==========Strings==========

A string is a sequence of letters, numbers, special characters and arithmetic values or combination of all. Strings can be created by enclosing the string literal (i.e. string characters) either within single quotes (’) or double quotes (''),

```
var myString = ’Hello World!’; // Single quoted string
var myString = "Hello World!"; // Double quoted string
```

You can use quotes inside a string, as long as they don’t match the quotes surrounding the string:

```
var str1 = "it’s okay";
var str2 = ’He said "Goodbye"’;
var str3 = "She replied ’Calm down, please’";
var str4 = ’Hi, there!"; // Syntax error - quotes must match
```

**However, you can still use single quotes inside a single quoted strings or double quotes inside double q uoted strings by escaping the quotes with a backslash character (\),

```
var str1 = ’it\’s okay’;
var str2 = "He said \"Goodbye\"";
var str3 = ’She replied \’Calm down, please\’’;
```

The backslash (\) is called an escape character, whereas the sequences \’ and \" that we’ve used in the e xample above are called escape sequences.

JavaScript Escape Sequences:
==========
Escape sequences are also useful for situations where you want to use characters that can’t be typed usi ng a keyboard. Here are some other most commonly used escape sequences.

\n is replaced by the newline character
\t is replaced by the tab character
\r is replaced by the carriage-return character
\b is replaced by the backspace character
\\ is replaced by a single backslash (\)
Here’s an example to clarify the how escape sequences actually works:

```
var str1 = "The quick brown fox \n jumps over the lazy dog.";
document.write("<pre>" + str1 + "</pre>"); // Create line break

var str2 = "C:\Users\Downloads";
document.write(str2); // Prints C:UsersDownloads
```

```
var str3 = "C:\\Users\\Downloads";
document.write(str3); // Prints C:\Users\Downloads
```

Getting the Length of a String:

The length property returns the length of the string, which is the number of characters contained in the stri ng. This includes the number of special characters as well, such as \t or \n.

```
var str1 = "This is a paragraph of text.";
document.write(str1.length); // Prints 28

var str2 = "This is a \n paragraph of text.";
document.write(str2.length); // Prints 30, because \n is only one character
```

Note: Since length is a property, not a function, so don’t use parentheses after it like str.length().
Instead just write str.length, otherwise it will produce an error.

Finding a String Inside Another String;

You can use the indexOf() method to find a substring or string within another string.
This method returns the index or position of the first occurrence of a specified string within a string.

```
var str = "If the facts don’t fit the theory, change the facts.";
var pos = str.indexOf("facts");
alert(pos); // 0utputs: 7
```

Similarly, you can use the lastIndexOf() method to get the index or position of the last occurrence of the s pecified string within a string,

```
var str = "If the facts don’t fit the theory, change the facts.";
var pos = str.lastIndexOf("facts");
alert(pos); // 0utputs: 46
```

```
var str = "If the facts don’t fit the theory, change the facts.";

// Searching forwards
var pos1 = str.indexOf("facts", 20);
alert(pos1); // 0utputs: 46

// Searching backwards
```

```
var pos2 = str.lastIndexOf("facts", 20);
alert(pos2); // 0utputs: 7
```

Searching for a Pattern Inside a String:

You can use the search() method to search a particular piece of text or pattern inside a string.

**Like indexOf() method the search() method also returns the index of the first match, and returns -1 if no matches were found,
but unlike indexOf() method this method can also take a regular expression as its argument to provide advanced search capabilities

```
var str = "Color red looks brighter than color blue.";

// Case sensitive search
var pos1 = str.search("color");
alert(pos1); // 0utputs: 30

// Case insensitive search using regexp
var pos2 = str.search(/color/i);
alert(pos2); // 0utputs: 0
```

Note: The search() method does not support global searches; it ignores the g flag or modifier (i.e. /pattern/g) of its regular expression argument.

Extracting a Substring from a String:

You can use the slice() method to extract a part or substring from a string.

This method takes 2 parameters: start index (index at which to begin extraction),
and an optional end index (index before which to end extraction),
like str.slice(startIndex, endIndex).

```
var str = "The quick brown fox jumps over the lazy dog.";
var subStr = str.slice(4, 15);
document.write(subStr); // Prints: quick brown
```

```
var str = "The quick brown fox jumps over the lazy dog.";
document.write(str.slice(-28, -19)); // Prints: fox jumps
document.write(str.slice(31)); // Prints: the lazy dog.
```

```
var str = "The quick brown fox jumps over the lazy dog.";
```

```
document.write(str.substring(4, 15)); // Prints: quick brown
document.write(str.substring(9, 0)); // Prints: The quick
document.write(str.substring(-28, -19)); // Prints nothing
document.write(str.substring(31)); // Prints: the lazy dog.


var str = "The quick brown fox jumps over the lazy dog.";
document.write(str.substr(4, 15)); // Prints: quick brown fox
document.write(str.substr(-28, -19)); // Prints nothing
document.write(str.substr(-28, 9)); // Prints: fox jumps
document.write(str.substr(31)); // Prints: the lazy dog.
```

Replacing the Contents of a String:

You can use the replace() method to replace part of a string with another string.
This method takes two parameters a regular expression to match or substring to be replaced and a replac ement string,
like str.replace(regexp|substr, newSubstr).

This replace() method returns a new string, it doesn't affect the original string that will remain unchanged.

```
var str = "Color red looks brighter than color blue.";
var result = str.replace("color", "paint");
alert(result); // 0utputs: Color red looks brighter than paint blue.


var str = "Color red looks brighter than color blue.";
var result = str.replace(/color/i, "paint");
alert(result); // 0utputs: paint red looks brighter than color blue.
```

**You can use the toUpperCase() method to convert a string to uppercase,

```
var str = "Hello World!";
var result = str.toUpperCase();
document.write(result); // Prints: HELLO WORLD!
```

Similarly, you can use the toLowerCase() to convert a string to lowercase:

```
var str = "Hello World!";
var result = str.toLowerCase();
document.write(result); // Prints: hello world!
```

*****You can concatenate or combine two or more strings using the + and += assignment operators.

```
var hello = "Hello";
```

```
var world = "World";
var greet = hello + " " + world;
document.write(greet); // Prints: Hello World

var wish  = "Happy";
    wish += " New Year";
document.write(wish); // Prints: Happy New Year
```

Accessing Individual Characters from a String:

You can use the charAt() method to access individual character from a string, like str.charAt(index).
The index specified should be an integer between 0 and str.length - 1.
If no index is provided the first character in the string is returned, since the default is 0.

```
var str = "Hello World!";
document.write(str.charAt());  // Prints: H
document.write(str.charAt(6)); // Prints: W
document.write(str.charAt(30)); // Prints nothing
document.write(str.charAt(str.length - 1)); // Prints: !
```

There is even better way to do this. Since ECMAScript 5, strings can be treated like read-only arrays, and you can access individual characters from a string using square brackets ([]) instead of the charAt() method

```
var str = "Hello World!";
document.write(str[0]); // Prints: H
document.write(str[6]); // Prints: W
document.write(str[str.length - 1]); // Prints: !
document.write(str[30]); // Prints: undefined
```