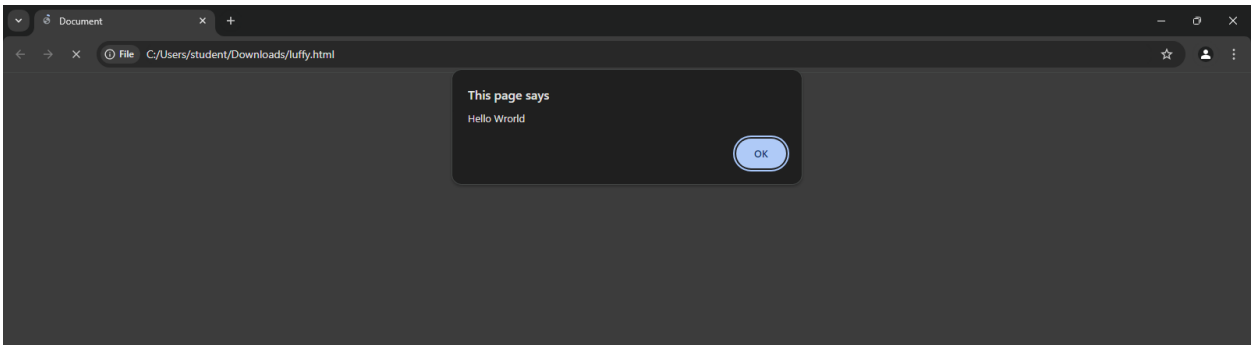
Task 1: Write a simple script that displays “Hello, World!” on the web page using

an alert box.

```
<!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>
        alert("Hello Wrorld");
    </script>

</body>
</html>
```

Output:



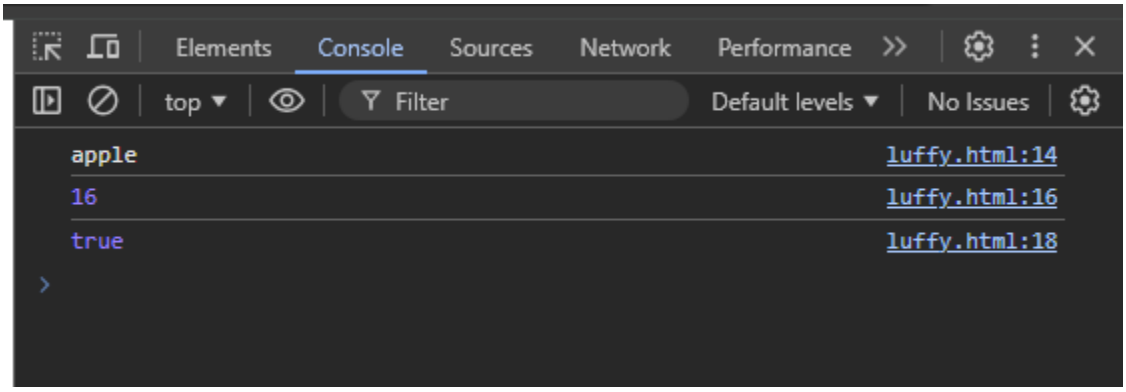
Task 2: Experiment with different data types in JavaScript (e.g., string, number,

boolean) by declaring and logging them in the console.

```
<!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>
        var  j ="apple";
        console.log(j);
        var  k =16;
        console.log(k);
        var  l=true ;
        console.log(l);

    </script>

</body>
</html>
```

Output:



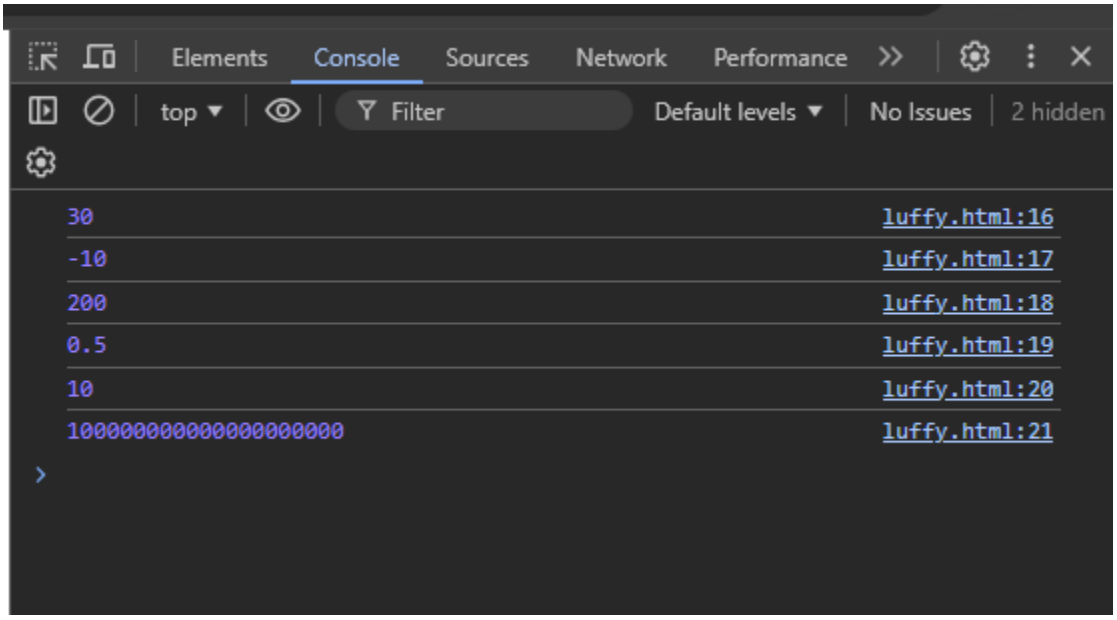
Task 3: Use the console to perform basic math operations like addition,

subtraction, multiplication, and division

```
      <!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>
        var a=10;
        var b=20;
        console.log(a+b);
        console.log(a-b);
        console.log(a*b);
        console.log(a/b);
        console.log(a%b);
        console.log(a**b);

    </script>


</body>
</html>
```

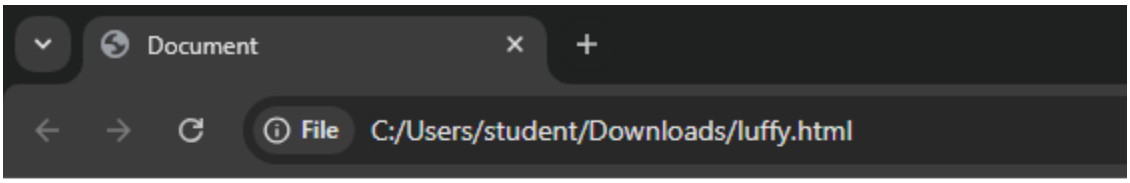
Output:



Task 4: Declare two strings and concatenate them using 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>
    </script>
    <script>
    var x="dragon";
    var y="fly";
    var z=x+y;
    document.writeln(z);
</script>

</body>
</html>
```

Output:



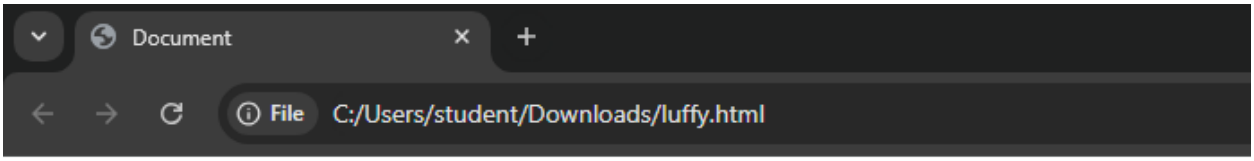
dragonfly

Task 5: Use the typeof operator to check the data type of various variables.

```
<!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>
    var m=10;
    var n="apple";
    var o=true;
    var p;
    document.writeln(m+"<br>");
    document.writeln(n+"<br>");
    document.writeln(o+"<br>");
    document.writeln(p+"<br>");
</script>
</body>
</html>
```

Output:



10
apple
true
undefined

Task 6: Write a multi-line JavaScript comment and a single-line comment.

Explain the difference

```
<!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>
    //This is a single line comment

    /*This is the multi line
    comment which is used to make
    the difference between single and multi line comment*/
</script>
</script>

</body>
</html>
```

Single-line Comment (`//`):

- Starts with `//` and comments out the rest of the line.
- Used for short, inline comments or quick explanations on a single line.

Multi-line Comment (`/* */`):

- Starts with `/*` and ends with `*/`, allowing comments to span multiple lines.
- Suitable for longer descriptions, explanations, or temporarily disabling multiple lines of code.

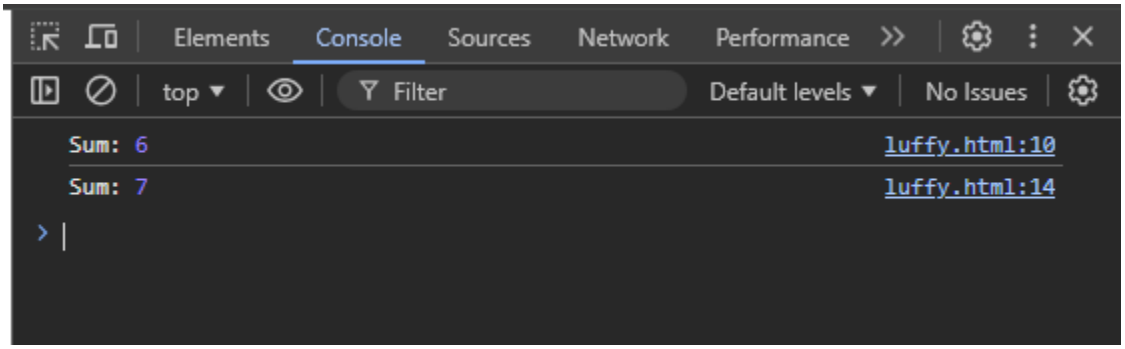
Task 7: Create a script with both semicolon-separated and not separated lines.

Note any differences in behavior.

```
<!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 x = 5; let y = 1; let z = x + y; console.log("Sum:", z);
let a = 2;
let b = 5;
let c = a + b;
console.log("Sum:", c);


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

Output:



DIFFERENCE:

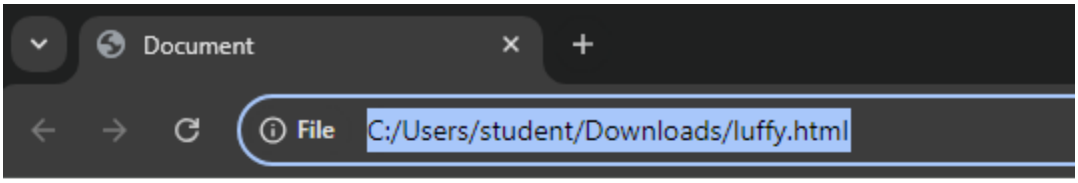
The only difference is while using separated semicolons it increases the clarity more than without seperating

Task 8: Use proper indentation to format a nested 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>
    <script>
 for(let a=0;a<5;a++)
 {
    document.writeln("<br>");
    for(let b=0;b<5;b++)
 {
    document.writeln("{"+a+","+b+"}");
 }
```

```
 }

</script>
</script>

</body>
</html>
```

Output:



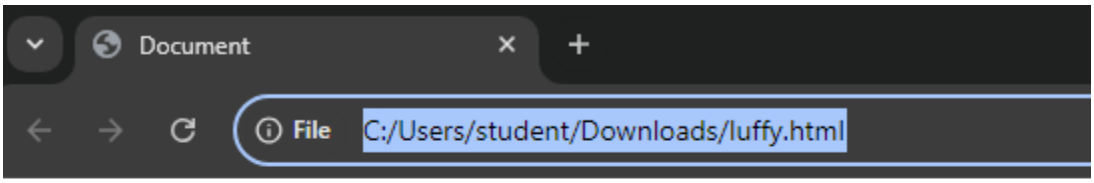
{0,0} {0,1} {0,2} {0,3} {0,4}
{1,0} {1,1} {1,2} {1,3} {1,4}
{2,0} {2,1} {2,2} {2,3} {2,4}
{3,0} {3,1} {3,2} {3,3} {3,4}
{4,0} {4,1} {4,2} {4,3} {4,4}

Task 9: Declare multiple variables in a single line.

```
<!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 a,b,c;
        a=0;
        b=9;
        c=5;
        document.writeln(a+"<br>"+b+"<br>"+c);

</script>
</script>

</body>
</html>
```

Output:



0
9
5

Task 10: Place a script tag at the top and bottom of an HTML document. Note

any differences in behavior

AT TOP:

```
<!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>
```

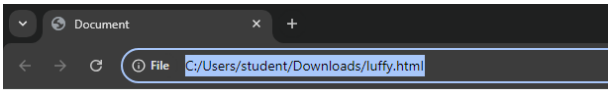
```
        let a,b,c;
        a=0;
        b=9;
        c=5;
        document.writeln(a+"<br>"+b+"<br>"+c);

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

AT BOTTOM:

```
<!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 a,b,c;
        a=0;
        b=9;
        c=5;
        document.writeln(a+"<br>"+b+"<br>"+c);

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

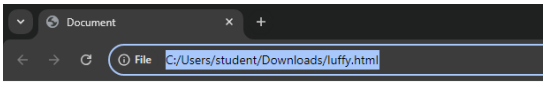
output:

AT TOP:



0
9
5

AT BOTTOM:



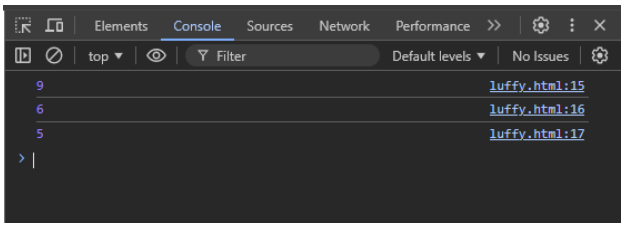
0
9
5

Task 16: Declare variables using let, const, and var. Discuss when each should be

used.

```
<!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 x = 5;
x=9;
var a = 2;
var a=6;
const b = 5;
console.log(x);
console.log(a);
console.log(b);
</script>
</body>
</html>
```

Output:



VAR:

Redeclareable and reassignable

LET:

Can not be redecleared but reassignable

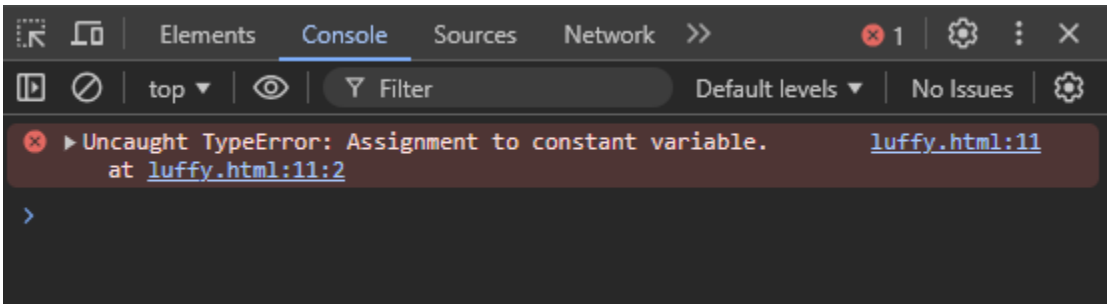
CONST:

Can not be redeclared and can not be reassigned

Task 17: Attempt to reassign a const variable and observe the result

```
<!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>
const b = 5;
b=1;
console.log(b);
</script>
</body>
</html>
```

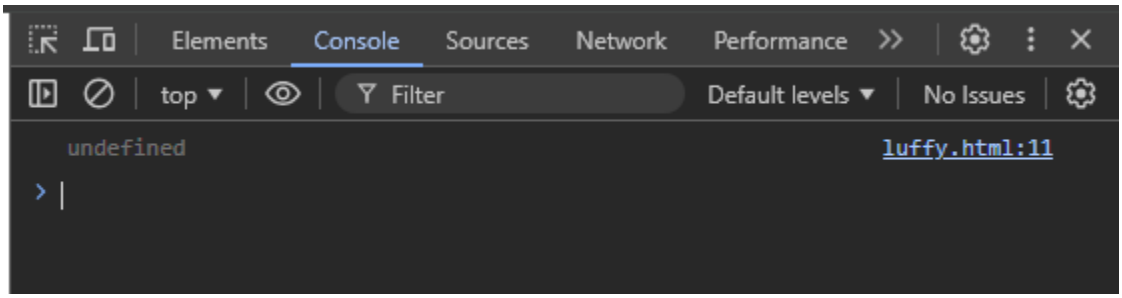
Output:



Task 18: Declare a variable without initializing it and print its value.

```
<!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 b;
console.log(b);
</script>
</body>
</html>
```

Output:



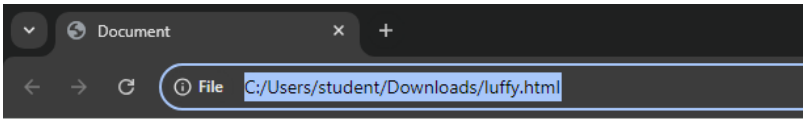
Task 19: Assign a number, string, and boolean value to a variable and print its

type using typeof.

```
<!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 a =1;
```

```
let b="apple";
let c=true;
document.writeln(typeof a);
document.writeln(typeof b);
document.writeln(typeof c);

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

output:

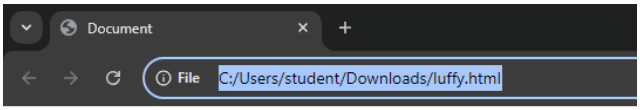


number string boolean

Task 20: Rename a variable and observe the outcome.

```
<!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>
var a=1;
var a=5;
document.writeln(a);
</script>
</body>
</html>
```

Output:



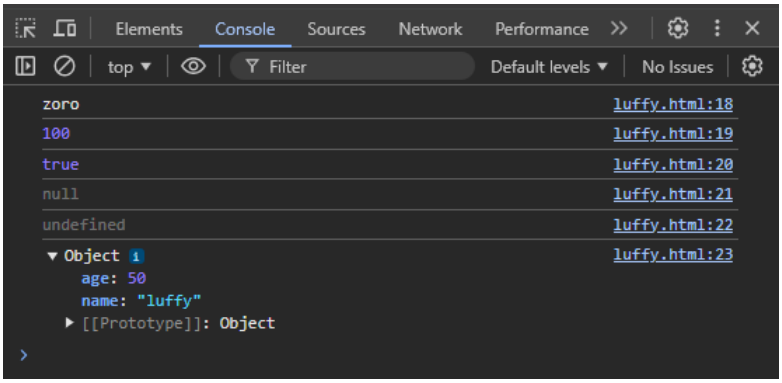
5

Task 21: Create variables of different data types (e.g., string, number, boolean,

null, undefined, object).

```
<!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 a = "zoro"
let b = 100
let c = true
let d = null
let e = undefined
let students={
   name:`luffy`,age:50
}
console.log(a)
console.log(b)
console.log(c)
console.log(d)
console.log(e)
console.log(students)
</script>
</body>
</html>
```

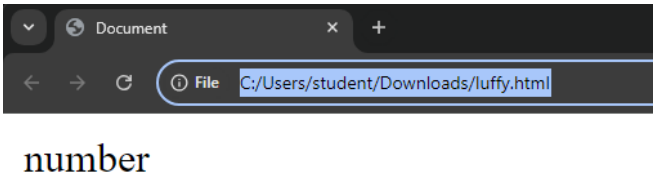
Output:



Task 22: Use the typeof operator to determine the type of various variables.

```
<!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>
var a=1;
document.writeln(typeof a);
</script>
</body>
</html>
```

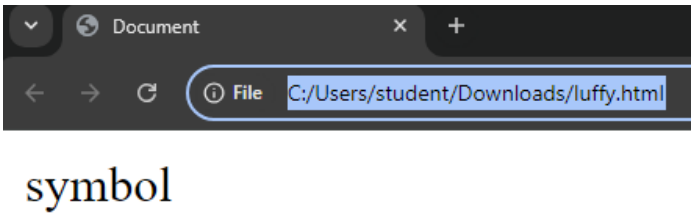
Output:



Task 23: Declare a symbol and print its type**.**

```
<!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>
var a=Symbol("zoro");
document.writeln(typeof a);
</script>
</body>
</html>
```

Output:

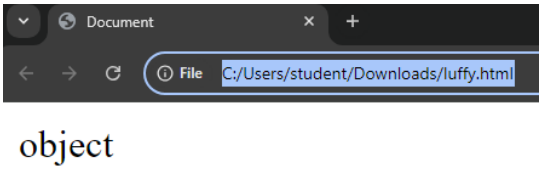


Task 24: Assign the value null to a variable and check its type using typeof.

```
<!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 d = null
document.writeln(typeof d);
```

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

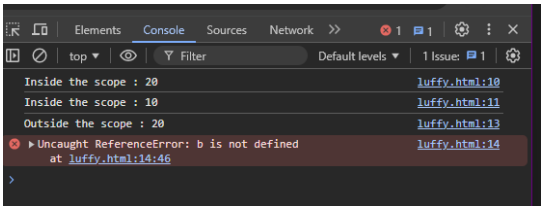
Output:



Task 25: Differentiate between declaring a variable using var and let in terms of scope.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
        {
            var a = 20;
            let b = 10;
            console.log("Inside the scope : " + a);
            console.log("Inside the scope : " + b);
        }
        console.log("Outside the scope : " + a);
        console.log("Outside the scope : " + b);
    </script>
</body>
</html>
```

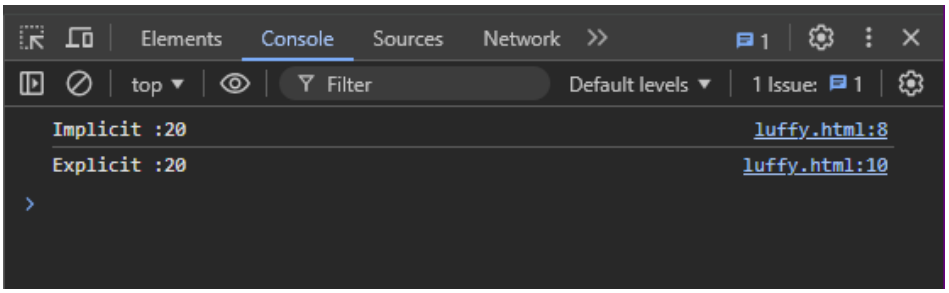
Output:



Task 26: Convert a string to a number using both implicit and explicit conversion.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
        var x="20";
        console.log("Implicit :"+(x-0));
        var y=Number("20");
        console.log("Explicit :"+y);
    </script>
</body>
</html>
```

Output:

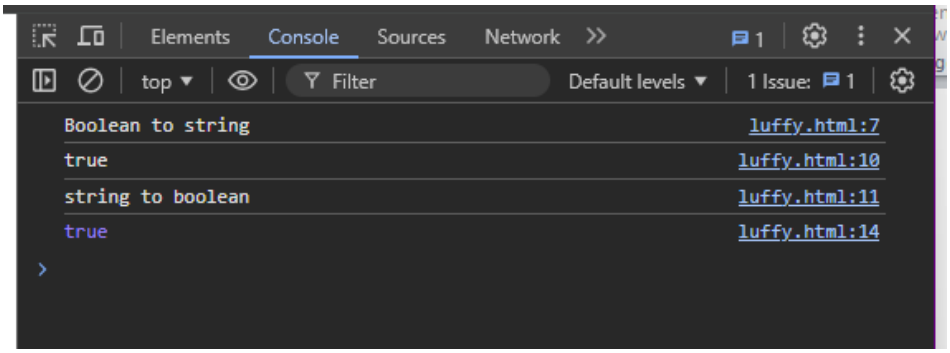


Task 27: Convert a boolean to a string and vice versa.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
        console.log("Boolean to string");
        var a=true;
        var x=String(a);
        console.log(x);
        console.log("string to boolean");
        var b="false";
        var y=Boolean(b);
        console.log(y);
    </script>
```

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

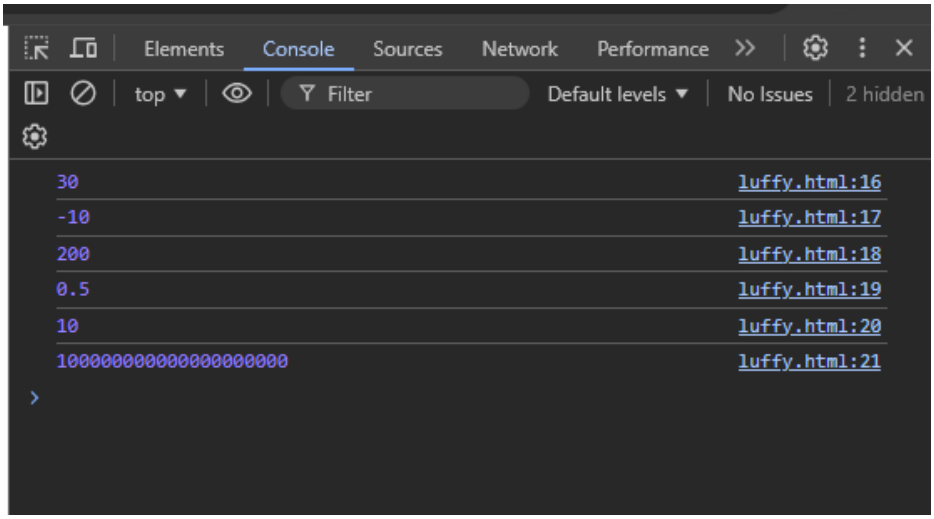
Output:



Task 28: Practice basic arithmetic operators (+, -, *, /, %).

```
      <!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>
        var a=10;
        var b=20;
        console.log(a+b);
        console.log(a-b);
        console.log(a*b);
        console.log(a/b);
        console.log(a%b);
        console.log(a**b);

    </script>


</body>
</html>
```

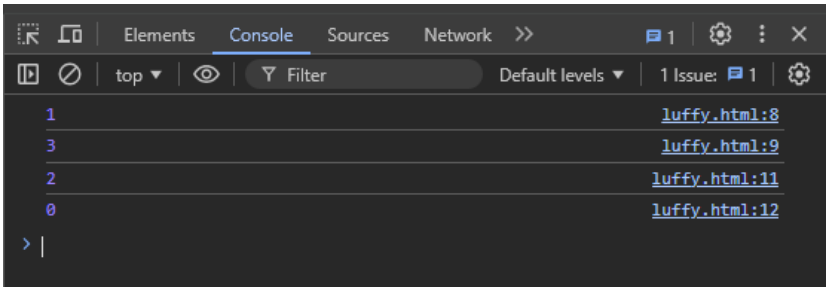
Output:



Task 29: Use the ++ and -- operators on a numeric variable.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
        var a=1;
        console.log(a++);
        console.log(++a);
        var b=2;
        console.log(b--);
        console.log(--b);

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

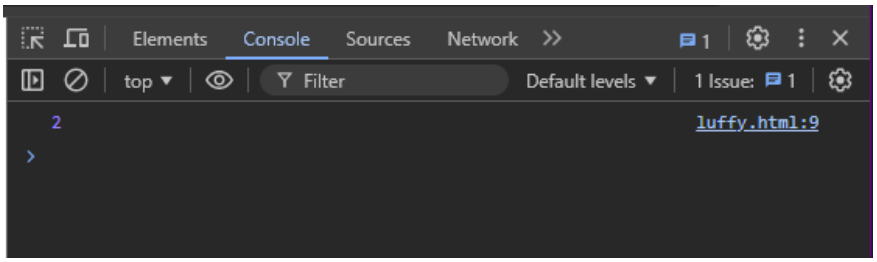
Output:



Task 30: Explore the precedence of operators by combining multiple operators in single expression.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
        var a=1;
        var b=2;
        console.log(a+b-b*a/b);

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

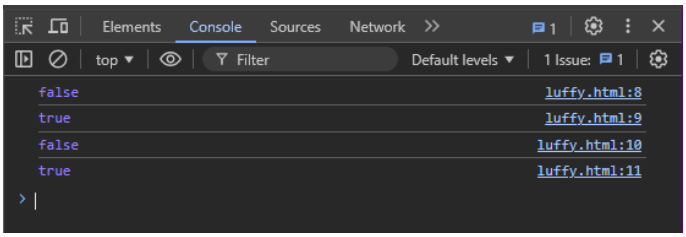
output:



Task 31: Compare two numbers using relational operators (>, <, >=, <=).

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a=5,b=7;
       console.log(a>b);
       console.log(a<b);
       console.log(a>=b);
       console.log(a<=b);
    </script>
</body>
</html>
```

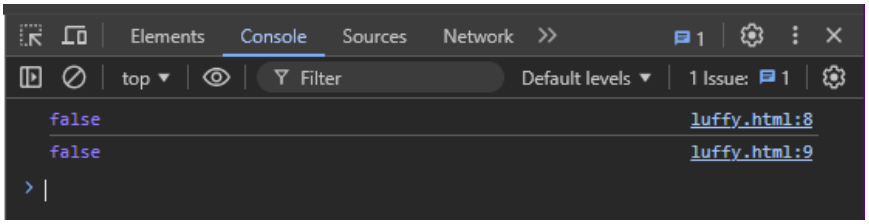
Output:



Task 32: Use equality () and strict equality (=) operators to compare different data types and note the differences.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a=5,b=7;
       console.log(a==b);
       console.log(a===b);
    </script>
</body>
</html>
```

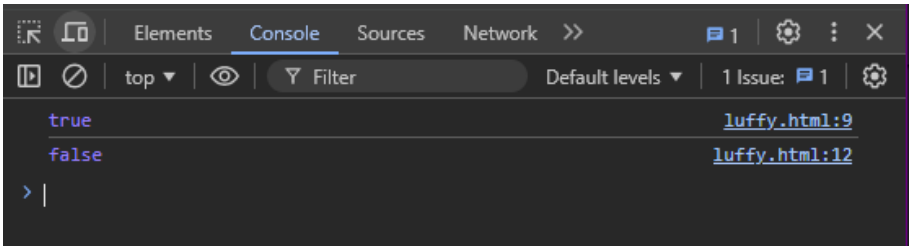
Output:



Task 33: Compare two strings lexicographically.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a="hello";
       var b="hello";
       console.log(a==b);
       var x="world";
       var y="World";
       console.log(x==y);
    </script>
</body>
</html>
```

Output:



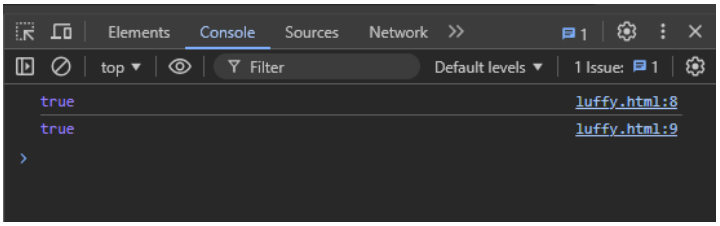
Task 34: Use the inequality (!=) and strict inequality (!==) operators to compare

values.

```
<html>
<head>
    <title>Document</title>
</head>
```

```
<body>
    <script>
       var a=5,b=7;
       console.log(a!=b);
       console.log(a!==b);
    </script>
</body>
</html>
```

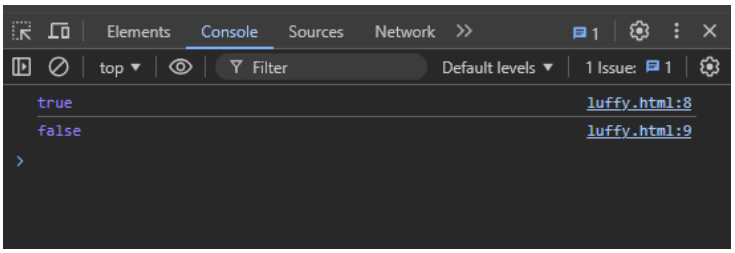
Output:



Task 35: Compare null and undefined using both == and ===.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a=null,b=undefined;
       console.log(a==b);
       console.log(a===b);
    </script>
</body>
</html>
```

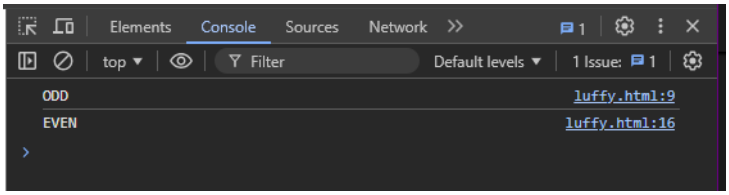
Output:



Task 36: Write an if statement that checks if a number is even or odd.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a=3;
       if(a%2)
       console.log("ODD");
       else
       console.log("EVEN");
       var b=4;
       if(b%2)
       console.log("ODD");
       else
       console.log("EVEN");

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

Output:



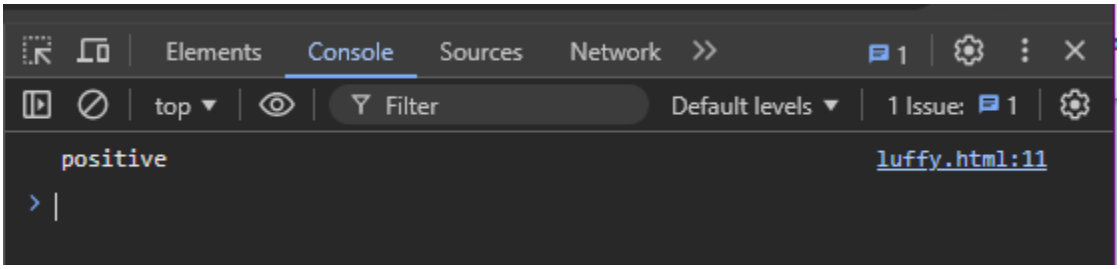
Task 37: Use nested if statements to classify a number as negative, positive, or

zero.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
       var a="hello";
       var b="hello";
       console.log(a==b);
       var x="world";
       var y="World";
       console.log(x==y);
```

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

OUTPUT:

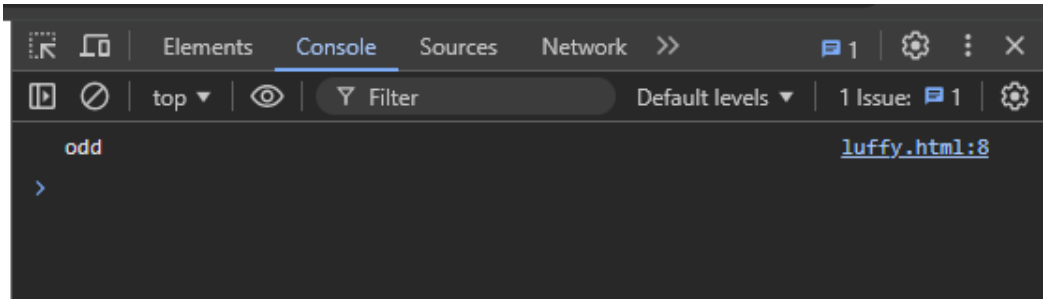


Task 38: Use the conditional (ternary) operator '?' to rewrite a simple if...else

statement.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
      var a=5;
      var b=(a%2==0)?console.log("even"):console.log("odd");
    </script>
</body>
</html>
```

Output:

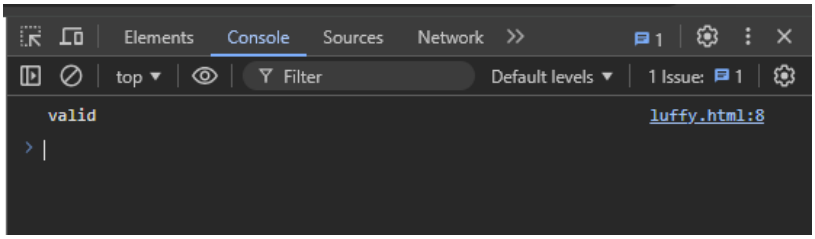


Task 39: Check the validity of a variable using the ? operator.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
```

```
      var a=5;
      var b=(a)?console.log("valid"):console.log("not valid");
    </script>
</body>
</html>
```

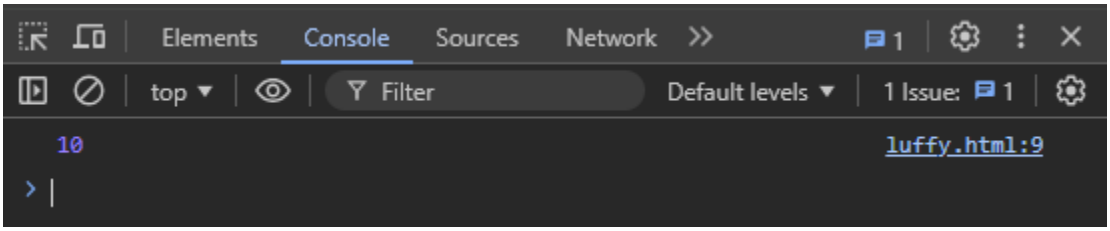
Output:



Task 40: Use the conditional operator to assign a value to a variable based on a condition.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
      var a=5;
      var b=(a==1)?5:10;
      console.log(b);
    </script>
</body>
</html>
```

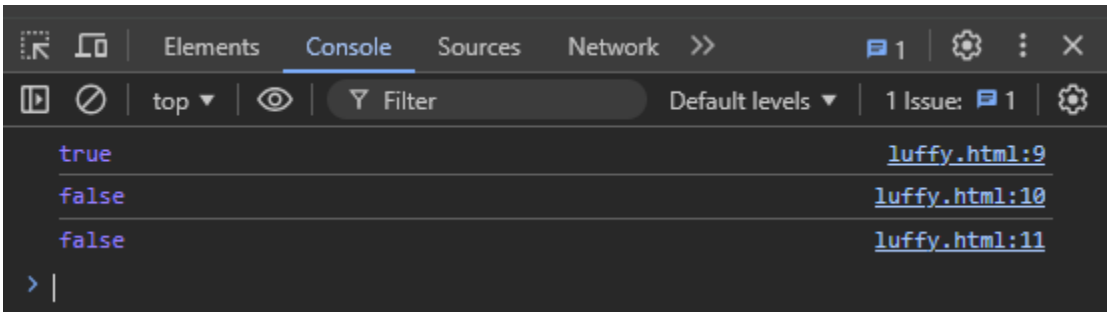
Output:



Task 41: Evaluate various combinations of logical operators (&&, ||, !).

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
      var a=5;
      var b=10;
      console.log((a<b)&&(b==10));
      console.log((a>b)||((b==5)));
      console.log(!b);
    </script>
</body>
</html>
```

Output:



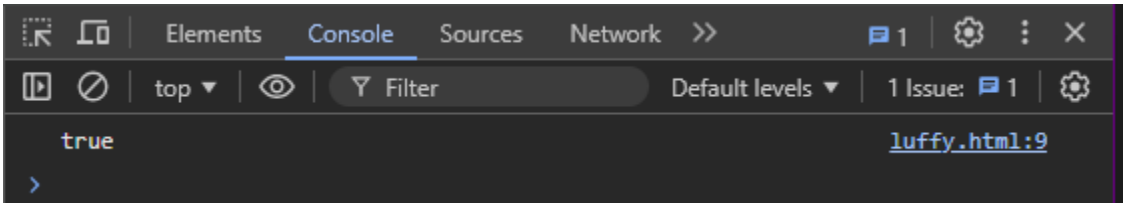
Task 42: Use logical operators to write a condition that checks if a number is in a

given range.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
      var a=5;
      if(a>0&&a<100)
      console.log("true");
    else
    console.log("false");
    </script>
</body>
```

```
</html>
```

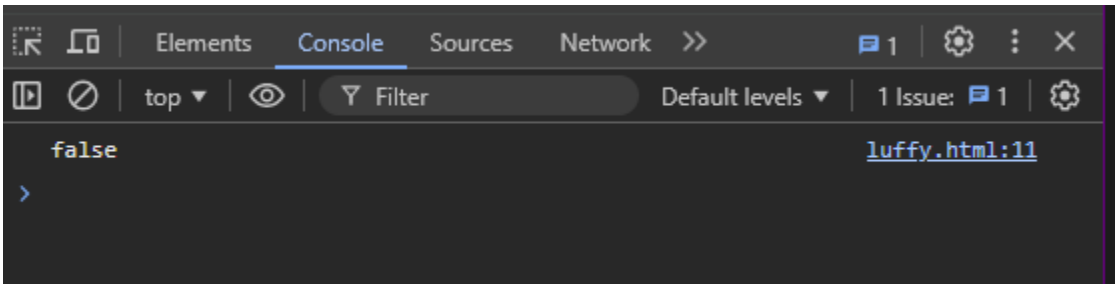
Output:



Task 43: Use the NOT (!) operator to invert a boolean value.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
      var a=true;
      if(!a)
      console.log("true");
    else
    console.log("false");
    </script>
</body>
</html>
```

Output:

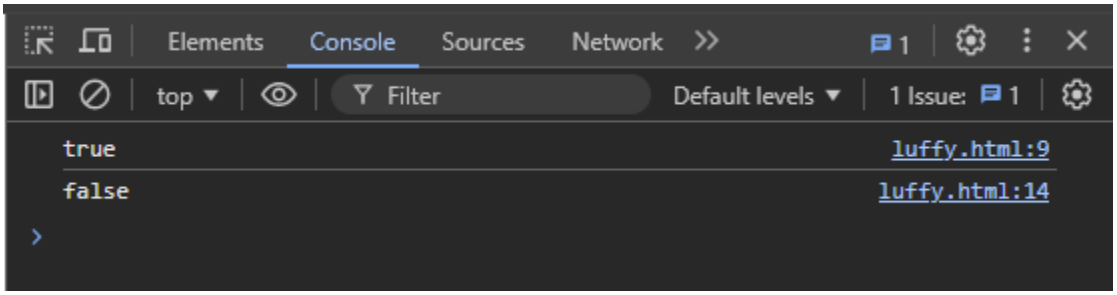


Task 44: Evaluate the short-circuiting nature of logical operators.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
```

```
      if(true||false||false)
      console.log("true");
      else
      console.log("false");
      if(false && true && true)
      console.log("true");
      else
      console.log("false");
    </script>
</body>
</html>
```

Output:

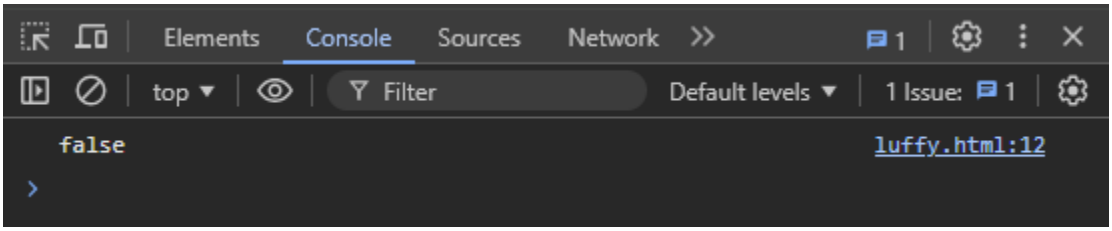


Task 45: Compare two non-boolean values using logical operators and observe

the result.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>

      var a=9,b=0;
      if(a==b)
      console.log("true");
      else
      console.log("false");
    </script>
</body>
</html>
```

output:

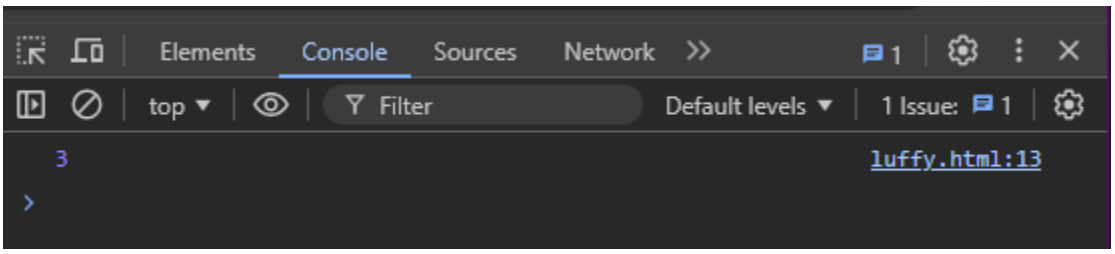


Task 46: Write a function that takes two numbers as arguments and returns their

sum.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
     function sum(a,b){
     return a+b;
     }
    var a=1;
    var b=2;
    var r=sum(a,b);
    console.log(r);
    </script>
</body>
</html>
```

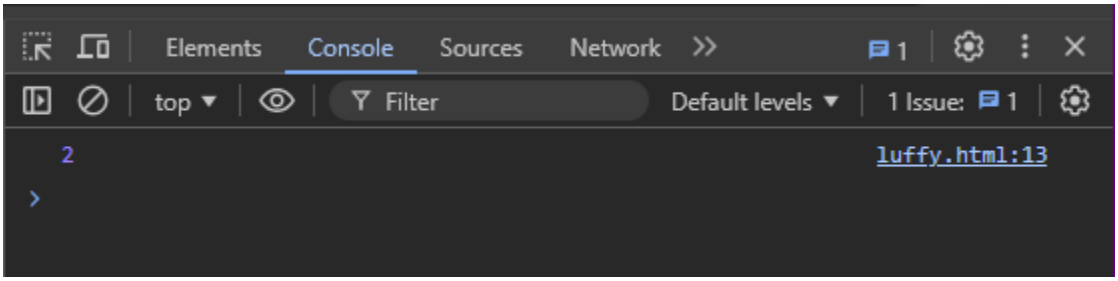
Output:



Task 47: Create a function that calculates the area of a rectangle.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
     function area(a,b){
     return a*b;
     }
    var a=1;
    var b=2;
    var r=area(a,b);
    console.log(r);
    </script>
</body>
</html>
```

Output:

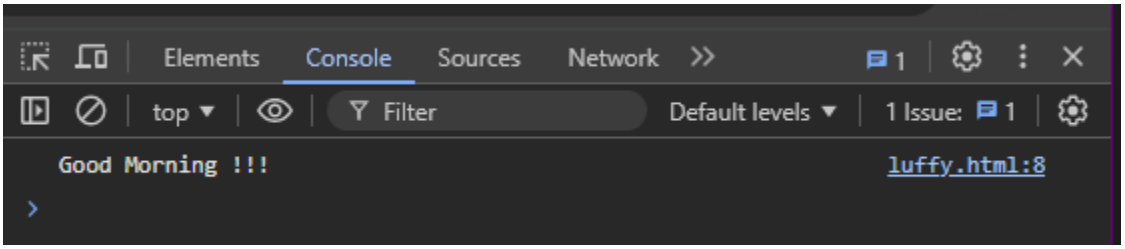


Task 48: Declare a function without parameters and call it.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
     function display(){
     console.log("Good Morning !!!");
    }
   display();
    </script>
</body>
```

```
</html>
```

Output:

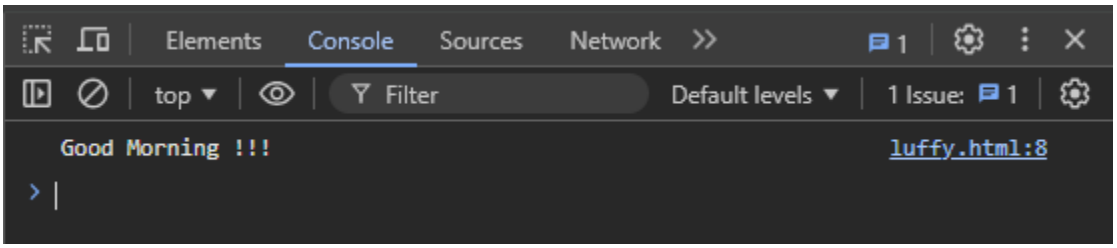


Task 49: Write a function that returns nothing and observe the default return

value.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
     function display(){
     console.log("Good Morning !!!");
    }
   var a=display();
   console.log(a);
    </script>
</body>
</html>
```

output:

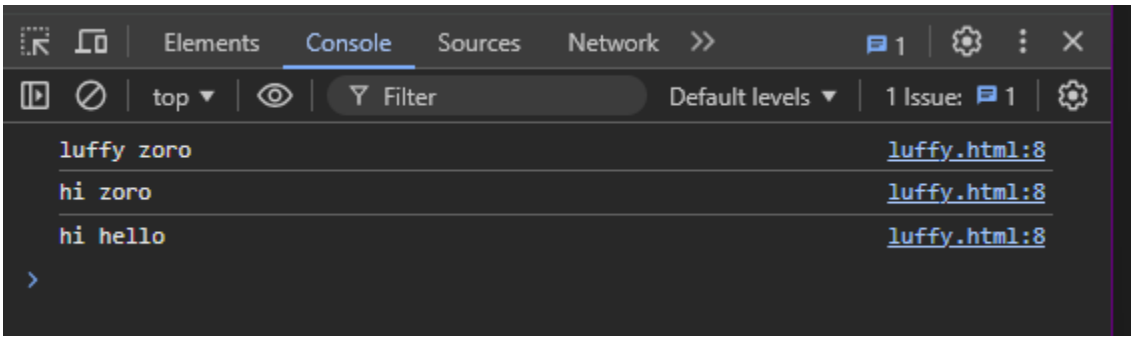


Task 50: Declare a function with default parameters and call it with different

arguments.

```
<html>
<head>
    <title>Document</title>
</head>
<body>
    <script>
     function display(a="luffy",b="zoro"){
     console.log(a+" "+b);
    }
   display();
   display("hi");
   display("hi","hello");
    </script>
</body>
</html>
```

Output:

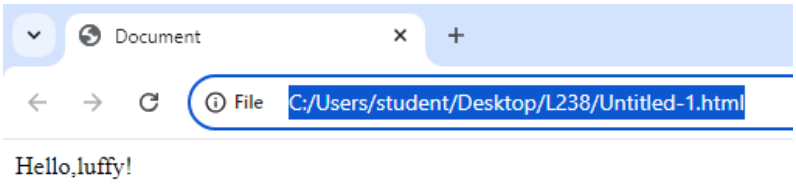


Task 51: Declare a simple arrow function named greet that takes one parameter name and returns the string “Hello, name!”. Test your function with various names.

```
<!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>
       var greet = (name) => "Hello,"+name+"!";
       document.writeln(greet("luffy"));
    </script>
</body>
</html>
```

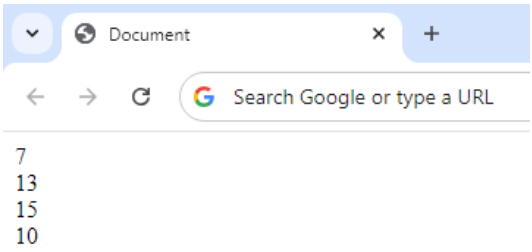
Output:



Task 52: Write an arrow function named add that takes two parameters and returns their sum. Validate your function with several pairs of numbers.

```
<!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>
       var add = (a,b) => {
        return a+b;
       };
       document.writeln(add(5,2)+"<br>");
       document.writeln(add(9,4)+"<br>");
       document.writeln(add(14,1)+"<br>");
       document.writeln(add(8,2)+"<br>");

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

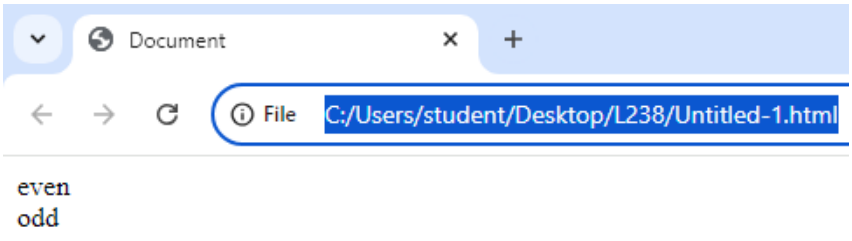
Output:



Task 53: Declare an arrow function named isEven that checks if a number is even. If the number is even, it should return true; otherwise, false. Remember that if the arrow function body has a single statement, you can omit the curly braces.

```
<!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>
       var iseven = (a)
=>  (a%2==0)?document.writeln("even"):document.writeln("odd");
      iseven(2);
      document.writeln("<br>");
      iseven(5);
       </script>
</body>
</html>
```

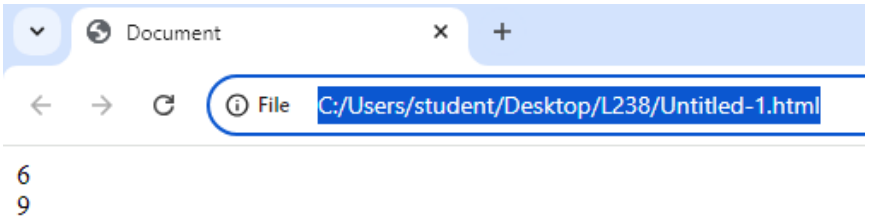
Output:



Task 54: Implement an arrow function named maxValue that takes two numbers as parameters and returns the larger number. Here, you'll need to use curly braces for the function body and the return statement.

```
<!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>
       var maxvalue = (a,b) =>  {
        if(a>b)
        return a;
        else
        return b;
       }
      document.writeln(maxvalue(5,6)+"<br>");
      document.writeln(maxvalue(2,9));


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

Output:



Task 55: Examine the behavior of the this keyword inside an arrow function vs a traditional function. Create an object named myObject with a property value set to 10 and two methods: multiplyTraditional using a traditional function and multiplyArrow using an arrow function. Both methods should attempt to multiply the value property by a number passed as a parameter. Check the value of this inside both methods.

```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 myobject ={
        value : 10,
        multiplyTraditional : function(n) {
            return this.value * n;
        },
        multiplyArrow : (n) => {
            return this.value * n;
      }};
      console.log(myobject.multiplyTraditional(2));
      console.log(myobject.multiplyArrow(2));
       </script>
</body>
</html>
```

Output: