Functions and Arrow Functions

Functions, Parameters, Return Value, Arrow Functions (Lambda)

SoftUni Team
Technical Trainers

Software University

http://softuni.bg

Table of Contents

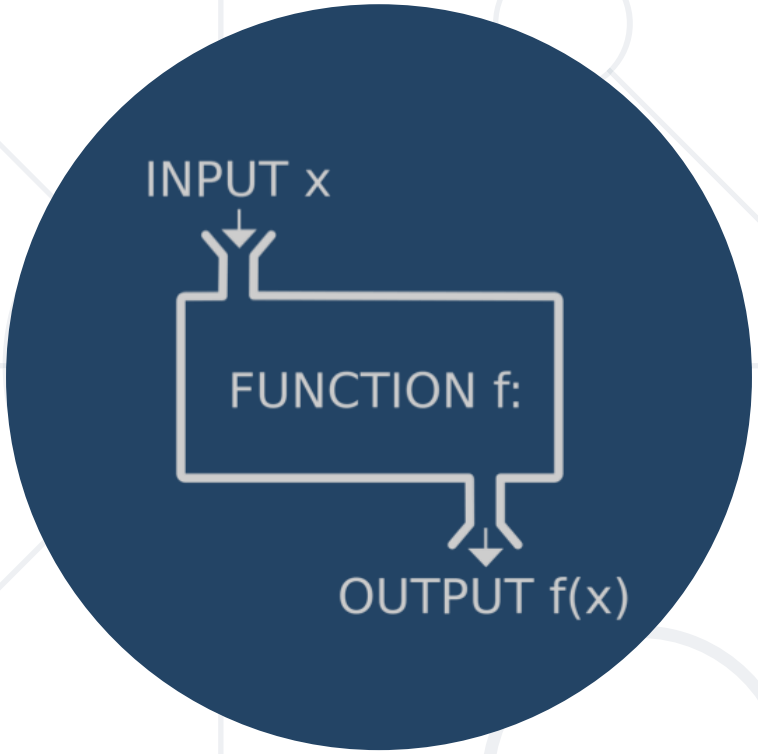
SoftUni Foundation

Have a Question?

SoftUni Foundation

sli.do

#JSCORE



JavaScript Functions Overview

Declaring and Invoking Functions

Functions in JS

- **Function** == named piece of code
  - Can take **parameters** and return **result**

Function name: use **camelCase**

Function **parameters**: use **camelCase**

```
function printStars(count) {
  console.log("*".repeat(count));
}
```

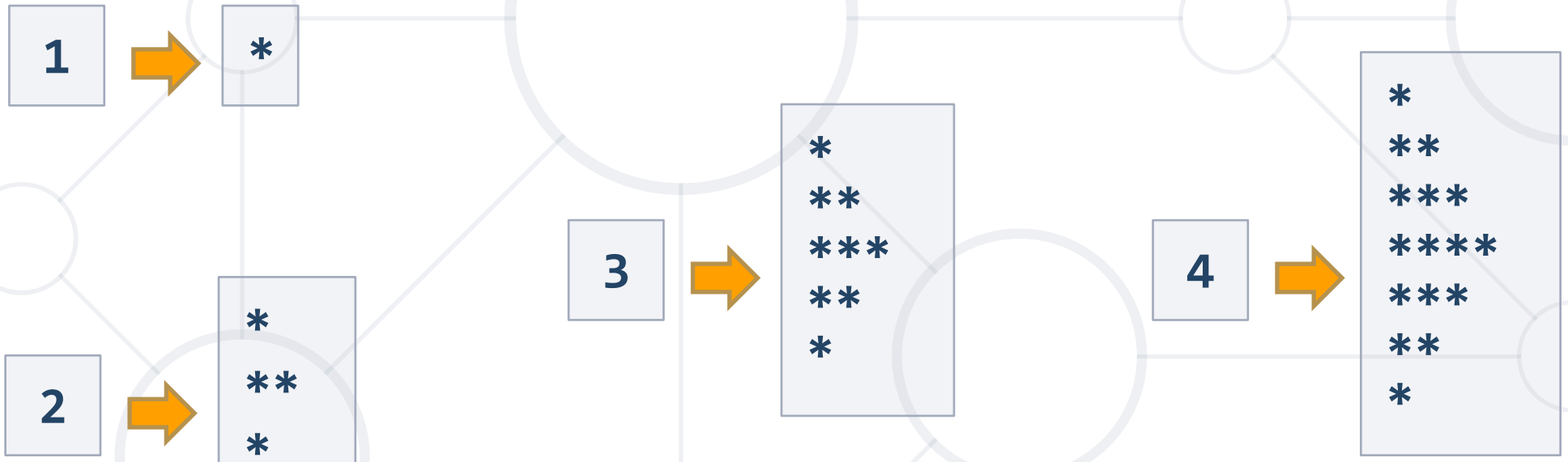
The { stays at the same line

```
printStars(10);
```

Invoke the function

Problem: Triangle of Stars

- Write a JS function to print a triangle of stars of size n



Solution: Triangle of Stars

- Functions in JS can be nested (function inside a function)

```
function printTriangle(n) {
  function printStars(count) {
    console.log("*".repeat(count));
  }

  for (let i=1; i<=n; i++) printStars(i);
  for (let i=n-1; i>0; i--) printStars(i);
}
```

```
*
**
***
**
*
```

```
printTriangle(3);
```

Check your solution here: https://judge.softuni.bg/Contests/306

Default Function Parameter Values

SoftUni Foundation

- Functions in JS can have default parameter values

```
function printStars(count = 5) {
  console.log("*".repeat(count));
}
```

```
printStars(); // *****
```

```
printStars(2); // **
```

```
printStars(3, 5, 8); // ***
```

Problem: Square of Stars

- Write a JS function to print a square of stars

```
function squareOfStars(n) {
  function printStars(count = n) {
    console.log("*" +
      " *".repeat(count-1));
  }
  for (let i=1; i<=n; i++)
    printStars();
}
```

3 →

```
* * *
* * *
* * *
```

4 →

```
* * * *
* * * *
* * * *
* * * *
```

Check your solution here: https://judge.softuni.bg/Contests/306

Function Overloading

- In C# / Java / C++ functions can be overloaded
  - **Function overloading** == same name, different parameters
- JavaScript (like Python and PHP) does not support overloading

```
function printName(firstName, lastName) {
  let name = firstName;
  if (lastName != undefined)
    name += ' ' + lastName;
  console.log(name);
}
```

Simulate overloading by parameter checks

```
printName('Maria');
printName('Maria','Nikolova');
```

Variable Number of Arguments

- JS functions have special array **arguments**

```
function sum() {
  console.log("args count: " + arguments.length);
  console.log(arguments);
  let sum = 0;
  for (let x of arguments)
    sum += x;
  console.log("sum = " + sum);
}
```

```
sum(); // 0 [] 0
```

```
sum(5, 3); // 2 [5, 3] 8
```

```
sum(4, 2, 3); // 3 [4, 2, 3] 9
```

Returning Values from a Function

Functions Can Return Values

```
function multiply(a, b) {
  return a * b;
}
```

```
let m = multiply(3, 5);
console.log(m); // 15
```

```
function hello() {
  console.log("hello");
}
```

```
let v = hello();
console.log(v); // undefined
```

Returning Values – Examples

```
function check(a) {
  if (a > 0)
    return "positive";
  if (a < 0)
    return "negative";
}
```

The function sometimes returns a **string**, sometimes returns **undefined**

```
console.log(check(5)); // positive
console.log(check(-5)); // negative
console.log(check(0)); // undefined
console.log(check()); // undefined
console.log(check("hello")); // undefined
```

Problem: Symmetry Check (Palindrome)

SoftUni Foundation

- Write a JS function to check a string for symmetry
  - Examples: "abcccba" → true; "xyz" → false

```
function isPalindrome(str) {
  for (let i=0; i<str.length/2; i++)
    if (str[i] !== str[str.length-i-1])
      return false;
  return true;
}
```

```
isPalindrome("abba"); // true
```

Check your solution here: https://judge.softuni.bg/Contests/306

Problem: Day of Week

- Write a JS function to return the day number by day of week
  - Example: "Monday" → 1, …, "Sunday" → 7, other → "error"

```
function dayOfWeek(day) {
  if (day === 'Monday') return 1;
  …
  if (day === 'Sunday') return 7;
  return "error";
}
```

JS functions can return **mixed data type**: e.g. number or string

```
dayOfWeek("Monday"); // 1
```

Check your solution here: https://judge.softuni.bg/Contests/306

Function Variables

Variables Holding Functions

Variables Holding Functions

SoftUni Foundation

- In JS variables can hold functions as their values

```
let f = function(x) { return x * x; }
console.log(f(3)); // 9
console.log(f(5)); // 25

f = function(x) { return 2 * x; }
console.log(f(3)); // 6
console.log(f(5)); // 10

f = undefined;
console.log(f(3)); // TypeError: f is not a function(…)
```

Functions as Parameters

```
function repeatIt(count, func) {
  for (let i = 1; i <= count; i++)
    func(i);
}

let starsFunc = function(i) {
  console.log("**".repeat(i))
};

repeatIt(3, starsFunc);

repeatIt(3, function(x) { console.log(2 * x); } );
```

```
**
****
******
```

```
2
4
6
```

Problem: Functional Calculator

- Write a calculator that takes two numbers and an operator and performs a calculation between them using the operator

```
function calculate(a, b, op) {

  let calc = function(a, b, op) { return op(a, b) };
  let add = function(a, b) { return a + b };
  let subtract = function(a, b) { return a - b };
  let multiply = function(a, b) { return a * b };
  let divide = function(a, b) { return a / b };
```

Problem: Functional Calculator (2)

```
  switch (op) {
    case '+': return calc(a, b, add);
    case '-': return calc(a, b, subtract);
    case '*': return calc(a, b, multiply);
    case '/': return calc(a, b, divide);
  }
}
```

```
console.log(calculate(2, 4, '+')) // 6
```

```
console.log(calculate(9, 2, '/')) // 4.5
```

Check your solution here: https://judge.softuni.bg/Contests/306

IIFE

- Immediately-invoked function expression (IIFE)

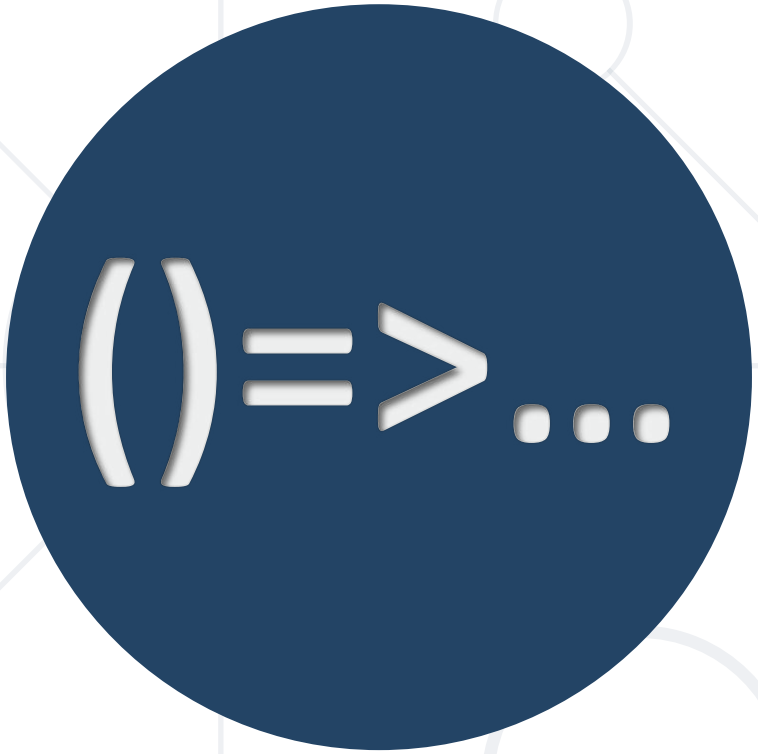
```
(function (count) {
  for (let i = 1; i <= count; i++)
    console.log('+'.repeat(i));
})(4);
```

```
+
++
+++
++++
```

```
let f = (function () {
  let x = 0;
  return function() { console.log(++x); }
})(); f(); f(); f(); f();
```

This is called "**closure**" (a state is closed inside)

```
1
2
3
4
```



Arrow (Lambda) Functions in JS

Short Syntax for Anonymous Functions

Arrow Functions

- Functions in JS can be written in short form using "=>" (arrow)

```
let increment = x => x + 1;
console.log(increment(5)); // 6
```

```
let increment = function(x) {
  return x + 1;
}
```

This is the same as the above function

```
let sum = (a, b) => a + b;
console.log(sum(5, 6)); // 11
```

Problem: Aggregate Elements

SoftUni Foundation

- Write a function to aggregate elements
  - The elements are given as array, e.g. [1, 2, 3]
  - Start by given initial value, e.g. 0
  - At each iteration apply given aggregate function e.g. a + b

```
aggregate([10, 20, 30], 0, (a, b) => a + b); // 60
```

```
aggregate([10, 20, 30], 1, (a, b) => a * b); // 6000
```

Input elements

Initial value

Aggregate functioan

Problem: Sum / Inverse Sum / Concatenate

SoftUni Foundation

- Using the aggregating function, calculate:
  - Sum of elements
    - e.g. [1, 2, 4] → 1 + 2 + 4 → 7
  - Sum of inverse elements ($1/a_i$)
    - E.g. [1, 2, 4] → 1/1 + 1/2 +1/4 → 7/4 → 3.5
  - Concatenation of elements
    - e.g. ['1', '2', '4'] → '1'+'2'+'4' → '124'

Solution: Aggregate Elements

SoftUni Foundation

```
function aggregateElements(elements) {
  aggregate(elements, 0, (a, b) => a + b);
  aggregate(elements, 0, (a, b) => a + 1 / b);
  aggregate(elements, '', (a, b) => a + b);
  function aggregate(arr, initVal, func) {
    let val = initVal;
    for (let i = 0; i < arr.length; i++)
      val = func(val, arr[i]);
    console.log(val);
  }
}
```

```
aggregateElements([10, 20, 30]); // 60 1.833 102030
```

Check your solution here: https://judge.softuni.bg/Contests/306

Nested Functions

Problem: Words Uppercase

SoftUni Foundation

- Functions in JS can be nested, i.e. hold other functions
  - Inner functions have access to variables from their parent

```
function wordsUppercase(str) {
  let strUpper = str.toUpperCase();
  let words = extractWords();
  words = words.filter(w => w != '');
  return words.join(', ');

  function extractWords() { return strUpper.split(/\W+/); }
}
```

```
wordsUppercase('Hi, how are you?'); // "HI, HOW, ARE, YOU"
```

```
extractWords('Hello functions'); // ReferenceError
```

Check your solution here: https://judge.softuni.bg/Contests/306

Live Exercises

Summary

SoftUni Foundation

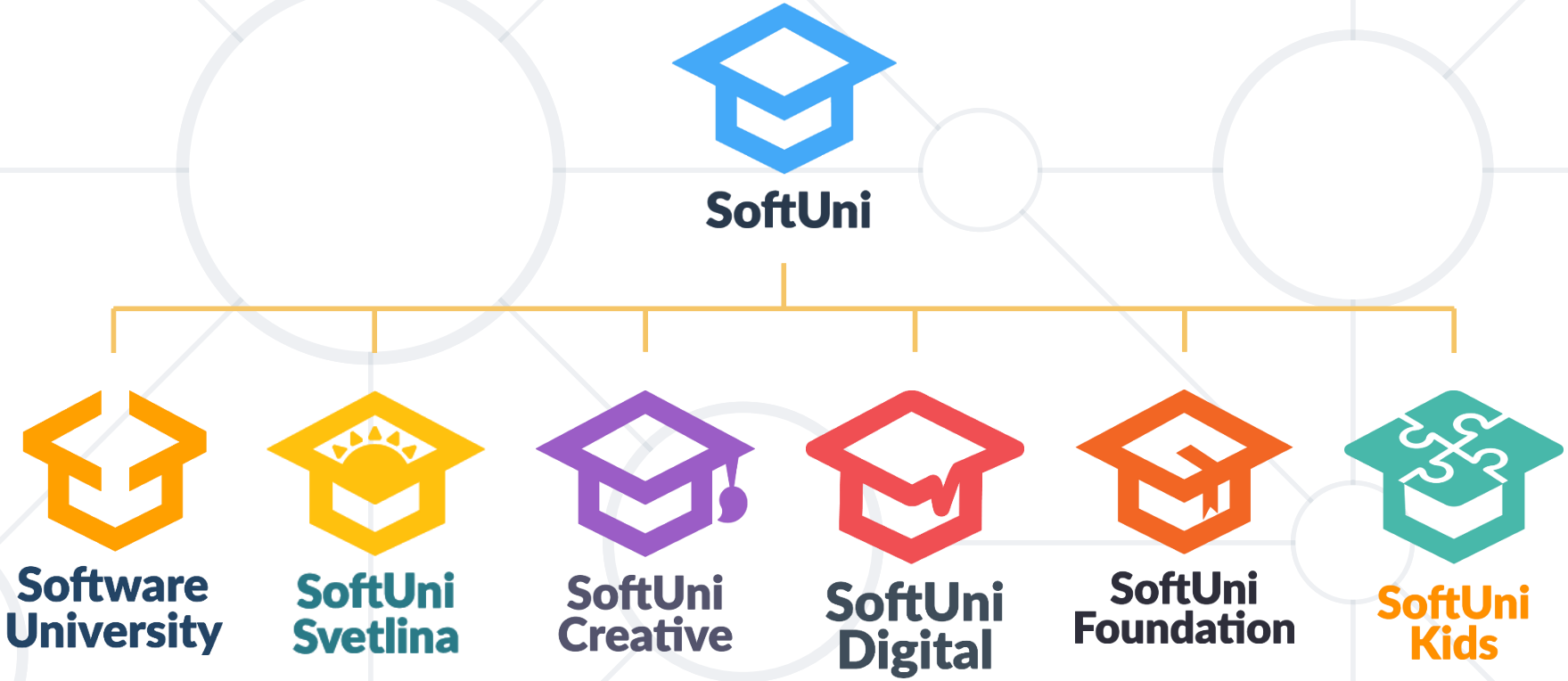
- Function == named piece of code
  - Can take parameters and return result

```
function calcSum(a, b) {
  let sum = a + b;
  return sum;
}
```

  - Arrow functions ≈ short function syntax

```
[10, 20, 30].filter(a => a > 15);
```

Questions?


SoftUni
Software University
SoftUni Svetlina
SoftUni Creative
SoftUni Digital
SoftUni Foundation
SoftUni Kids

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech we know sports

telenor

SoftwareGroup doing it right

NETPEAK

SmartIT

Postbank Решения за твоето утре

SUPER HOSTING .BG

INDEAVR Serving the high achievers

INFRAGISTICS

LIEBHERR

æternity

codexio

SoftUni Organizational Partners

SoftUni Foundation

ИНФОРМАЦИОННО ОБСЛУЖВАНЕ

OneBit SOFTWARE

WORLD OF MYTHS

Lukanet.com

codexio

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license