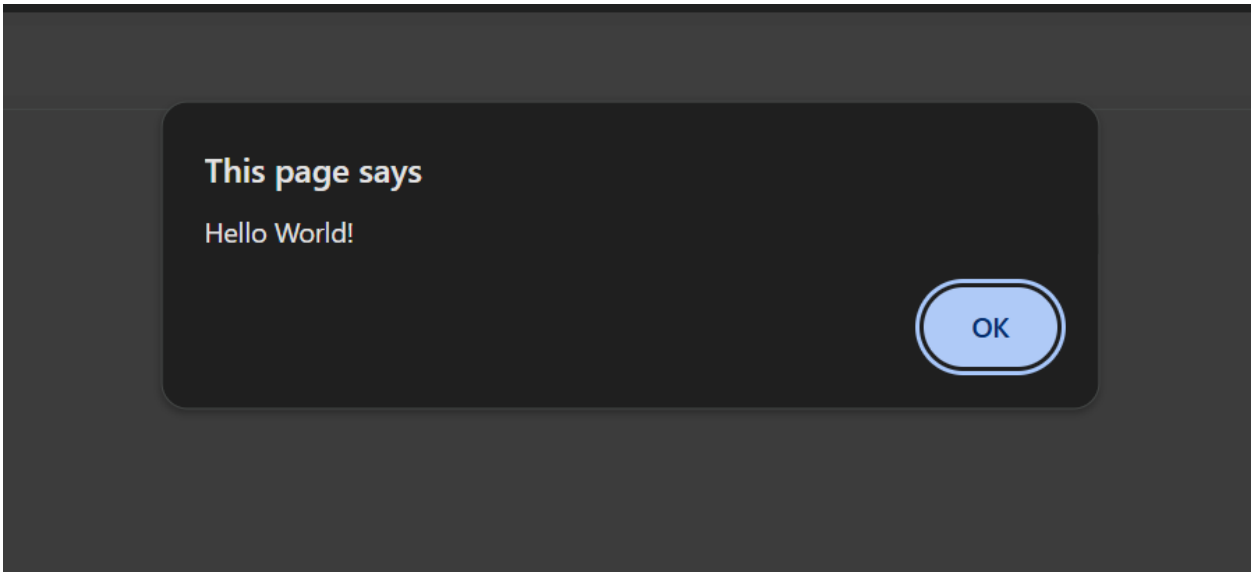
JAVA SCRIPT BASICS TASKS :

1. <!DOCTYPE html>

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


</body>
</html>
```

OUTPUT :



2.

index.html

C: > Users > hp > index.html > html > body > script

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
        let name = "Hazeeba";
        let age=19;
        let isboolean=true;
        console.log("String    : "+name);
        console.log("number    : "+age);
        console.log("isboolean : "+true);
    </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \

```
String    : Hazeeba
number    : 19
isboolean : true
```

3.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
</head>
<body>
    <script>
        let num1=10;
        let num2=15;
        let add=num1+num2;
        let sub=num1-num2;
        let mul=num1*num2;
        let div=num1/num2;
        console.log("Addition : "+add);
        console.log("Substraction : "+sub);
        console.log("Multiplication : "+mul);
        console.log("division : "+div);
    </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Addition : 25
Substraction : -5
Multiplication : 150
division : 0.6666666666666666
```

4.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
</head>
<body>
    <script>
        let firstname="Hazeeba";
        let lastname="Abuthahir";
        console.log("After concatenation : " + firstname + lastname);
    </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !e

```
After concatenation : HazeebaAbuthahir
```

7.

index.html

C: > Users > hp > index.html > html > body > script

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
        let firstname="Hazeeba";
        let age=19
        /*No,There is no diffrences with semicolon separated and non separated string,
        both are treared equally by the compiler.
        But using semicolon is good practice and preferable also.*/
    </script>
    </body>
</html>
```

5.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
</head>
<body>
    <script>
        let firstname="Hazeeba";
        let age=19;
        let lastname;
        let rollno=717822n;
        let isboolean=true;
        console.log(firstname + " : " + typeof(firstname));
        console.log(age + " : " + typeof(age));
        console.log(lastname + " : " + typeof(lastname));
        console.log(rollno + " : " + typeof(rollno));
        console.log(isboolean + " : " + typeof(isoolean));
    </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e

```
Hazeeba : string
19 : number
undefined : undefined
717822 : bigint
true : undefined
```

6.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
     //let firstname="Hazeeba";
     let age=19
     // - single line command use to command the single line of code block
     /*multi line command use to command the
     multiple line of code block.*/
    </script>
    </body>
</html>
```

8.

```html
    <title>JS Task</title>
</head>
<body>
    <script>
    for(var i=0;i<5;i++){
        for(var j=i+1;j<5;j++){
            console.log(i+j);
        }
        console.log(" ");
    }
    </script>
    </body>
</html>
```

9.

```
index.html ×
C: > Users > hp > index.html > html > body > script
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>JS Task</title>
7  </head>
8  <body>
9      <script>
10     let a=9,b=1,c=2;
11     console.log("a : "+a);
12     console.log("b : "+b);
13     console.log("c : "+c);
14     </script>
15     </body>
16 </html>

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   Filter (e.g. text, !exclude,

a : 9
b : 1
c : 2
```

10.

There is no diffrence in the output.I t works same in both cases.

11.

```
8  <body>
9      <script>
10     //without using strict
11      a=1;
12     let b=1;
13     console.log("a :"+a);
14     console.log("b :"+b);
15     </script>
16     </body>
17 </html>

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMIN

a :1
b :1
```

.

12.

```
index.html ×
C: > Users > hp > index.html > html > body > script
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <title>JS Task</title>
7   </head>
8   <body>
9       <script>
10      //with using strict
11      "use strict"
12      a=1;
13      let b=1;
14      console.log("a :"+a);
15      console.log("b :"+b);
16      </script>
17      </body>
18  </html>
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   Filter (e.g. text, !exclude, \esc
> Uncaught ReferenceError ReferenceError: a is not defined
    at <anonymous> (c:\Users\hp\index.html:12:7)
```

14.

```
index.html ×
C: > Users > hp > index.html > html > body
2   <html lang="en">
3   <head>
7   </head>
8   <body>
9       <script>
10      //without using strict
11      a=1;
12      let b=1;
13      console.log("a :"+a);
14      console.log("b :"+b);
15      "use strict"
16      c=1;
17      console.log("c :"+c);
18
19      </script>
20      </body>
21  </html>
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL
a :1
b :1
c :1
```

13.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
</head>
<body>
    <script>
    //with using strict
    "use strict"
    a=1;
    let b=1;
    add();
    function add(){
        console.log(5+9);
    }
    delete(b);
    delete(add);
    console.log("a :"+a);
    console.log("b :"+b);
    </script>
    </body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Filter (e.g. text, !exclude, \escape)

```
Uncaught SyntaxError SyntaxError: Delete of an unqualified identifier in strict mode.
    at (program) (c:\Users\hp\index.html:18:13)
```

15.

index.html 5

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
</head>
<body>
    <script>
    //with using strict
    "use strict"
    let if =1;
    let b=1;
    console.log("if :"+if);
    console.log("b :"+b);
    </script>
    </body>
</html>
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclu

```
Uncaught SyntaxError SyntaxError: Unexpected strict mode reserved word
    at (program) (c:\Users\hp\index.html:12:5)
```

16.

index.html

C: > Users > hp > index.html > html > body > script

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
        let a = 10;
        var b = 20;
        const c = 30;
     /*Explanation:
            let: Global scope, we can access the variable declare with let in anywhere in the code.
            Var: Local scope , we access the variable declare with var in within the block which it is defined.
            Const : constant scope*/
        </script>
</body>
</html>
```

17.

index.html

C: > Users > hp > index.html > html > body > script

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
        const c = 30;
        c=50;
        console.log("c : "+c);
        </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \esca

```
Uncaught TypeError TypeError: Assignment to constant variable.
    at <anonymous> (c:\Users\hp\index.html:11:10)
```

18.

index.html

C: > Users > hp > index.html > html > body > script

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \escape)

```
c : undefined
```

19.

```html
</head>
<body>
    <script>
        let name="hazeeba";
        let age=19;
        let isboolean=true;
        console.log(name + " : " + typeof(name));
        console.log(age + " : " + typeof(age));
        console.log(isboolean + " : " + typeof(isboolean));

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
hazeeba : string
19 : number
true : boolean
```

20.

index.html

C: > Users > hp > index.html > html > body > script

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JS Task</title>
</head>
<body>
    <script>
        let name="hazeeba";
        let age=19;
        let newname=name;
        console.log(newname + " : " + typeof(newname));
         </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \e

```
hazeeba : string
```

21 & 22.

```
<body>
  <script>
    let name="hazeeba";
    let age=19;
    let isboolean=true;
    let lastname;
    let offer=null;
    let details={
     add(){
        a=1;
        b=2;
     }
    }
    console.log(name + " : " + typeof(name));
    console.log(age + " : " + typeof(age));
    console.log(isboolean + " : " + typeof(isboolean));
    console.log(lastname + " : " + typeof(lastname));
    console.log(offer + " : " + typeof(offer));
    console.log(details + " : " + typeof(details));

     </script>
</body>
```

OUTPUT :

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORT

hazeeba : string
19 : number
true : boolean
undefined : undefined
null : object
[object Object] : object
```

23.

```
8   <body>
9       <script>
10          var a = Symbol("hello")
11          console.log(typeof(a));
12      </script>
13  </body>
14  </html>
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

symbol
```

24.

```
</head>
<body>
    <script>
        var a = null;
        console.log(typeof(a));
    </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PO

```
object
```

25.

```
<body>
    <script>
        var a = 10;
        if(a==10)
      {
        let b=20;
      }
        console.log(b);
      </script>

</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

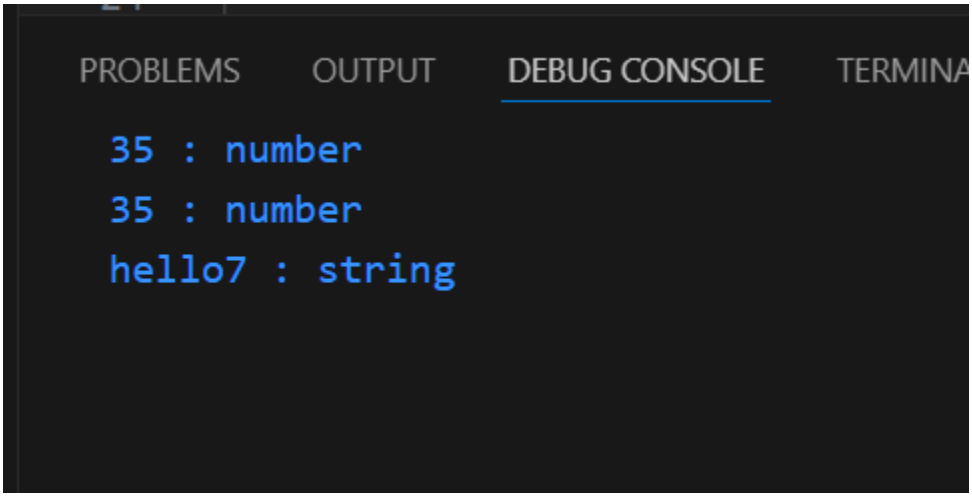
```
> Uncaught ReferenceError ReferenceError: b is not defined
    at <anonymous> (c:\Users\hp\index.html:15:21)
```

26. <script>

```
//implicit conversion

let a="7";

let b="7"*5;

console.log(b + " : " +typeof(b));

//Explicit Conversion

let m="7";

let n=5*(parseInt(m));

console.log(n +" : "+typeof(n));

let h="hello";

let g=7;

let j=h+(String(g));

console.log(j +" : "+typeof(j));
```

</script>

OUTPUT :



27.

```
8   <body>
9       <script>
10          var c = true
11         var a =  c.toString();
12          console.log(typeof(a));
13          var str = "hi"
14          var st = !!str;
15          console.log(typeof(st));
16         </script>
17
18  </body>
19  </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
string
boolean
```

28.

```
9       <script>
10          var a = 15;
11          var b = 3;
12          console.log("Addition : "+ (a+b));
13          console.log("Substraction :"+(a-b));
14          console.log("Multiplication : "+a*b);
15          console.log("Division : "+a/b);
16         </script>
17
18
19  </body>
20  </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Addition : 18
Substraction :12
Multiplication : 45
Division : 5
```

29.

```
</head>
<body>
    <script>
        var a = 15;
        var b = 3;
        console.log(++a);
        console.log(--b);
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERM

```
16
2
```

30.

```
<body>
    <script>
        var a = 15;
        var b = 3;
        var c=a+b*a-b;
        let result = 5 + 3 > 2 && 4 < 10 || false;
        console.log(c);
        console.log(result);
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
57
true
```

31.

```
<script>
    var a = 15;
    var b = 3;
    if(a<=b){
    console.log(a);
    }
    else{
        console.log(b);
    }if(a>=b){
    console.log(a);
    }
    else{
        console.log(b);
    }
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PO

```
3
15
```

32.

```
<body>

  <script>

    console.log("Equality Operator : It only checks the value");

    console.log(5 == '5');      // true (string '5' is converted to number 5)

    console.log(0 == false);     // true (false is converted to 0)

    console.log(null == undefined); // true (special case where they are considered equal)

    console.log('' == false);    // true (empty string is converted to 0, false is 0)

    console.log([1] == '1');     // true (array is converted to string '1')

    console.log(42 == true);     // false (42 is not converted to 1)

    console.log("Strict Equality Operator : It checks the value and data type ");

    console.log(5 === '5');      // false (number !== string)

    console.log(0 === false);    // false (number !== boolean)
```

```
    console.log(null === undefined); // false (different types)

    console.log('' === false);   // false (string !== boolean)

    console.log(42 === 42);      // true (both are the same number)

  </script>

</body>
```

OUTPUT :

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   Filt

Equality Operator : It only checks the value
true
true
true
true
true
false
Strict Equality Operator : It checks the value and data type
false
false
false
false
true
```

33.

```
8   <body>
9       <script>
10          //RELATIONAL OPERATORES COMPARES THE STRINGS BASED ON THIER UNICODES
11         const str1 = "Apple";
12          const str2 = "apple";
13          console.log(str1 < str2);
14          console.log(str1 > str2);
15          console.log(str1 === str2);
16
17      </script>
18  </body>
19  </html>
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   Filter (e.g. text, !exclude, \esca

true
false
false
```

34.

```
<body>
    <script>
        const str1 = "Apple";
        const str2 = "Apple";
        const num=7;
        console.log(str1 !== str2);
        console.log(str1 != num);

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
false
true
```

35.

```
    <script>
        const str1 = null;
        let str2;
        console.log(str1 === str2); //it checks the datatype and its value
        console.log(str1 == str2); //it only checks the value

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \es

```
false
true
```

36.

```
    <script>
        let a=5;
        if(a%2==0){
            console.log("EVEN");
        }
        else{
            console.log("ODD");
        }
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORT

```
ODD
```

37.

```
<body>
    <script>
        let a=5;
        if(a>=0){
            if(a==0)
            console.log("ZERO");
        else
            console.log("POSITIVE")
        }
        else{
            console.log("NEGATIVE");
        }
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
POSITIVE
```

38.

```
    <script>
        let a=5;
        let c=(a>0)?"Positive":"Negative";
        console.log(c);
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Positive
```

39.

```
    <script>
    let myVar = "Hello, world!";
    let result = (myVar != null) ? "Valid" : "Invalid";
    console.log(result);
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Valid
```

40.

```
<body>
    <script>
    let myvar = -15;
    let result = (myvar >0) ? myvar : -myvar;
    console.log(result);
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
15
```

41.

```
<script>
let and=true && false;
let or=true|| false;
let not=!true;
console.log("Logical AND : "+and);
console.log("Logical OR : "+or);
console.log("Logical NOT : "+not);
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Logical AND : false
Logical OR : true
Logical NOT : false
```

42.

```
<script>
let a=5;
if(a>=10 && a<100 || a>=0 && a<10){
console.log("true");
}
if(!a){
console.log("false");
}
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
true
```

43.

```
<script>
let bool=false;
console.log(!bool);
</script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINA

```
true
```

44.

```
<body>
   <script>
    /*Logical AND (&&)
        Evaluates the left-hand operand first.
        If the left-hand operand is falsy (false, 0, null, undefined, NaN, or ""),
        the entire expression evaluates to the left-hand operand, and the right-hand operand is not evaluated.
   */
   let a = false;
   let result = a && console.log("This won't be logged");
   console.log(result);
   /*  Logical OR (||)
       Evaluates the left-hand operand first.
       If the left-hand operand is truthy, the entire expression evaluates to the left-hand operand, and the right-hand operand is not evaluated.
   */
   let b = true;
   let result1 = b || console.log("This won't be logged");
   console.log(result1);
   /* Logical NOT (!)
   Not short-circuiting but inverts the truthiness of the operand.
   */
   let c = false;
    console.log(!c);
</script>
```

</body>

OUTPUT :

```
PROBLEMS    OUTPUT

false
true
true
```

If we use console.log in AND Operation :

```
8   <body>
9       <script>
10      /*The result is undefined because the console.log() function in JavaScript does not return any value.
11       It returns undefined by default.*/
12       let a = true;
13  let result = a && console.log("This won't be logged");
14  console.log(result);
15
16  </script>
17  </body>
18  </html>

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   Filter (e.g. text, !exclude, \escape)

This won't be logged
undefined
```

45.

```
9       <script>
10      /* The && operator returns:
11         The first falsy value if one exists, or
12         The last operand if all are truthy.*/
13      let a = true;
14      let b=" ";
15      console.log(a && b);//output b
16      /* The || operator returns:
17         The first truthy value, or
18         The last operand if all are falsy.
19      */
20      let c=true;
21      let d=false;
22      console.log(c || d)//output d
23  </script>
24  </body>
25  </html>

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

true
```

46.

```
8  <body>
9      <script>
10       function add(a,b){
11         console.log("sum : "+(a+b));
12       }
13       add(5,4);
14   </script>
15   </body>
16   </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sum : 9
```

If we missed the bracket in the place of console.log

```
9      <script>
10       function add(a,b){
11         console.log("sum : "+a+b);
12       }
13       add(5,4);
14   </script>
15   </body>
16   </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sum : 54
```

47.

```
9      <script>
10       function AreaOfRectangle(length,width){
11         console.log("AreaOfRectangular : "+(length * width));
12       }
13       AreaOfRectangle(5,4);
14   </script>
15   </body>
16   </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e

```
AreaOfRectangular : 20
```

48.

```
8   <body>
9       <script>
10          function great(){
11              console.log("Great Work!");
12          }
13          great();
14      </script>
15  </body>
16  </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Great Work!
```

49.

```
9       <script>
10          function great(){
11              console.log("Greate Work!!");
12          }
13          let ans=great();
14          console.log(ans);//Output : Undefined
15      </script>
16  </body>
17  </html>
```

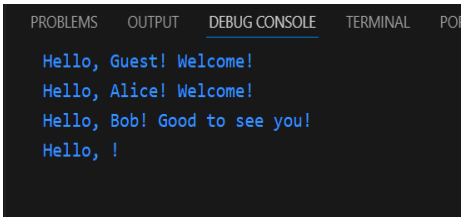
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Greate Work!!
undefined
```

50.

```
<body>
   <script>
    function greet(name = "Guest",message="Welcome!") {
   console.log(`Hello, ${name}! ${message}`);
   }

   // Call the function without any arguments
   greet(); // Default parameters are used

   // Call the function with one argument
   greet("Alice"); // Default message is used

   // Call the function with both arguments
   greet("Bob", "Good to see you!");

   // Call the function with empty strings as arguments
   greet("", ""); // Overrides defaults with empty strings

   </script>
</body>
```

OUTPUT :



51.

```
    <script>
      const greet=(name) => {"hello , ${name}!"};
      console.log("Hazeeba");
      console.log("KCE");
      console.log(" ");
      console.log("null");
    </script>
  </body>
  </html>
```

ROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Hazeeba
KCE

null
```

52.

```
  <body>
    <script>
      const greet=(a,b) => `sum : ${a+b}`;
      console.log(greet(1,2));
      console.log(greet(10,4));
      console.log(greet(30,40));
      console.log(greet(0,5));
    </script>
  </body>
  </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sum : 3
sum : 14
sum : 70
sum : 5
```

53.

```
<body>
    <script>
        const isEven=(num) => num%2===0;
        console.log(isEven(6));
        console.log(isEven(10));
        console.log(isEven(5));
        console.log(isEven(3));
    </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
true
true
false
false
```

54.

```
    <script>
        const maxValue=(num1,num2) => {
            if(num1>num2){
                return num1;
            }
            else{
                return num2;
            }
        };
        console.log(maxValue(6,16));
        console.log(maxValue(6,4));
        console.log(maxValue(6,300));
        console.log(maxValue(6,30));

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
16
6
300
30
```

55.

Traditional Function (multiplyTraditional):

- In a traditional function, when it's called as a method of an object (myObject.multiplyTraditional(6)), the this keyword inside the function refers to the object itself (myObject).
- Therefore, this.value will be 10 (the property of myObject), and the multiplication will work as expected.

Arrow Function (multiplyArrow):

- Arrow functions **don't have their own this**. Instead, they **inherit this** from the surrounding context where they are defined (in this case, the global context or function scope).
- Since this inside the arrow function doesn't refer to myObject, it leads to an unexpected result. In the browser, for example, this might refer to the window object, and this.value would be undefined in the global context, so the multiplication will likely result in NaN or undefined.

```
<body>
    <script>
     const myObject={
        value : 10,
        multiplyTraditional : function(num){
            console.log("Traditional Function This : "+this);
            return this.value*num;
        },
        multiplyArrow : (num)=>{
            console.log("Arrow Function This :"+this);
            return this.value*num;

        }
     };
     console.log(myObject.multiplyTraditional(6));
     console.log(myObject.multiplyArrow(60));
   </script>
</body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e

```
Traditional Function This : [object Object]
60
Arrow Function This :[object Window]
NaN
```