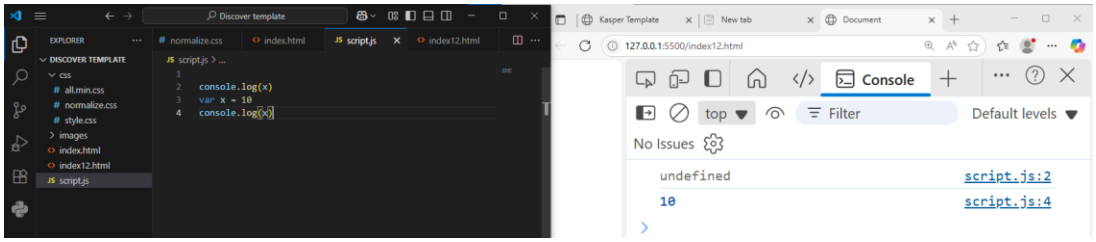
1. Explain how var works in JavaScript. What is variable hoisting? Give a code example.

➔To declare a variable we used – var – keyword. The js engine can detect the type of variable. There two types of variables

- Primitive type
- Reference type

var x = 10;

➔ Hoisting : it is about moving variable declaration to the top of their scope



2. What is the scope of a variable declared with var inside a function? What about inside a block (e.g., an if statement)?

- When we declare a variable in side function, it is called function scope which we can not call this variable outside.

- When we declare inside block, it is in global scope which we can call it outside block.

3. List all JavaScript primitive types in ES5. Give an example of each.
   - Number => 1.001 , 0.03 , 1
   - String => “string”
   - Boolean => true , false
   - Undefined
   - Null

```
> typeof 10
< 'number'
> typeof 'abc'
< 'string'
> typeof true
< 'boolean'
> typeof x
< 'undefined'
```

4. What is the difference between a primitive type and an object type? Give an example where this difference is important.
   - The primitive data types is immutable data types which stored in the heap and store the actual value .
   - The object type stored in heap and stores a reference for actual data.

```
> x = 10
< 10
> y = 10
< 10
> x == y
< true
> x = new Number(x)
< ▶ Number {10}
> y = new Number
< ▶ Number {0}
> y = new Number(y)
< ▶ Number {0}
> x == y
< false
>
```

5. Create a number, string, and boolean using both literal and constructor syntax. Show the difference in their types using typeof.

```
> str = 'abc'
< 'abc'
> str1 = new String('abc')
< ▶ String {'abc'}
> typeof str
< 'string'
> typeof str1
< 'object'
> num1 = 30
< 30
> num2 = new Number(30)
< ▶ Number {30}
> typeof num1
< 'number'
> typeof num2
< 'object'
> bool1 = true
< true
> bool2 = false
< false
> bool2 = new Boolea
✕ ▶ Uncaught ReferenceError: Boolea is not defined                VM874:1
    at <anonymous>:1:1

[NEW] Explain Console errors by using Copilot in Edge: click
to explain an error. Learn more                    Don't show again

> bool2 = new Boolean(bool2)
< ▶ Boolean {false}
```

5. Why is it generally recommended to use literals instead of constructors for primitive types?
   - As literal is faster than constructor. Engine does not have to call constructor.

- When we use constructor, we make a wrapper for value not a primitive type

7. Given the following code, what will be the output? Explain why.

var x = 123.4567;

console.log(x.toFixed(2)); -> return a string with two numbers rounded after the point.

console.log(x.toPrecision(4)); -> return a string of significant digit of number

```
> var x  = 123.4567
< undefined
> x.toFixed(2)
< '123.46'
> x.toPrecision(4)
< '123.5'
```

8. What is NaN? How can you check if a value is NaN? Give an example.

- nan is a value for a number which means that is not a number.

- check if number is nan using (isNaN)

```
> notNum = new Number('a')
<· ▶ Number {NaN}
> isNaN(notNum)
<· true
> isNan
```

9. What is the difference between parseInt, parseFloat, and Number? Give an example for each.

- parseInt() used to convert string into integer number

- parseFloat() used to convert string into decimal number

```
> x = 10.242354
<· 10.242354
> parseInt(x)
<· 10
> parseFloat(x)
<· 10.242354
>
```

10. What is the difference between implicit and explicit type casting? Give an example of each.

- implicit means that the type is converted automatic
- explicit means that you tells the system to convert it

```
> x = 'a' + 1
<· 'a1'
> x = 'a' + new String(1)
<· 'a1'
>
```

11. What will be the result and type of the following expressions? Explain your answer.

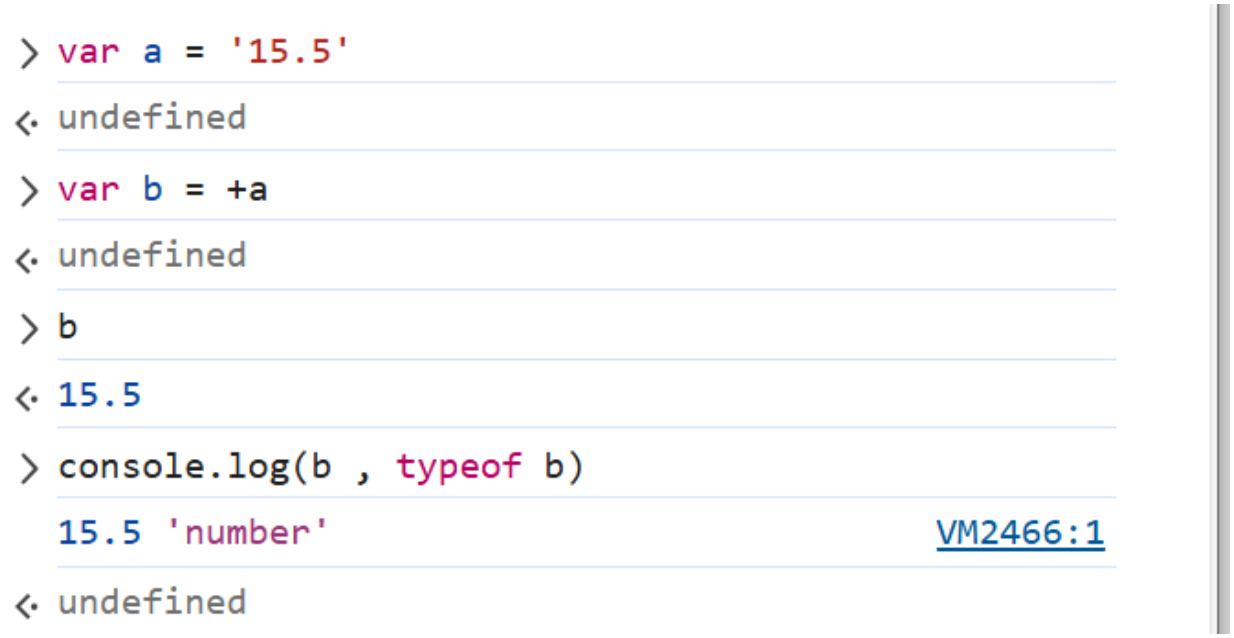
- true + 5 => 6
- "10" – 2 => 8
- 12 - "1a" => NaN
- 5 / 0 => infinity
- 5 + undefined => NaN

12. What will be logged to the console in the following code? Explain each step.

var a = "15.5";

var b = +a;

console.log(b, typeof b);



13. What will be the output of:

var result = 20 > true < 5 == 1;

console.log(result); // true

Explain why.

20 > true => true

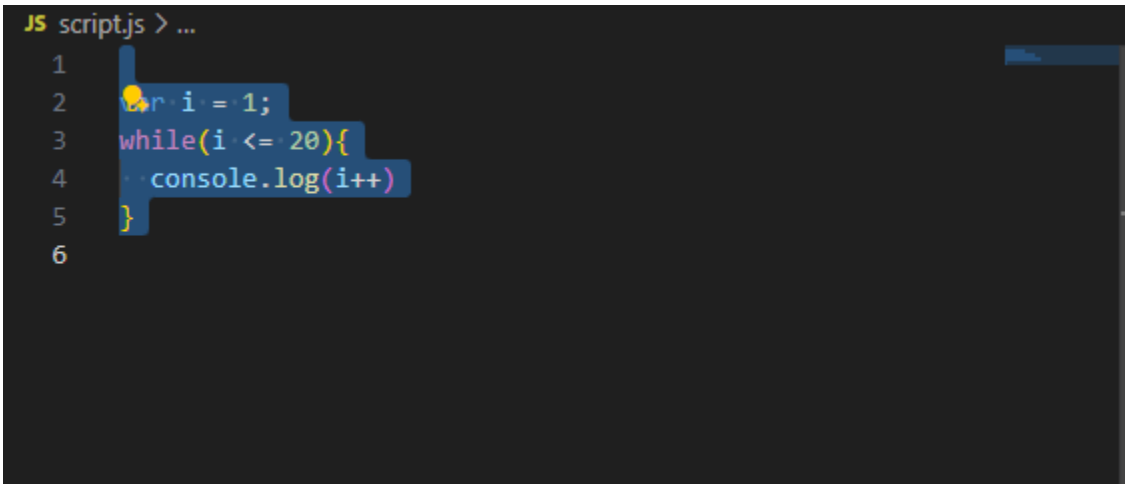
True < 5 => true

True == 1 => true

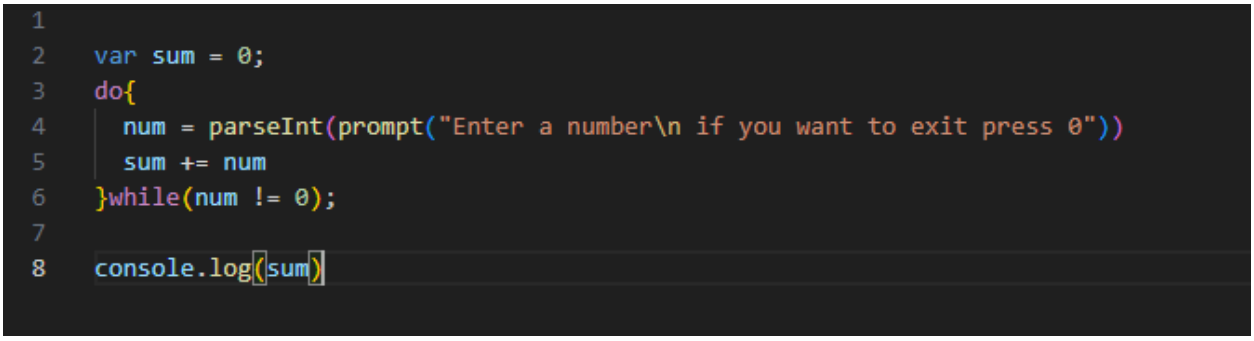
14. Write a function that takes a string and returns true if it can be converted to a valid number, and false otherwise.

```
> myFunc = function(str){
      return !isNaN(Number(str)) &&
  isFinite(Number(str))
  }
< f (str){
      return !isNaN(Number(str)) &&
  isFinite(Number(str))
  }
> myFunc('134')
< true
> myFunc('a')
< false
>
```

15. Write a program that prints all numbers from 1 to 20 using a while loop.



16. Write a program that asks the user to enter numbers until they enter 0, using a do...while loop. After the loop ends, print the sum of all entered numbers (excluding 0).



```
1
2   var sum = 0;
3   do{
4     num = parseInt(prompt("Enter a number\n if you want to exit press 0"))
5     sum += num
6   }while(num != 0);
7
8   console.log(sum)
```

17. Write a program that takes a number from 1 to 7 and prints the corresponding day of the week using a switch statement. Use a for loop to test your program with all numbers from 1 to 7.

```
function getDayOfWeek(num) {
  switch (num) {
    case 1:
      return "Monday";
    case 2:
      return "Tuesday";
    case 3:
      return "Wednesday";
    case 4:
      return "Thursday";
    case 5:
      return "Friday";
    case 6:
      return "Saturday";
    case 7:
      return "Sunday";
    default:
      return "invalid input"
  }
  return day;
}

for (let i = 1; i <= 7; i++) {
  console.log(i + " : " + getDayOfWeek(i));
}
```