts at 0
    let colors = ["red", "green", "blue"];
    console.log(colors[0]); // "red"
    console.log(colors[1]); // "green"
    //length: Returns the number of elements in an array
    console.log(fruits.length); // 3
 </script>
</html>
```



Array Operation

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
    //push/add: element to array
    let fruits = ["apple", "banana", "cherry"];
    fruits.push("orange");
    console.log(fruits);

    //pop:Removes the last element from the array
    fruits.pop();
    console.log(fruits);

    //unshift: Adds an element to the beginning of the array.
    fruits.unshift("kiwi");
    console.log(fruits);

    //shift: Removes the first element from the array.
    fruits.shift();
    console.log(fruits);

 </script>
</html>
```

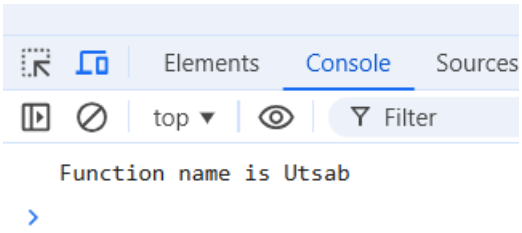


4.Function

Syntax:

```
function functionName(parameters) {
    // code to be executed
}
```

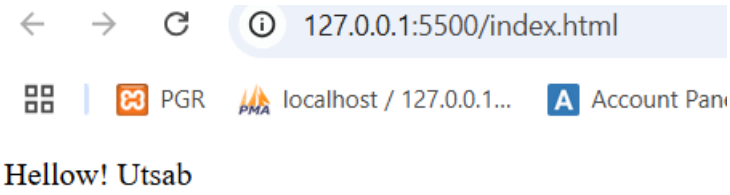
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   function showName(name){
        console.log("Function name is "+name);
   }
   showName("Utsab");
 </script>
</html>
```



Anonymous Function(যে ফাংশন এর নাম নাই)

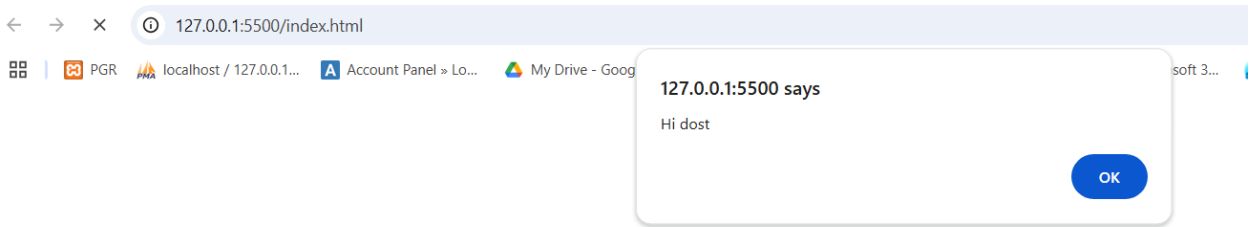
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
```

```
</body>
<script>
   let greet=function(name){
        return "Hellow! "+name;
   }
   document.write(greet("Utsab"));
 </script>
</html>
```



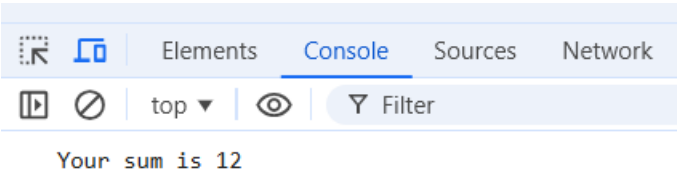
Arrow Function

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   (() => {
           alert("Hi dost");
       })();
 </script>
</html>
```



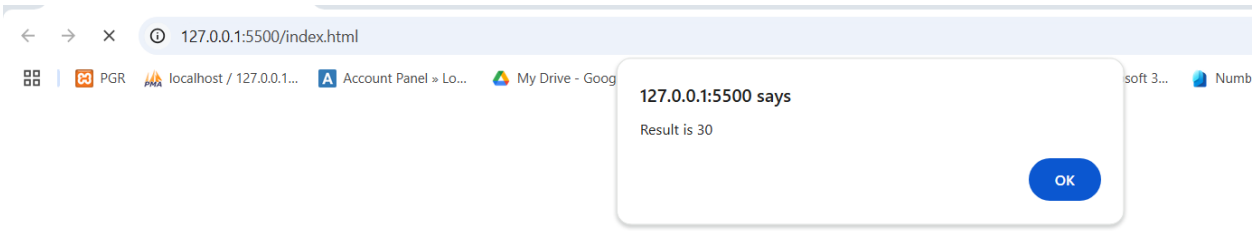
ফাংশন থেকে VALUE RETURN করা:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   function multiply(a,b){
        return "Your sum is "+ a*b;
   }
   console.log(multiply(3,4));
 </script>
</html>
```



Arrow Functions(shorter syntax)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   let sum_show=(a,b)=>{
    alert("Result is "+ (a+b));
   }
   sum_show(10,20);
 </script>
</html>
```



5. display variable's value

Variable এর জায়গায় value replace হওয়াকে বলে string interpolation

Using Concatenation with the + Operator

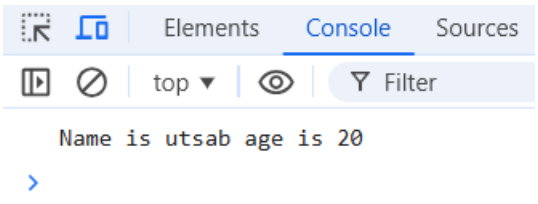
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
   let name='utsab';
   let age=20;
   let message="Name is "+name+" age is "+age;
   console.log(message);
 </script>
</html>
```

Using Template Literals (ES6+)

Single quotation(Backtrick symbol) এবং ${} ব্যবহার করতে হবে:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
```

```
<script>
   let name='utsab';
   let age=20;
   let message=`Name is ${name} age is ${age}`;
   console.log(message);
 </script>
</html>
```



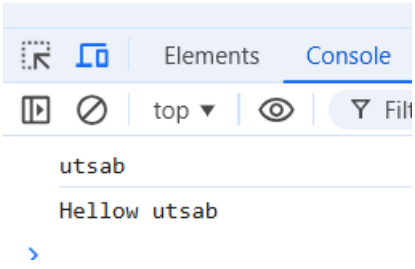
Link:

https://eloquentjavascript.net/

6.Non Primitive Data Type

Object(class create করার মত:)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
    let person={
        name : "utsab",
        age : 25,
        professtion : "Programmer",
        greet: function(){
            return "Hellow "+this.name;
        }
    };
    console.log(person.name);
    console.log(person.greet());
 </script>
</html>
```



Map

Set

Extension:

Eslint(ভুল হলে সেটা ঠিক করে দেয়।)

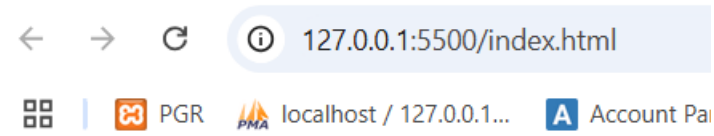
Github Copilot(অটোম্যাটিক suggestion দিবে।)

Branching Technique:

If-else

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
```

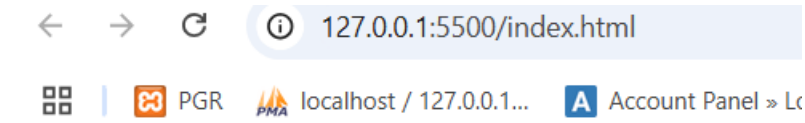
```
</body>
<script>
  let age=15;
  if(age>=18){
    document.write('You are eligible to vote');
  }else{
    document.write('You are not eligible to vote');
  }
 </script>
</html>
```



You are not eligible to vote

SWITCH

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
    let mark=70;
    switch(mark/10){
        case 9:
        case 8:
            document.write('A+');
            break;
        case 7:
            document.write('A');
            break;
        default:
            document.write('Fail');
    }

 </script>
</html>
```

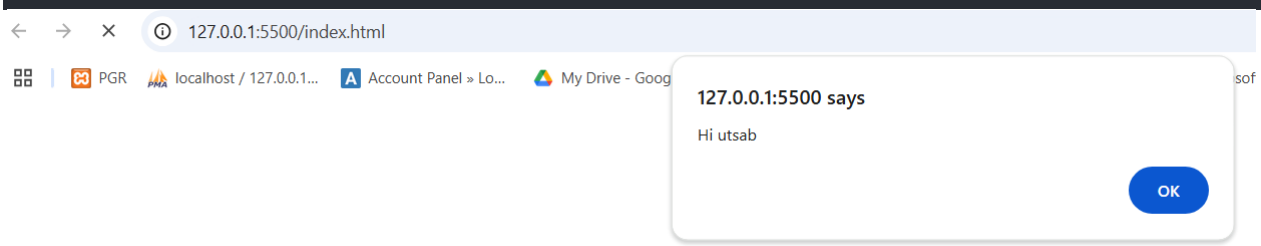


A

Short-circuit operators

Logical AND (&&) - Conditional Execution

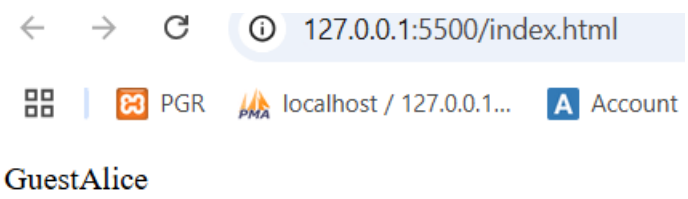
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
<script>
    let clicked=true;
    clicked && alert('Hi utsab');

 </script>
</html>
```



Logical OR (||) - Default Value or Fallback

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>

</body>
```

```
<script>
    let user = null;
    let defaultName = "Guest";

    // Using || to assign a default value
    let name = user || defaultName;
    document.write(name); // Output: "Guest"

    // If user is not null, it would take the user's value
    user = "Alice";
    name = user || defaultName;
    document.write(name); // Output: "Alice"
</script>

</html>
```

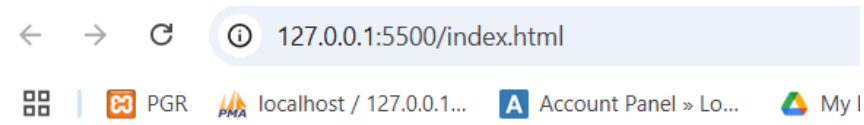


Loop:

For / while / do-while

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>

</body>
<script>

    // For loop
    for (let i = 1; i <= 3; i++) {
        document.write(`For Loop ${i}<br>`);
    }
```

```
    // While loop
    let j = 1;
    while (j <= 2) {
        document.write(`While Loop ${j}<br>`);
        j++;
    }

    // Do-while loop
    let k = 1;
    do {
        document.write(`Do-while Loop ${k}<br>`);
        k++;
    } while (k <= 2);
</script>

</html>
```

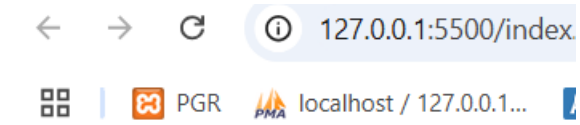


For Loop 1
For Loop 2
For Loop 3
While Loop 1
While Loop 2
Do-while Loop 1
Do-while Loop 2

for...of loop for...in loop forEach loop

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>

</body>
<script>
   //The for...of loop is used to iterate over iterable objects like arrays or strings.
   const fruits=["apple","mangoe","orange"];
   for (const single_fruit of fruits){
        document.write(single_fruit+'<br>');
   }
```

```
   //The for...in loop iterates over the keys of an object.
   const object={name:"John",age:30,city:"New York"};
   for(const key in object){
        document.write(`key is ${key} value is ${obj[key]}<br>`);
   }

</script>

</html>
```



apple
mangoe
orange
key is name value is John
key is age value is 30
key is city value is New York