

# Text Processing

|   |   |   |   |   |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |
| H | E | L | L | O |

**SoftUni Team**  
Technical Trainers



**SoftUni**



**Software University**

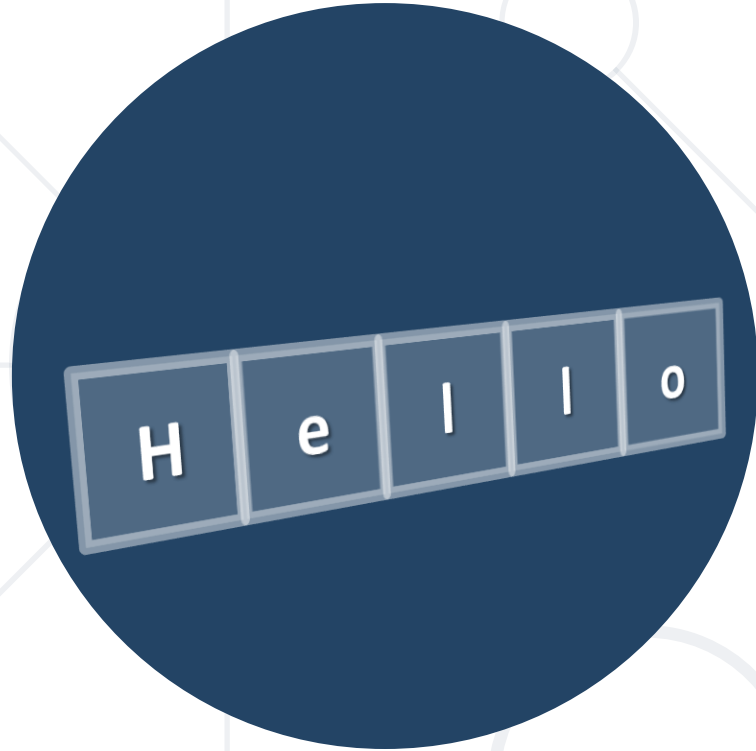
<https://softuni.bg>

1. Strings in JavaScript
2. Manipulating Strings
  - Searching, Substring
  - Trim
  - Split
  - More Functions



sli.do

#fund-js



# Strings

What is String?

# What is String?

- Strings are sequences of characters
  - Like arrays, they have a **length** property
- Strings can be declared with **three types** of quotes

|                                 |                                 |                                 |
|---------------------------------|---------------------------------|---------------------------------|
| <code>let str = "Hello";</code> | <code>let str = 'Hello';</code> | <code>let str = `Hello`;</code> |
|---------------------------------|---------------------------------|---------------------------------|


- Concatenated using the "+" operator

|  |
|--|
| <code>let s = "Hello" + " " + "JS";</code> |
|--|



# Strings Are Immutable

- Strings are **immutable** (read-only) sequences of characters
- Accessible by **index**

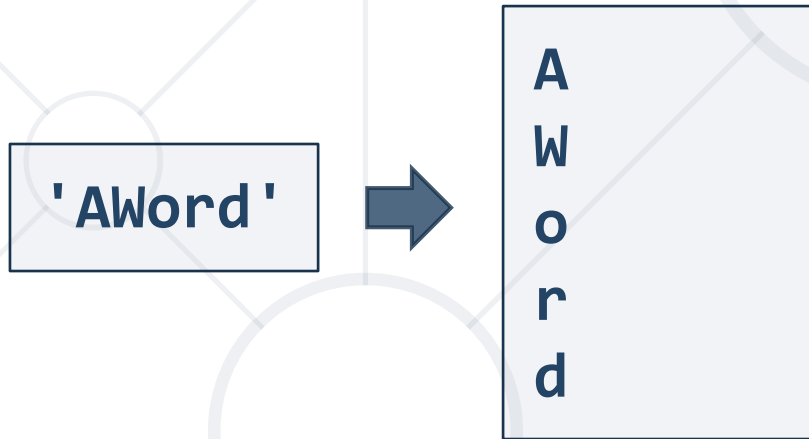


```
let str = "Hello, JS";  
let ch = str[2]; // Expected output: l  
ch = str.charAt(2); // Expected output: l  
// Both declarations are the same
```

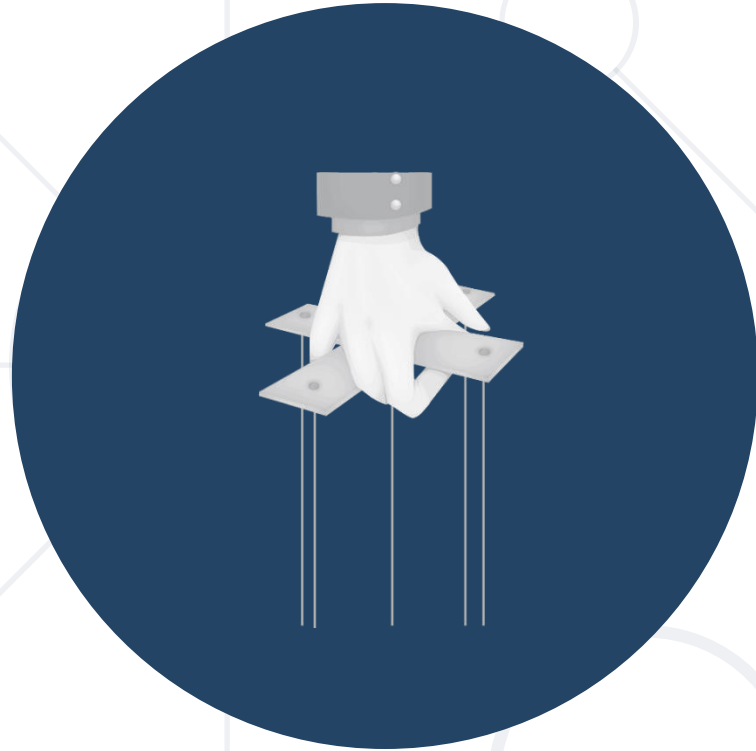
- Strings can also be **iterated** using **for-of**

# Problem: Print Characters

- Receive a **string**
- Print all the characters on separate lines



```
function solve(string) {  
  for (let ch of string) {  
    console.log(ch);  
  }  
}
```



# Manipulating Strings



- Use the "+" or the "+=" operators

```
let text = "Hello" + ", ";  
// Expected output: "Hello, "  
text += "JS!"; // "Hello, JS!"
```

- Use the `concat()` method

```
let greet = "Hello, ";  
let name = "John";  
let result = greet.concat(name);  
console.log(result); // Expected output: "Hello, John"
```

- **indexOf(substr)**

```
let str = "I am JavaScript developer";  
console.log(str.indexOf("Java")); // Expected output: 5  
console.log(str.indexOf("java")); // Expected output: -1
```

- **lastIndexOf(substr)**

```
let str = "Intro to programming";  
let last = str.lastIndexOf("o");  
console.log(last); // Expected output: 11
```

- **substring(startIndex, endIndex)**

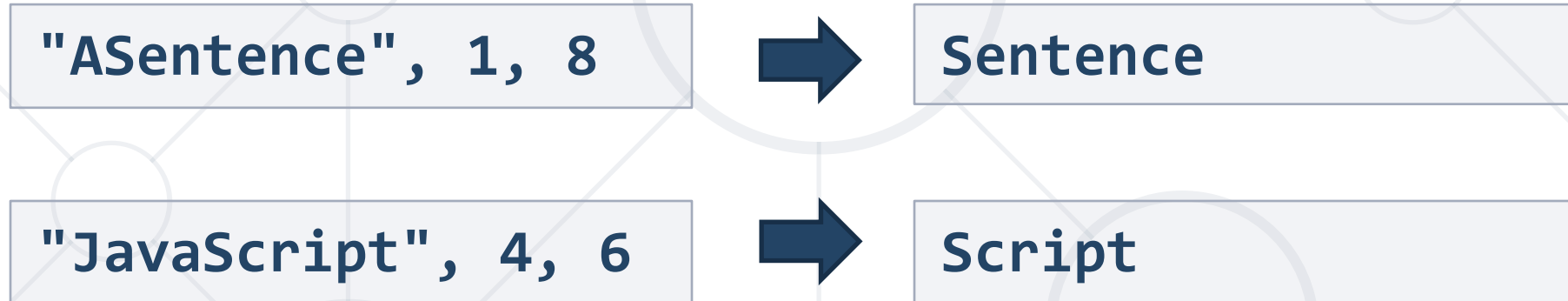
```
let str = "I am JavaScript developer";  
let sub = str.substring(5, 10);  
console.log(sub); // Expected output: JavaS
```

- `replace(search, replacement)`

```
let text = "Hello, john@softuni.bg, you have been  
using john@softuni.bg in your registration.";  
let replacedText = text.replace(".bg", ".com");  
console.log(replacedText);  
// Hello, john@softuni.com, you have been using  
john@softuni.bg in your registration.
```

# Problem: Substring

- Receives a **string**, a **start index**, and **count** characters
- Print the **substring** of the received string



# Solution: Substring

```
function solve(text, startIndex, count) {  
  let substring = text  
    .substring(startIndex, startIndex + count);  
  console.log(substring);  
}
```

- **split(separator)**

```
let text = "I love fruits";  
let words = text.split(' ');  
console.log(words); // Expected output: ['I', 'love', 'fruits']
```

- **includes(substr)**

```
let text = "I love fruits.";  
console.log(text.includes("fruits")); // Expected output: True  
console.log(text.includes("banana")); // Expected output: False
```

- **repeat(count)** - Creates a new string repeated count times

```
let n = 3;  
for(let i = 1; i <= n; i++) {  
  console.log('*'.repeat(i));  
}
```



```
// *  
// **  
// ***
```



# Problem: Censored Words

- Receives a **text** and a **single word**
- Find all **occurrences** of that word in the text and **replace** them with the corresponding amount of **'\*'**

A small sentence with some words,  
small



A \*\*\*\*\* sentence with some words

# Solution: Censored Words

```
function solve(text, word) {  
  while (text.includes(word)) {  
    text = text.replace(word, '*'.repeat(word.length));  
  }  
  console.log(text);  
}
```

- Use **trim()** method to remove **whitespaces** (spaces, tabs, no-break space, etc. ) from **both ends** of a string

```
let text = "    Annoying spaces    ";  
console.log(text.trim()); // Expected output: "Annoying spaces"
```

- Use **trimStart()** or **trimEnd()** to remove whitespaces **only** at the beginning or at the end

```
let text = "    Annoying spaces    ";  
text = text.trimStart(); text = text.trimEnd();  
console.log(text); // Expected output: "Annoying spaces"
```

# Starts With/Ends with

- Use **startsWith()** to determine whether a string **begins** with the characters of a specified substring

```
let text = "My name is John";  
console.log(text.startsWith('My')); // Expected output: true
```

- Use **endsWith()** to determine whether a string **ends** with the characters of a specified substring

```
let text = "My name is John";  
console.log(text.endsWith('John')); // Expected output: true
```

# Padding at the Start and End

- Use **padStart()** to add to the current string **another substring** at the **start** until a **length** is reached

Receives **length** and **substring**

```
let text = "010";  
console.log(text.padStart(8, '0')); // Expected output: 00000010
```

- Use **padEnd()** to add to the current string **another substring** at the **end** until a **length** is reached

```
let sentence = "He passed away";  
console.log(sentence.padEnd(20, '.'));  
// Expected output: He passed away.....
```

# Problem: Count String Occurrences

- Receive a **text** and a **word** that you need to **search**
- Find the number of **all occurrences** of that word and print it

"This is a word and it also is a sentence",  
"is"



2

# Solution: Count String Occurrences

```
function solve(text, search) {  
  let words = text.split(' ');  
  let counter = 0;  
  for (let w of words) {  
    if (w === search) {  
      counter++;  
    }  
  }  
  console.log(counter);  
}
```



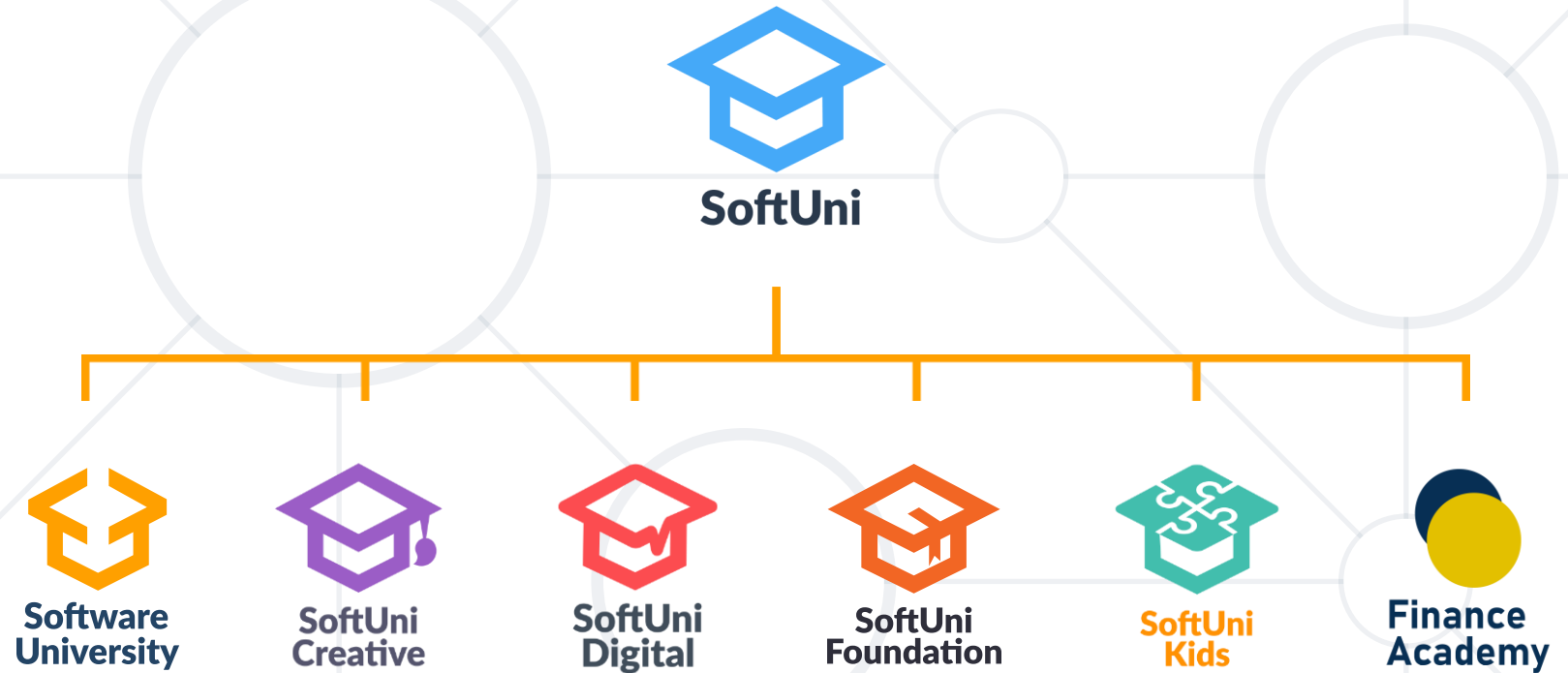
**Live Exercises**



- Strings are **immutable** sequences of Unicode characters
- Strings can be **indexed** like arrays
- String processing methods
  - **concat(), indexOf(), includes(), substring(), split()**



# Questions?



# SoftUni Diamond Partners

**SUPER  
HOSTING  
.BG**



**Coca-Cola HBC  
Bulgaria**



**POKERSTARS**  
POKER | CASINO | SPORTS  
a Flutter International brand

**INDEAVR**  
Serving the high achievers



**AMBITIONED**

 **DRAFT  
KINGS**



**SOFTWARE  
GROUP**

createX



**Postbank**

Решения за твоето утре

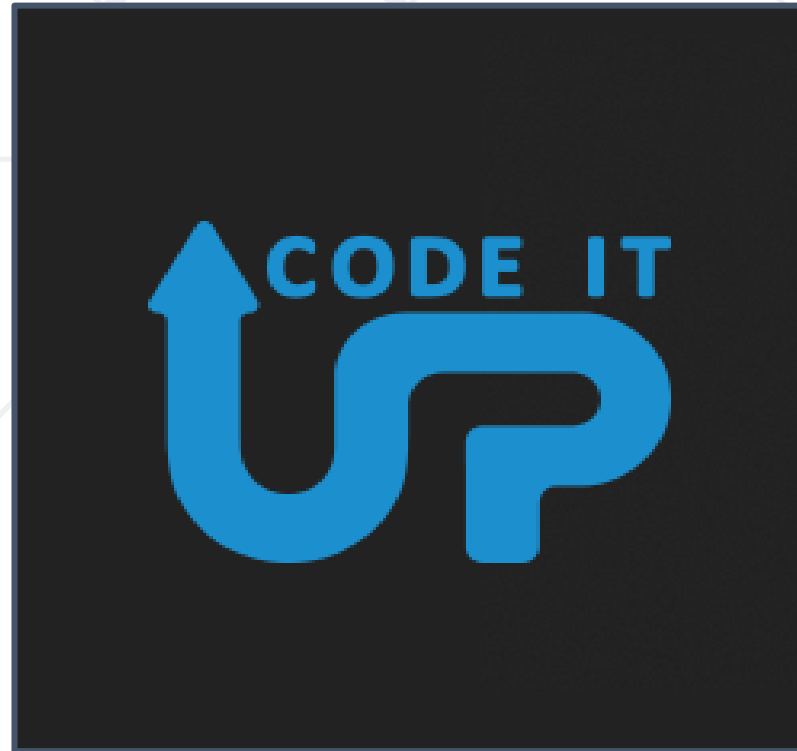


**BOSCH**

**DXC**  
TECHNOLOGY



**SmartIT**



- Software University – High-Quality Education, Profession and Job for Software Developers
  - [softuni.bg](http://softuni.bg)
  - Software University Foundation
    - [softuni.foundation](http://softuni.foundation)
- Software University @ Facebook
  - [facebook.com/SoftwareUniversity](https://facebook.com/SoftwareUniversity)
- Software University Forums
  - [forum.softuni.bg](http://forum.softuni.bg)



- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://about.softuni.bg/>
- © Software University – <https://softuni.bg>

