Day 28 task

Invoke function :

In JavaScript, functions are known as the building blocks based on a set of statements. These sets of statements are used to perform defined specific tasks. The functions can take input values called parameters and return an output value if required. You can use the defined function multiple times just by calling it because functions are reusable pieces of code.

Difference between break and continue :

The break statement "jumps out" of a loop. The continue statement "jumps over" one iteration in the loop.

Function type:

There are 3 ways of writing a function in JavaScript:

- **Function Declaration.**
- **Function Expression.**
- **Arrow Function.**

A function expression may be a part of a larger expression. One can define "named" function expressions (where the name of the expression might be used in the call stack for example) or "anonymous" function expressions.

Here is an example of an anonymous function expression (the name is not used):

```
var myFunction = function() {
  statements
}
```

It is also possible to provide a name inside the definition in order to create a named function expression:

```
var myFunction = function namedFunction() {
  statements
}
```

String Method:

This is easy — you use the length property. Try entering the following lines:

```
const browserType = 'mozilla';
browserType.length;
```

This should return the number 7, because "mozilla" is 7 characters long.

Finding the position of a substring in a string

the string contains the substring, indexOf() returns the index of the first occurrence of the substring. If the string does not contain the substring, indexOf() returns -1.

```
const tagline = 'MDN - Resources for developers, by developers';
console.log(tagline.indexOf('developers')); // 20
```

Copy to Clipboard

Starting at 0, if you count the number of characters (including the whitespace) from the beginning of the string, the first occurrence of the substring "developers" is at index 20.

Extracting a substring from a string

You can extract a substring from a string using the slice() method. You pass it:

- the index at which to start extracting
- the index at which to stop extracting. This is exclusive, meaning that the character at this index is not included in the extracted substring.

For example:

```
const browserType = 'mozilla';
console.log(browserType.slice(1, 4)); // "ozi"
```

The character at index 1 is "o", and the character at index 4 is "l". So we extract all characters starting at "o" and ending just before "l", giving us "ozi".

Difference between replace and replaceAll:

replaceAll()

The replaceAll() method returns a new string with all matches of a pattern replaced by a replacement. The pattern can be a string or a RegExp, and the replacement can be a string or a function to be called for each match.

const p = 'The quick brown fox jumps over the lazy dog. If the dog reacted, was it really lazy?';

console.log(p.replaceAll('dog', 'monkey'));

Replace ()

The `replace()` method searches a string for a value or a regular expression.

The `replace()` method returns a new string with the value(s) replaced.

The `replace()` method does not change the original string.

```
string.replace(searchValue, newValue)
```

`Ternary operator:`

The conditional (ternary) operator is the only JavaScript operator that takes three operands: a condition followed by a question mark (`?`), then an expression to execute if the condition is truthy followed by a colon (`:`), and finally the expression to execute if the condition is falsy. This operator is frequently used as an alternative to an if...else statement.

Parameters

condition
An expression whose value is used as a condition.

exprIfTrue

An expression which is executed if the condition evaluates to a truthy value (one which equals or can be converted to true).

exprIfFalse
An expression which is executed if the condition is falsy (that is, has a value which can be converted to false).

```
var age = 26;
var beverage = (age >= 21) ? "Beer" : "Juice";
console.log(beverage); // "Beer"
```