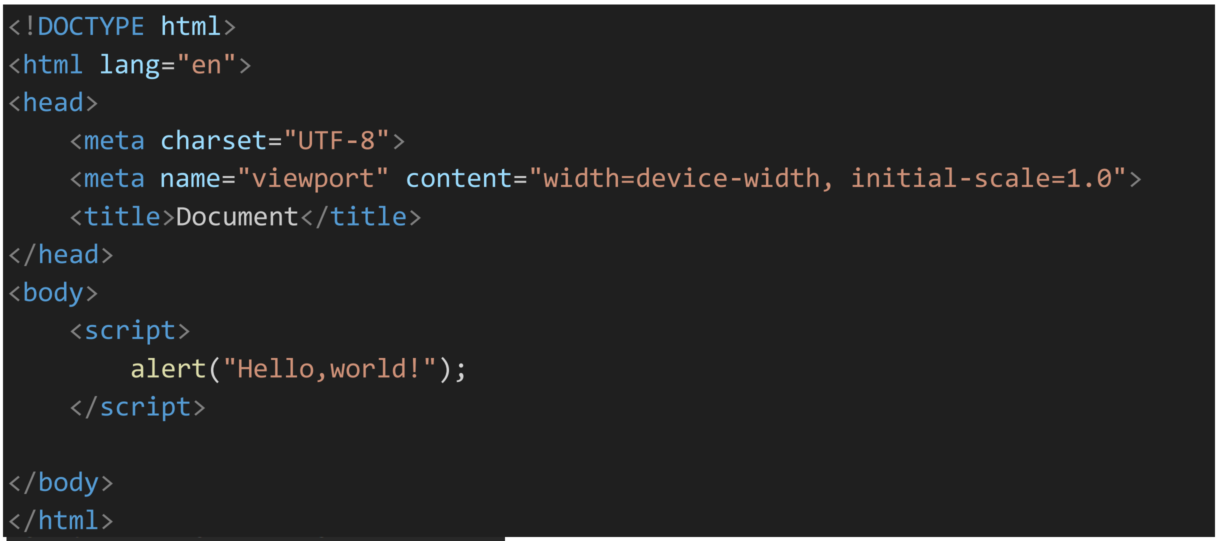
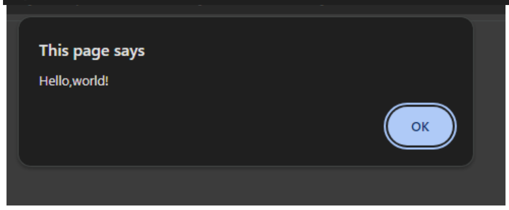
JAVASCRIPT LANGUAGE

1.AN INTRODUCTION TO JAVASCIPT

:TASK1:



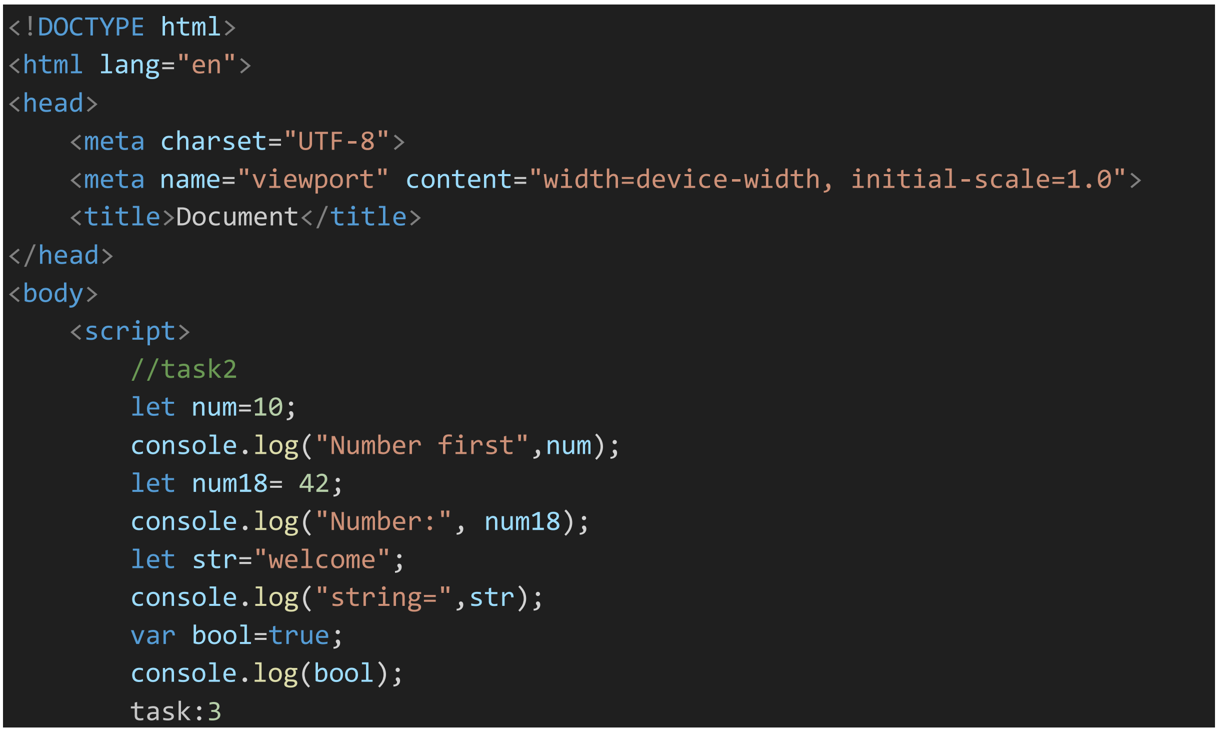
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        alert("Hello,world!");
    </script>

</body>
</html>
```



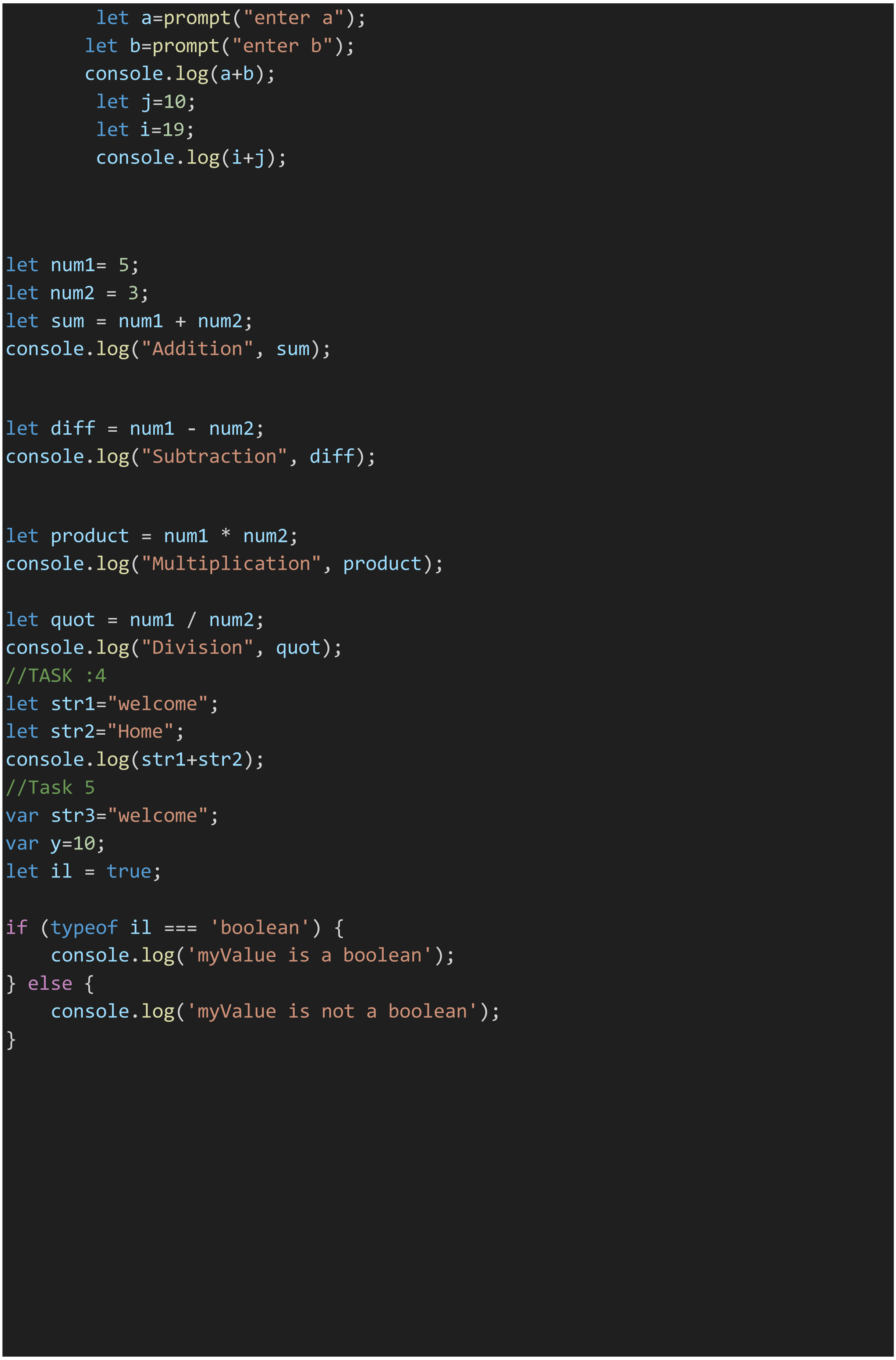
This page says

Hello,world!

TASK :2,3,4,5;

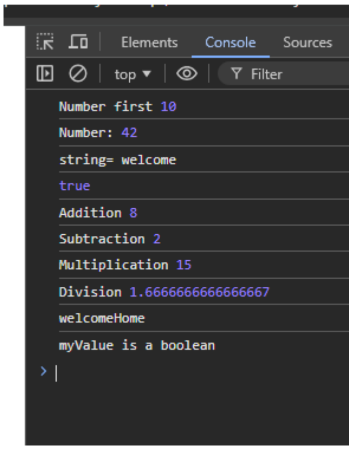


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        //task2
        let num=10;
        console.log("Number first",num);
        let num18= 42;
        console.log("Number:", num18);
        let str="welcome";
        console.log("string=",str);
        var bool=true;
        console.log(bool);
        task:3
```



```
        let a=prompt("enter a");
       let b=prompt("enter b");
       console.log(a+b);
        let j=10;
        let i=19;
        console.log(i+j);



let num1= 5;
let num2 = 3;
let sum = num1 + num2;
console.log("Addition", sum);


let diff = num1 - num2;
console.log("Subtraction", diff);


let product = num1 * num2;
console.log("Multiplication", product);

let quot = num1 / num2;
console.log("Division", quot);
//TASK :4
let str1="welcome";
let str2="Home";
console.log(str1+str2);
//Task 5
var str3="welcome";
var y=10;
let il = true;

if (typeof il === 'boolean') {
    console.log('myValue is a boolean');
} else {
    console.log('myValue is not a boolean');
}
```

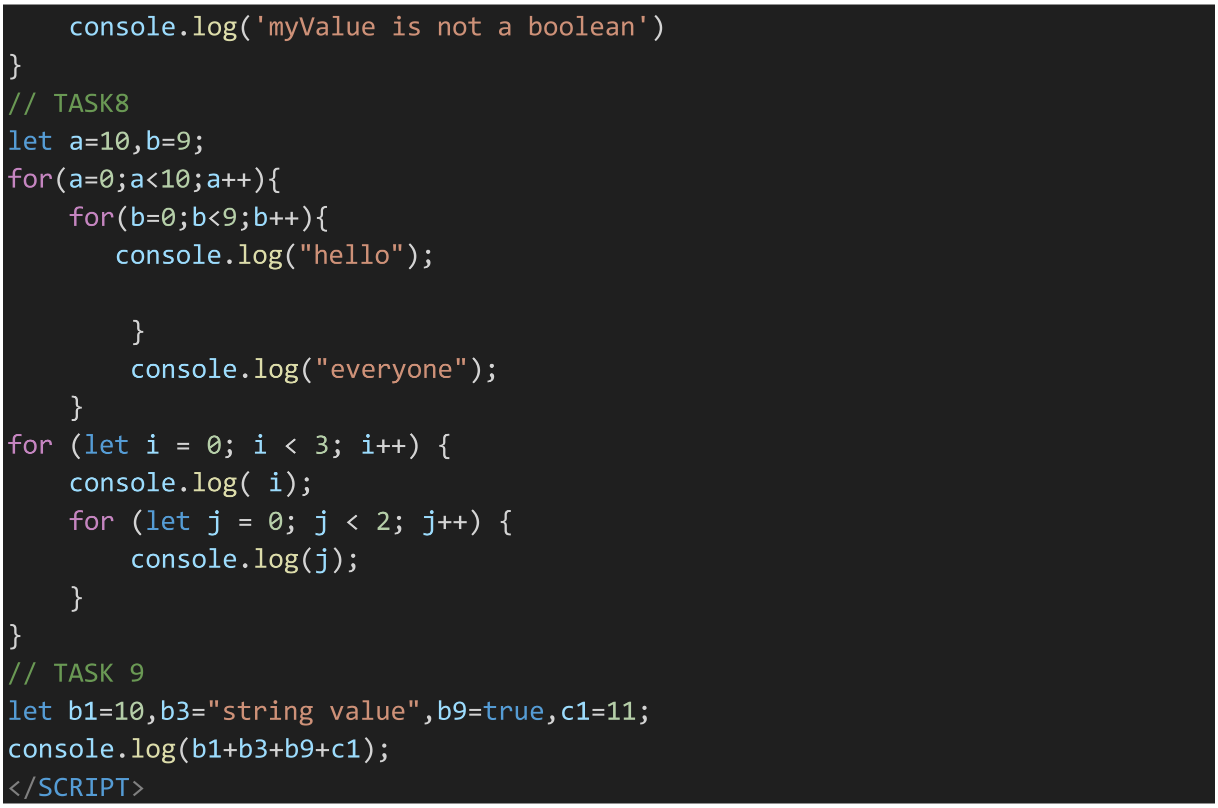
```
    </script>

</body>
</html>
```



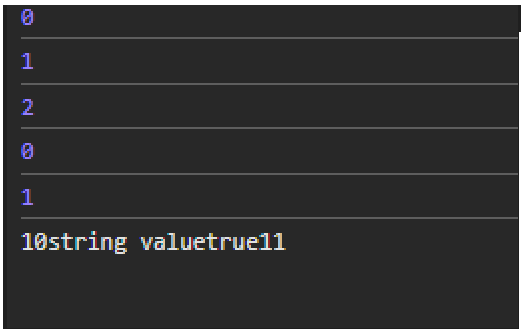
```
Number first 10
Number: 42
string= welcome
true
Addition 8
Subtraction 2
Multiplication 15
Division 1.6666666666666667
welcomeHome
myValue is a boolean
```

TASK 6,7,8,9,10:

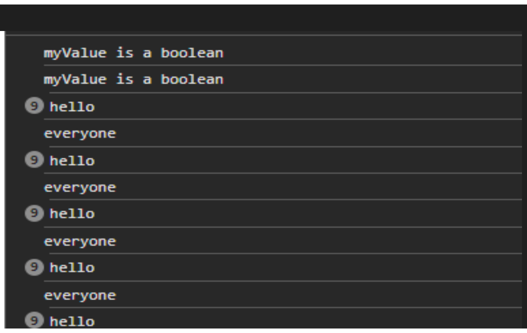
```
<SCRIPT>
        //Task 6
    //Single-line comment:
    //LET A=19;
/*
    Multi-line comment:
*/
/*
if (typeof il === 'boolean') {
    console.log('myValue is a boolean');
} else {
    console.log('myValue is not a boolean');
}
    */
   //TASK 7:
   //Semicolon seperated
   let il=false;
   if (typeof il === 'boolean') {
    console.log('myValue is a boolean');
} else {
    console.log('myValue is not a boolean');
}
//semicolon not separated

if (typeof il === 'boolean') {
    console.log('myValue is a boolean')
} else {
```



```
    console.log('myValue is not a boolean')
}
// TASK8
let a=10,b=9;
for(a=0;a<10;a++){
    for(b=0;b<9;b++){
       console.log("hello");

        }
        console.log("everyone");
    }
for (let i = 0; i < 3; i++) {
    console.log( i);
    for (let j = 0; j < 2; j++) {
        console.log(j);
    }
}
// TASK 9
let b1=10,b3="string value",b9=true,c1=11;
console.log(b1+b3+b9+c1);
</SCRIPT>
```

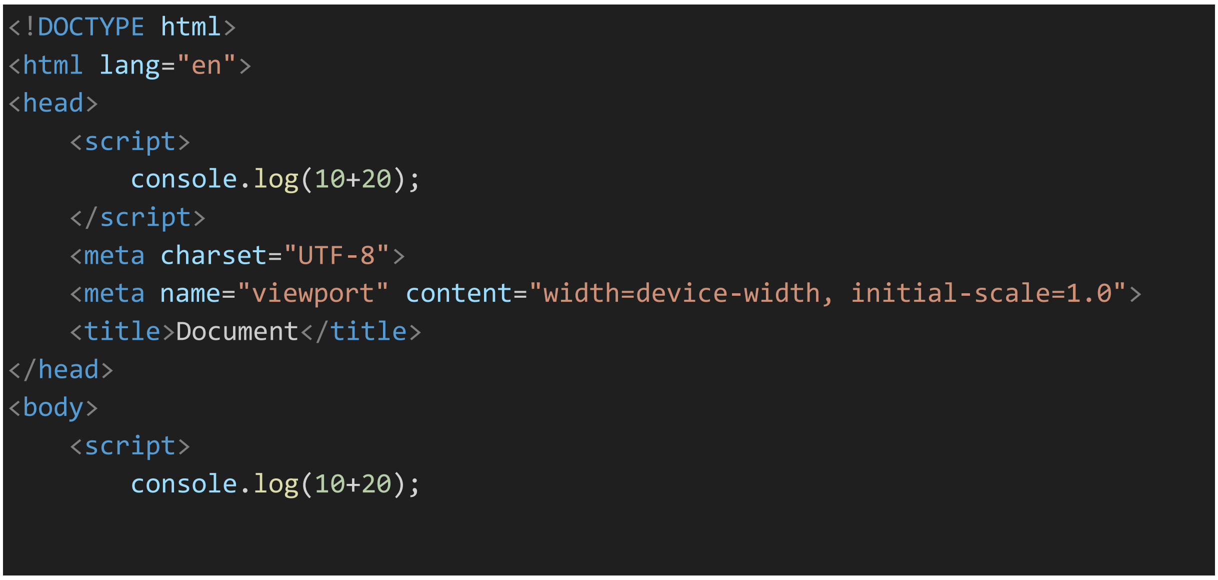


```
0
1
2
0
1
10string valuetrue11
```



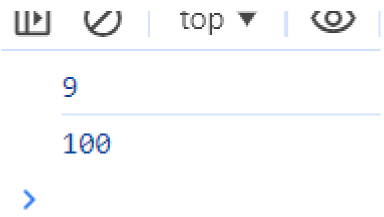
```
  myValue is a boolean
  myValue is a boolean
9 hello
  everyone
9 hello
  everyone
9 hello
  everyone
9 hello
  everyone
9 hello
```

Task:10;



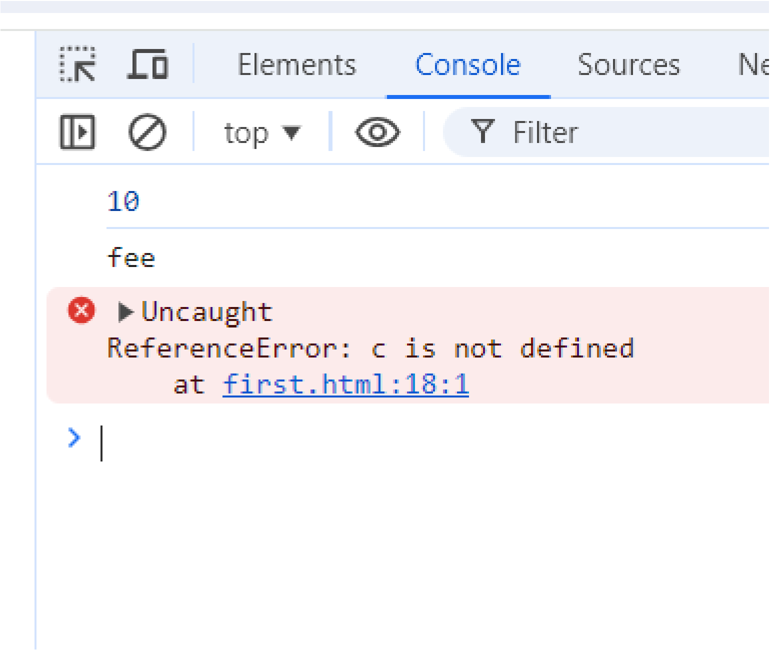
```
<!DOCTYPE html>
<html lang="en">
<head>
    <script>
        console.log(10+20);
    </script>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        console.log(10+20);
```

```
//task1
1x=10;
console.log(x);
//task12
"use script";
x1=100;
console.log(x1);
//task13
"use strict";
 let x=6;
 console.log(x);
 delete x();
```



```
9
100
>
```

```
//task 16:
let a=10;
console.log(a);
var b="fee";
console.log(b);
const let v=10;
console.let(v);
```



```
10
fee
▶Uncaught
ReferenceError: c is not defined
    at first.html:18:1
>
```

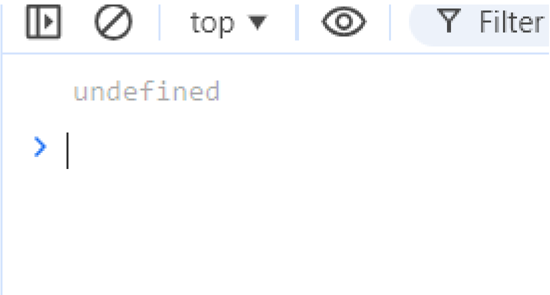
//task 17

const let v=10;

console.let(v);

let v=20;

console.let(v)



```
undefined
>
```

//task

18let

a=10;

console.log(typeof(a));

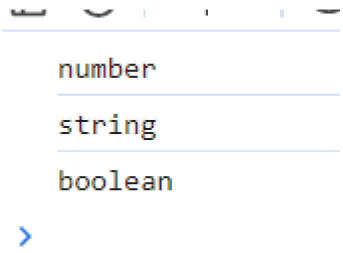
let b="string hy";

console.log(typeof(b));

let w=true;

//task 26

```
   let h="welcome";
   let num=1+h;
   console.log(num);
   console.log(typeof num);
   let g="welcome";
   let num0=parseInt(g);
   console.log(num0);
   console.log(typeof(num0));
console.log(typeof(w));
```



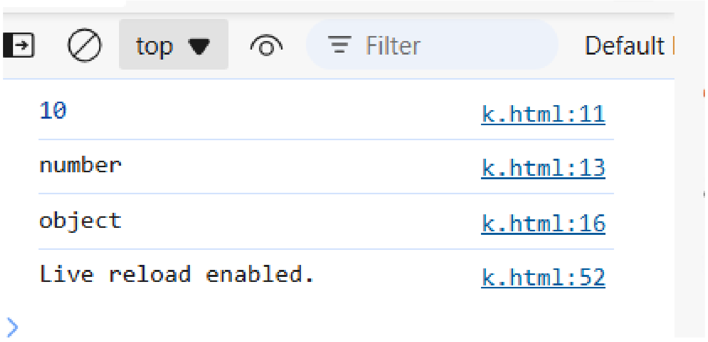
```
number
string
boolean
>
```

//task 20

```
let num=10;
num=20;
console.log(num);
```

//task 21,22.23.24.25

```
let a=10;
    console.log(a);
    let y=Symbol('a');
    console.log(typeof a);
    //task 24
    let o=null;       console.log(typeof o);
```



```
10                          k.html:11
number                      k.html:13
object                      k.html:16
Live reload enabled.        k.html:52
```

);

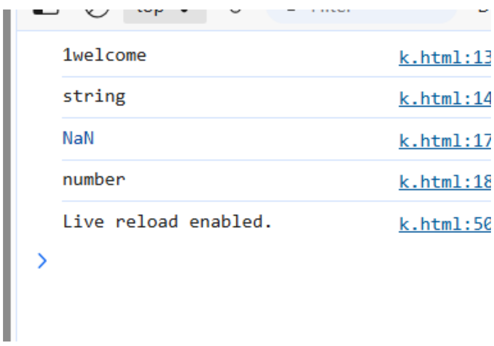
//task 25

//if you use var you cannot reassign or redeclare the variables

//if you use let you cannot redeclare

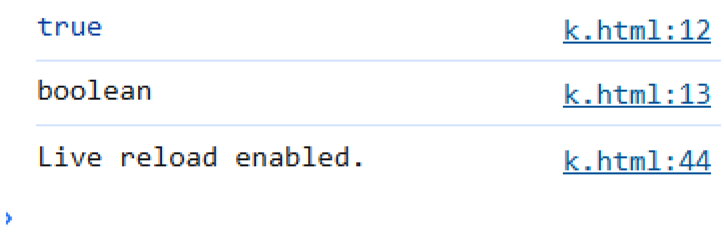
//task 26

```
let h="welcome";

let num=1+h;

console.log(num);

console.log(typeof num);

let g="welcome";

let num0=parseInt(g);

console.log(num0);

console.log(typeof(num0));
```



```
1welcome                k.html:13
string                  k.html:14
NaN                     k.html:17
number                  k.html:18
Live reload enabled.    k.html:50
```

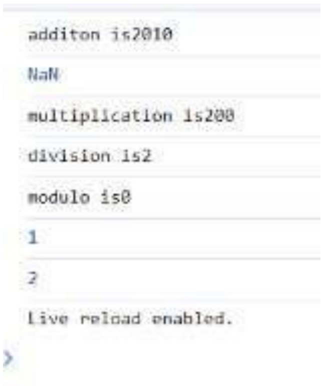
```
let bool=true;

let value=bool;

console.log(value);

console.log(typeof value);
```



```
true                                  k.html:12
boolean                               k.html:13
Live reload enabled.                  k.html:44
›
```

//TASK 28,29,30

```
let y=10;
    let x=20;
    console.log("additon is"+x+y);
    console.log("subtraction is"+x-y);
    console.log("multiplication is"+x*y);
    console.log("division is"+x/y);
    console.log("modulo is"+x%y);
    let i=1;
    console.log(i);
    i++;
    console.log(i);
```



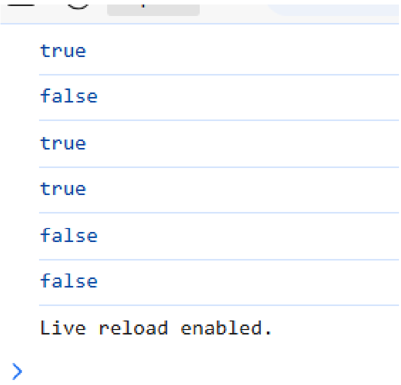
```
additon is2010
NaN
multiplication is200
division is2
modulo is0
1
2
Live reload enabled.
›
```

//TASK 31

```
 let a=8;
   let k=9;
   console.log(a<k);
   console.log(a>=k);
   console.log(a<k);
   console.log(a<=k);
```

//task 32

```
console.log(a==k);
console.log(a===k);
```



```
true
false
true
true
false
false
Live reload enabled.
>
```

//task 32

//task 33

```
let l="hello";
let j="organ";
console.log(l<j);
```

//task

34let

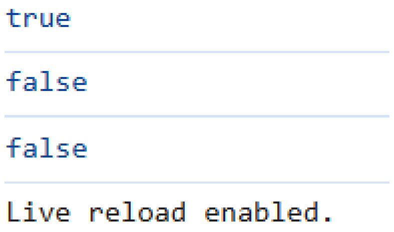
va=8;

```
   let ve=9;
   console.log(va!=ve);
   console.log(va!==ve);
```

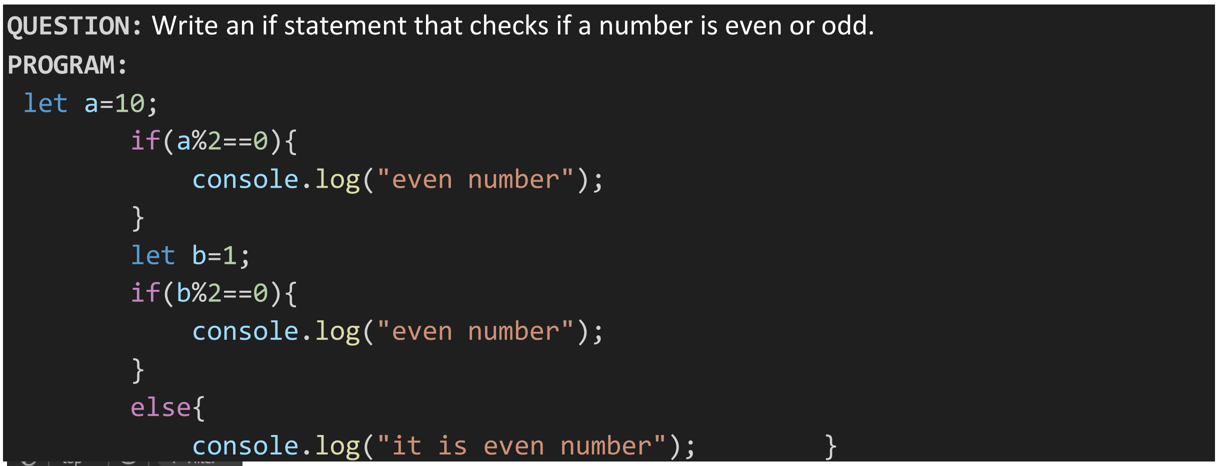
```
true                      [illegible]
false                     [illegible]
true                      [illegible]
true                      [illegible]
false                     [illegible]
false                     [illegible]
true                      [illegible]
true                      [illegible]
true                      [illegible]
Live reload enabled.      [illegible]
>
```

.//TASK 35

```
let va=8;
    let ve=9;
    console.log(va==ve);
    console.log(va===ve);
```



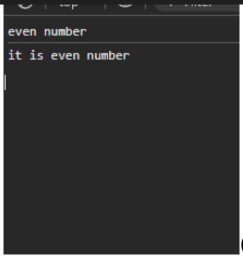
```
true
false
false
Live reload enabled.
```



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

PROGRAM:

```
 let a=10;
        if(a%2==0){
            console.log("even number");
        }
        let b=1;
        if(b%2==0){
            console.log("even number");
        }
        else{
            console.log("it is even number");        }
```



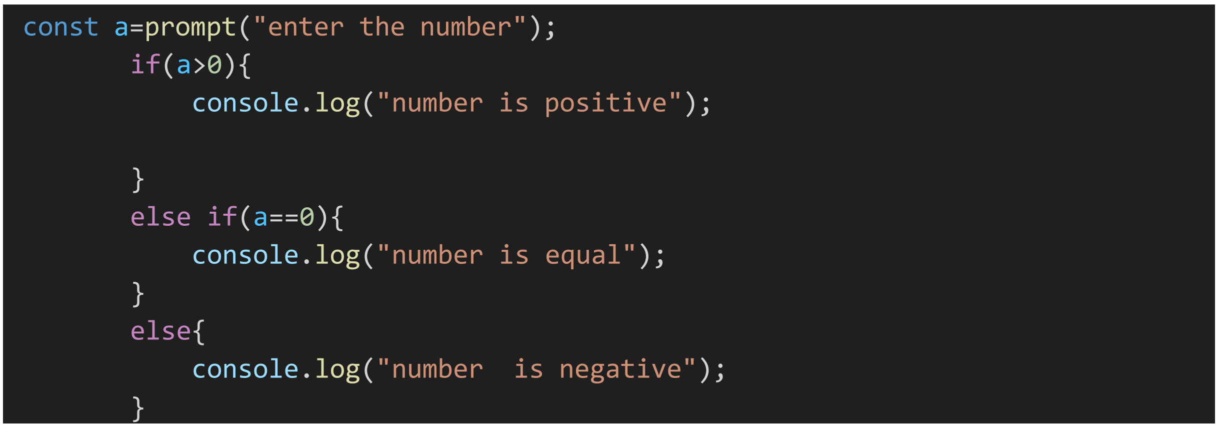
```
even number
it is even number
```

OUTPUT:

//task:37

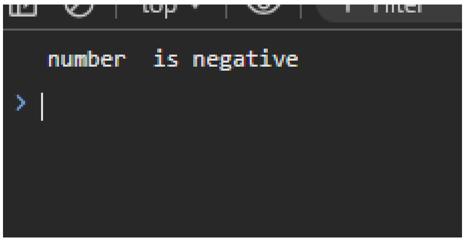
QUESTION: Use nested if statements to classify a number as negative, positive, or zero

PROGRAM:



```
 const a=prompt("enter the number");
        if(a>0){
            console.log("number is positive");

        }
        else if(a==0){
            console.log("number is equal");
        }
        else{
            console.log("number  is negative");
        }
```

OUTPUT:

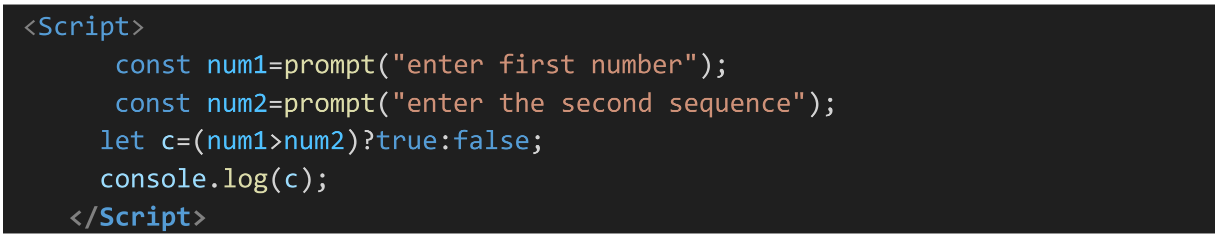


```
  number  is negative
> |
```

//task38:

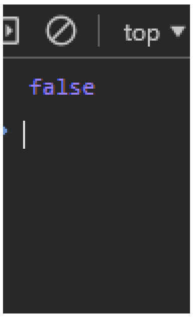
QUESTION: Use the conditional (ternary) operator ‘?’ to rewrite a simple if…else statement.

PROGRAM:



```
<Script>
      const num1=prompt("enter first number");
      const num2=prompt("enter the second sequence");
     let c=(num1>num2)?true:false;
     console.log(c);
   </Script>
```

OUTPUT:

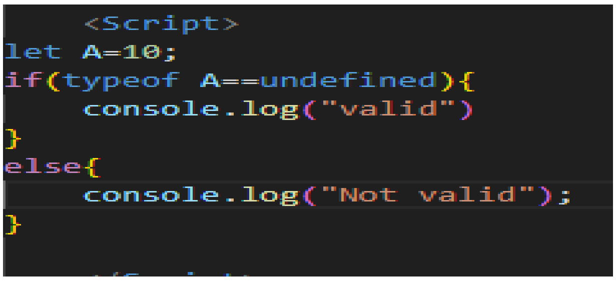


```
false
```

//task39

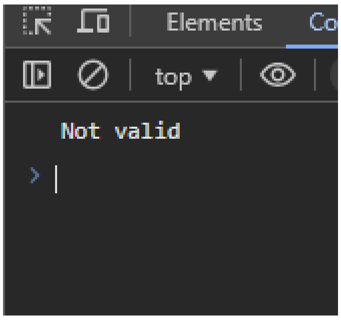
QUESTIONS: Check the validity of a variable using the ? operator.

PROGRAM:



```
    <Script>
let A=10;
if(typeof A==undefined){
    console.log("valid")
}
else{
    console.log("Not valid");
}
```

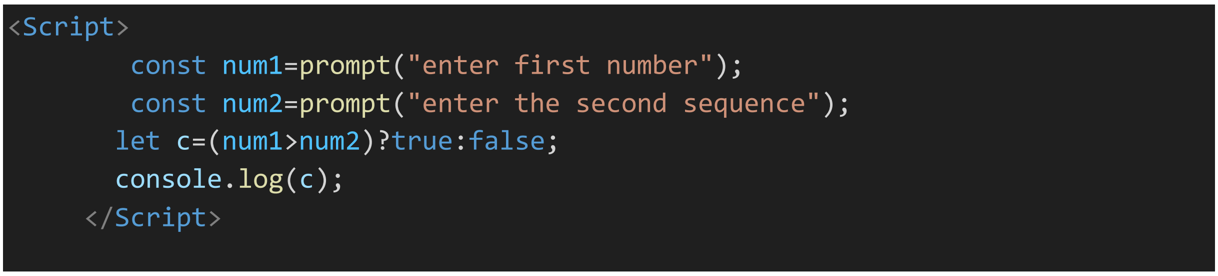
OUTPUT:



```
Not valid
```

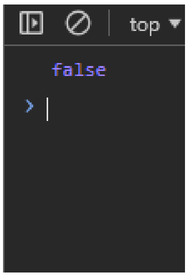
QUESTION : Use the conditional operator to assign a value to a variable based on a condition.

PROGRAM:



```
<Script>
        const num1=prompt("enter first number");
        const num2=prompt("enter the second sequence");
       let c=(num1>num2)?true:false;
       console.log(c);
     </Script>
```

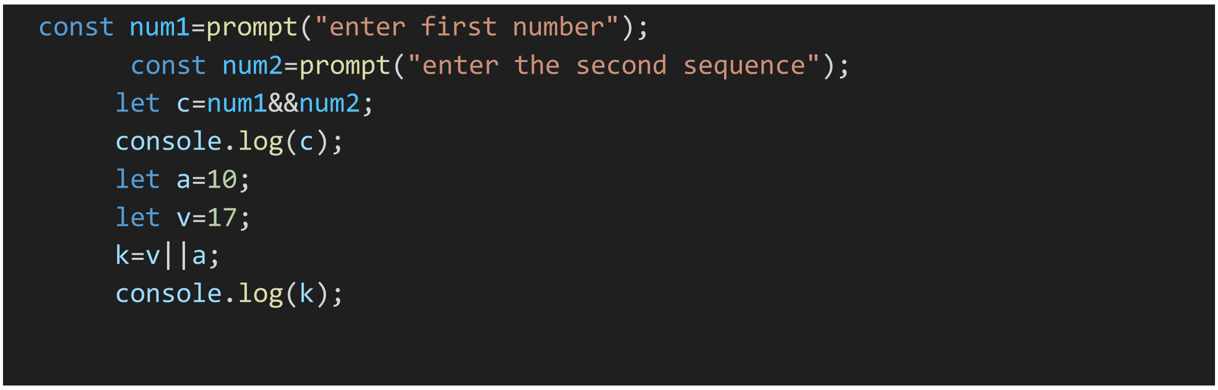
Output:



```
false
>
```

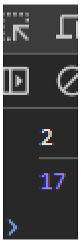
Question: Evaluate various combinations of logical operators (&&, ||, !).

PROGRAM:



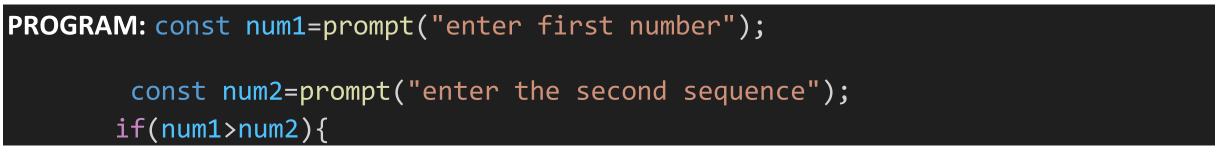
```
  const num1=prompt("enter first number");
        const num2=prompt("enter the second sequence");
       let c=num1&&num2;
       console.log(c);
       let a=10;
       let v=17;
       k=v||a;
       console.log(k);
```

OUTPUT:



```
2
17
>
```

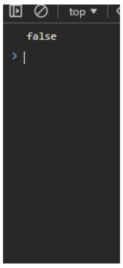
QUESTION: Use logical operators to write a condition that checks if a number is in a given range.



PROGRAM:

```
const num1=prompt("enter first number");

        const num2=prompt("enter the second sequence");
       if(num1>num2){
        console.log("true");
       }

        else{
            console.log("false");
        }
```

Output:



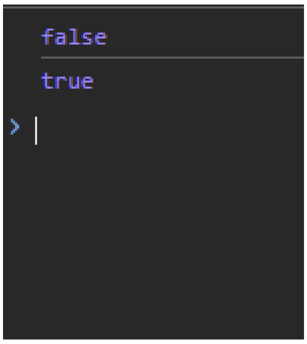
```
top ▼
false
›
```

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

PROGRAM:

```
let isTrue = true;
let isFalse = false;

let True = !isTrue;
let False = !isFalse;

console.log(negatedTrue);
console.log(negatedFalse);
```

OUTPUT:



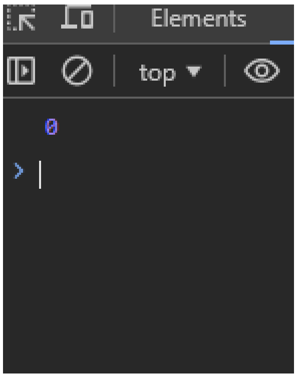
```
false
true
›
```

QUESTION: Evaluate the short-circuiting nature of logical operators.

PROGRAM:

```
const v = 0;
const result = v && 'Tree Value';
console.log(result);
```

OUTPUT:



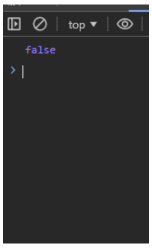
```
0
```

QUESTION; Compare two non-boolean values using logical operators and observe the result

PROGRAM:

```
const a = 9;
const b = -8;

console.log(!(a > 0 || b > 0));
     </Script>
```

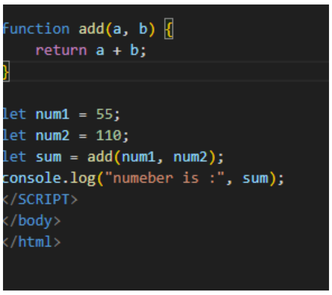
OUTPUT:



```
false
```

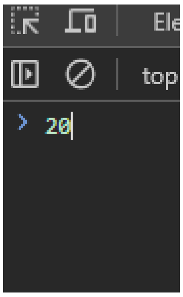
QUESTION: Write a function that takes two numbers as arguments and returns their sum

PROGRAM:



```
function add(a, b) {
    return a + b;
}

let num1 = 55;
let num2 = 110;
let sum = add(num1, num2);
console.log("numeber is :", sum);
</SCRIPT>
</body>
</html>
```

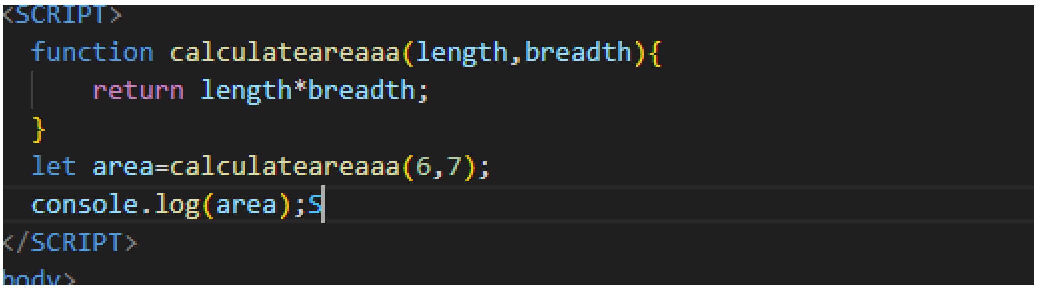
OUTPUT:



```
> 20
```

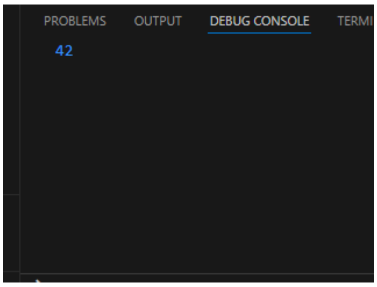
QUESTION: Create a function that calculates the area of a rectangle.

PROGRAM:



```
<SCRIPT>
  function calculateareaaaa(length,breadth){
      return length*breadth;
  }
  let area=calculateareaaaa(6,7);
  console.log(area);S
</SCRIPT>
body>
```

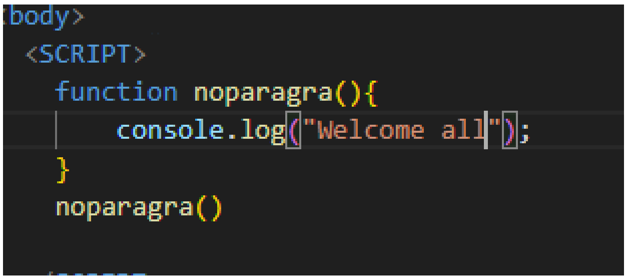
OUTPUT:



```
42
```

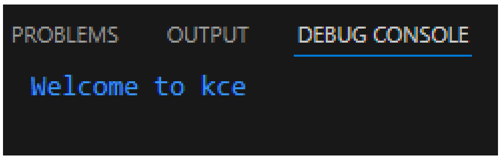
QUESTION: Declare a function without parameters and call it.

PROGRAM:



```
body>
  <SCRIPT>
    function noparagra(){
        console.log("Welcome all");
    }
    noparagra()
```

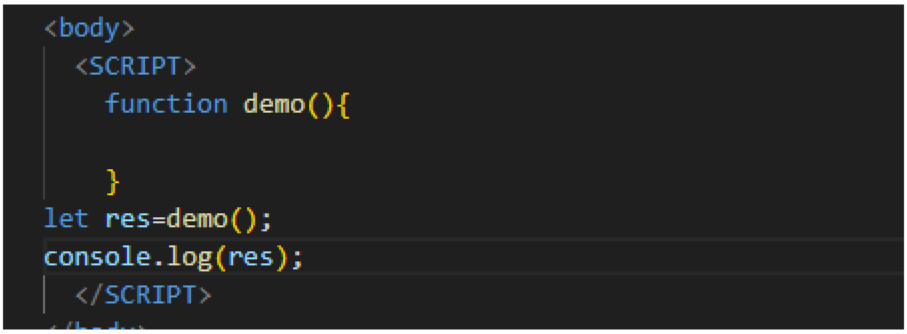
OUTPUT:



```
Welcome to kce
```

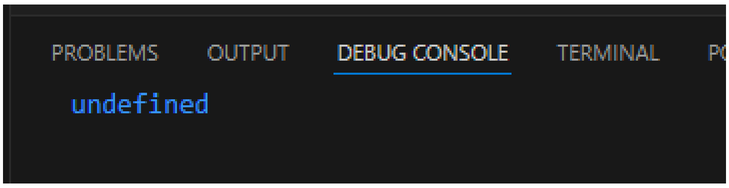
QUESTION: Write a function that returns nothing and observe the default return value.

PROGRAM:



```
<body>
  <SCRIPT>
    function demo(){

    }
let res=demo();
console.log(res);
  </SCRIPT>
</body>
```

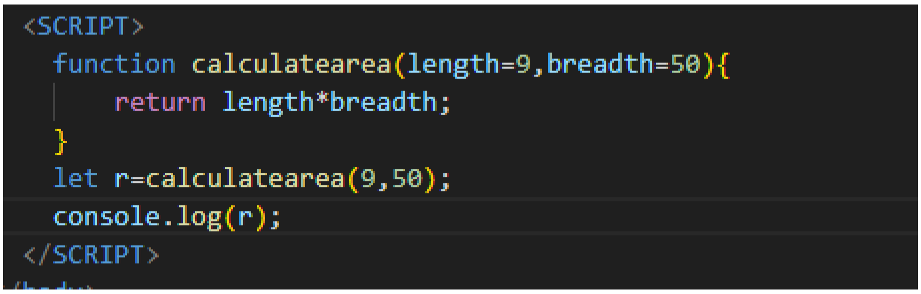
OUTPUT:



```
undefined
```

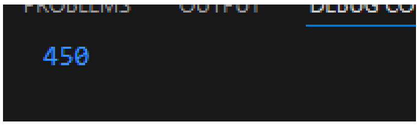
QUESTION: Declare a function with default parameters and call it with different arguments.

PROGRAM:



```
<SCRIPT>
  function calculatearea(length=9,breadth=50){
      return length*breadth;
  }
  let r=calculatearea(9,50);
  console.log(r);
</SCRIPT>
</body>
```

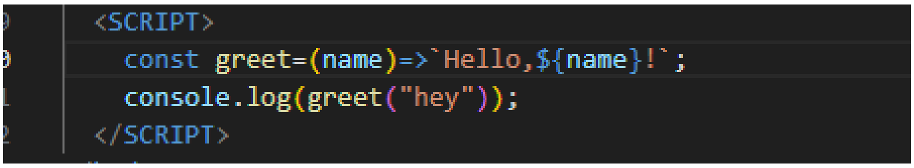
OUTPUT:



```
450
```

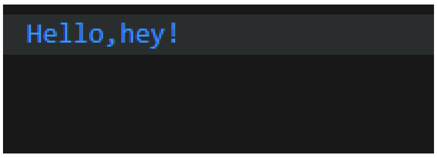
QUESTION: Declare a simple arrow function named greet that takes one parameter name and returns the string “Hello, name!”. Test your function with various names.

PROGRAM:



```
<SCRIPT>
  const greet=(name)=>`Hello,${name}!`;
  console.log(greet("hey"));
</SCRIPT>
```

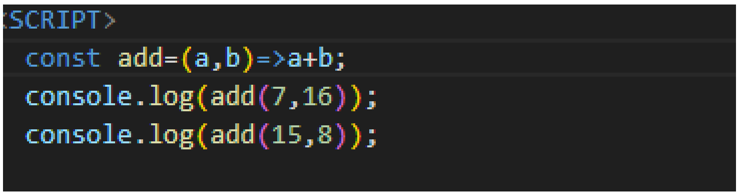
OUTPUT:



```
Hello,hey!
```

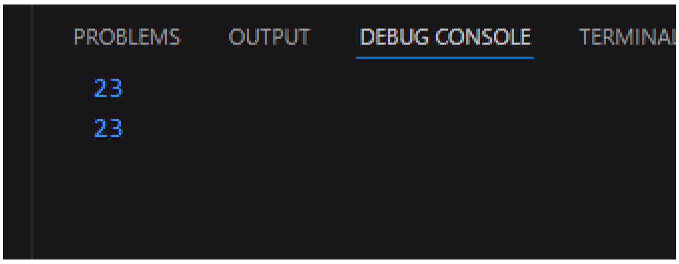
QUESTION:Write an arrow function named add that takes two parameters and returns their sum. Validate your function with several pairs of numbers.

PROGRAM:



```
<SCRIPT>
 const add=(a,b)=>a+b;
 console.log(add(7,16));
 console.log(add(15,8));
```

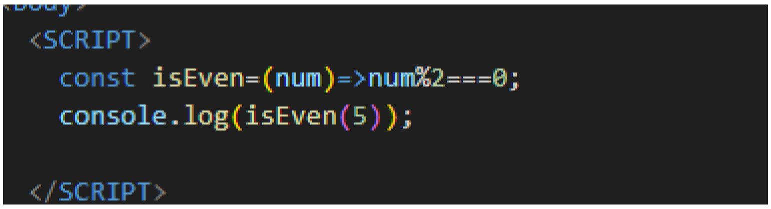
OUTPUT:



```
23
23
```

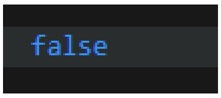
QUESTION : Declare an arrow function named isEven that checks if a number is even. If the number is even, it should return true; otherwise, false. Remember that if the arrow function body has a single statement, you can omit the curly braces.

PROGRAM:



```
 <SCRIPT>
   const isEven=(num)=>num%2===0;
   console.log(isEven(5));

 </SCRIPT>
```

OUTPUT:

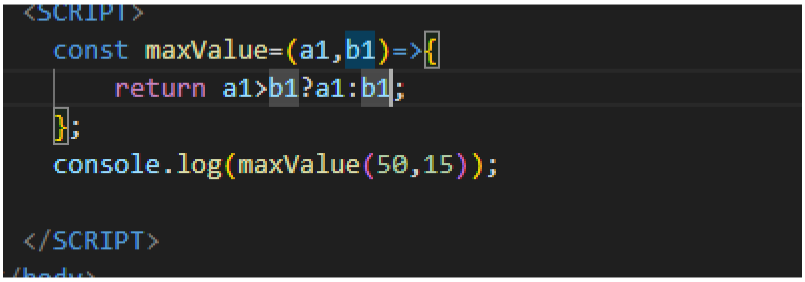


```
false
```

QUESTION: Implement an arrow function named maxValue that takes two numbers

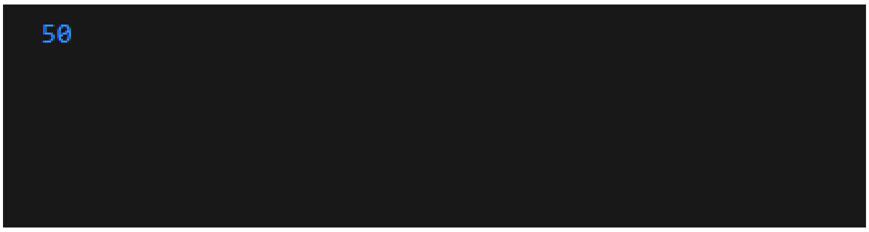
as parameters and returns the larger number. Here, you'll need to use curly braces for the function body and the return statement.

PROGRAM:



```
<SCRIPT>
  const maxValue=(a1,b1)=>{
      return a1>b1?a1:b1;
  };
  console.log(maxValue(50,15));

</SCRIPT>
</body>
```

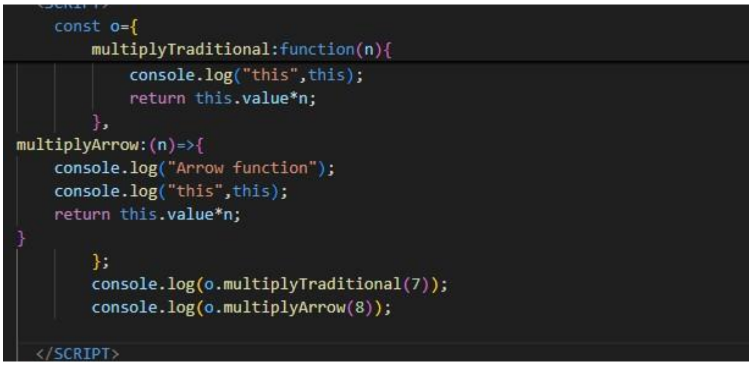
OUTPUT:



```
50
```

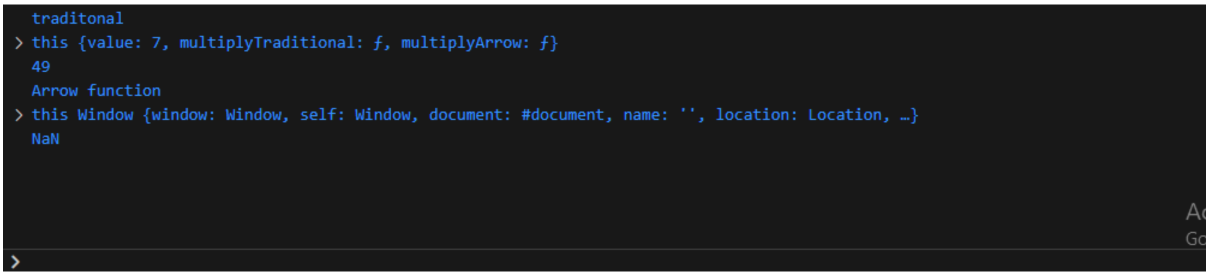
QUESTION: Examine the behavior of the this keyword inside an arrow function vs a traditional function. Create an object named myObject with a property value set to 10 and two methods: multiplyTraditional using a traditional function and multiplyArrow using an arrow function. Both methods should attempt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

PROGRAM:



```
    const o={
        multiplyTraditional:function(n){
            console.log("this",this);
            return this.value*n;
        },
multiplyArrow:(n)=>{
    console.log("Arrow function");
    console.log("this",this);
    return this.value*n;
}
        };
        console.log(o.multiplyTraditional(7));
        console.log(o.multiplyArrow(8));

  </SCRIPT>
```

OUTPUT:



```
  traditonal
> this {value: 7, multiplyTraditional: ƒ, multiplyArrow: ƒ}
  49
  Arrow function
> this Window {window: Window, self: Window, document: #document, name: '', location: Location, …}
  NaN
```