

JS

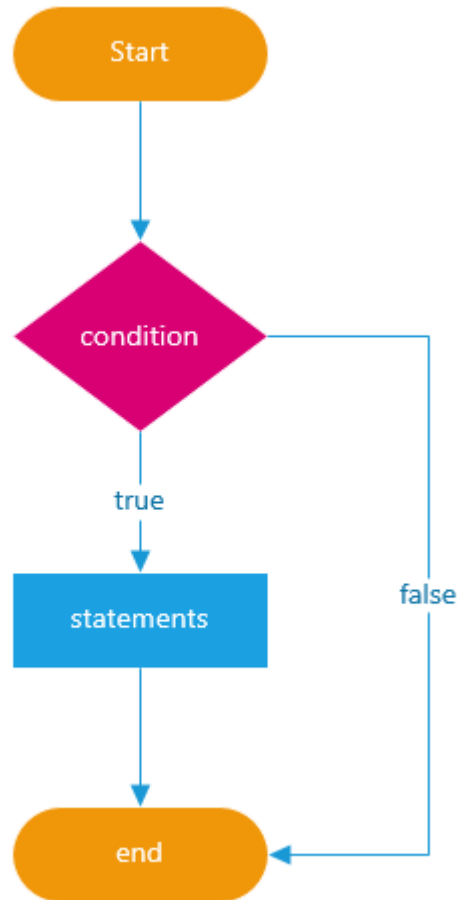
SCSA JavaScript Courses

1) IF_ELSE STATEMENTS

2) STRICT MODE

3) FUNCTIONS, DECLARATIONS VS.
EXPRESSIONS, ARROW FUNCTIONS,
FUNCTIONS CALLING OTHER FUNCTIONS

IF_else Statements



Conditional statements are used to decide the flow of execution based on different conditions. If a condition is true, you can perform one action and if the condition is false, you can perform another action.

If statement

If Statement

```
if (condition)
{
  lines of code to be executed if
  condition is true
}
```

- You can use If statement if you want to check only a specific condition.

If...Else statement

```
if (condition){
  lines of code to be executed if the
  condition is true
}
else
{
  lines of code to be executed if the
  condition is false
}
```

You can use If....Else statement if you have to check two conditions and execute a different set of codes.

If...Else If...Else statement

```
if (condition1)
{
  lines of code to be executed if
  condition1 is true }
else if(condition2)
{
  lines of code to be executed if
  condition2 is true
}
else {
  lines of code to be executed if
  condition1 is false and condition2
  is false
}
```

დავალება 1

// დავითის ასაკია 25. ანას ასაკი არის 23;

// გამოიყვანე ბრძანება რომლის შეამოწმებს რომლის ასაკი არის მეტი და რამდენით.

//output: დავითი უფროსია ანაზე 2 წლით. მისი ასაკია 25. ანას ასაკია 23.

//გამოიყენეთ Template Literal შედეგის გამოსაყვანად.

Use Strict

"use strict"; Defines that JavaScript code should be executed in "strict mode".

```
"use strict";  
var x = 0.10;    // This will cause an error  
console.log(x)
```

```
var x = 010;    // This will cause an error  
      ^^^
```

```
SyntaxError: Octal literals are not allowed in  
strict mode.
```

Functions


JavaScript functions are defined with the function keyword. You can use a function declaration or a function expression.

Functions

JavaScript functions are defined with the function keyword. You can use a function declaration or a function expression.

Function declaration


Function that can be used before it's declared



```
// function Declaration
function myfunc(){
    console.log("Its My Function declaration")
}
```

Function expression

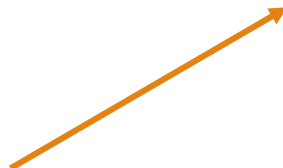
Essentially a function value stored in a variable



```
// function expressino
let myfunc=function(){
    console.log("Its My Function expression")
}
```

Arrow function

Great for a quick one-line functions. Has no this keyword (more later...)



```
//arrow function
let myfunc= () =>{
    console.log('Its My arrow function')
}
```

Parameters

We can pass arbitrary data to functions using parameters (also called function arguments) . In the example below, the function has two parameters: from and text.

```
1 function showMessage(from, text) { // arguments: from, text
2   alert(from + ': ' + text);
3 }
4
5 showMessage('Ann', 'Hello!'); // Ann: Hello! (*)
6 showMessage('Ann', "What's up?"); // Ann: What's up? (**)
```


სავარჯიშო

დაწერეთ ფუნქცია, რომელსაც გადავცემთ 2 პარამეტრს. შეყვანილი რიცხვებიდან გავიდოთ პირველი ატრიბუტია მეტი, თუ მეორე;

Scopes

GLOBAL SCOPE

```
const me = 'Jonas';  
const job = 'teacher';  
const year = 1989;
```

- 👉 Outside of **any** function or block
- 👉 Variables declared in global scope are accessible **everywhere**

FUNCTION SCOPE

```
function calcAge(birthYear) {  
  const now = 2037;  
  const age = now - birthYear;  
  return age;  
}  
  
console.log(now); // ReferenceError
```

- 👉 Variables are accessible only **inside function**, **NOT** outside
- 👉 Also called local scope

BLOCK SCOPE (ES6)

```
if (year >= 1981 && year <= 1996) {  
  const millenial = true;  
  const food = 'Avocado toast';  
} ← Example: if block, for loop block, etc.  
  
console.log(millenial); // ReferenceError
```

- 👉 Variables are accessible only **inside block** (block scoped)
- ⚠️ **HOWEVER**, this only applies to **let** and **const** variables!
- 👉 Functions are **also block scoped** (only in strict mode)

დაწერეთ ფუნქცია, რომელიც გამოითვლის თუ რამდენი წლისაა ძაღლი

- ფუნქციას გადაეცით მხოლოდ ერთი პარამეტრი (age)
- გამოიანგარიშეთ ძაღლის ასაკი, თუ ადამიანის 1 წელი უდრის ძაღლის 7 წელს
- შედეგი: შენი ძაღლი არის 'რაღაც' წლის

```
`use strict`
```

```
const yearsUntilRetirement = function (birthYeah, firstName) {  
  const age = 2021-birthYeah;
```

```
  if (age > 60) {  
    console.log(`${firstName} is retired and his age is ${age}`);  
    return age  
  } else {  
    console.log(`${firstName} is not retired and his age is ${age}`);  
  }  
}
```

```
}  
yearsUntilRetirement(1991, 'Jonas');  
yearsUntilRetirement(1950, 'Mike');
```

Exersice

```
`use strict`
const calcAge = function (birthYeah) {
  return 2021 - birthYeah;
}

const yearsUntilRetirement = function (birthYeah, firstName) {
  const age = calcAge(birthYeah);

  if (age > 60) {
    console.log(`${firstName} is retired and his age is ${age}`);
  } else {
    console.log(`${firstName} is not retired and his age is ${age}`);
  }
}

yearsUntilRetirement(1991, 'Jonas');
yearsUntilRetirement(1950, 'Mike');
```

HomeWork

// გვაქვს ორი საფეხბურთო კლუბი. დინამო და ტორპედო; თითოეულმა გუნდმა ითამაშა 3-ჯერ;

// Task 1: შექმენით Arrow Function, რომელიც გამოიანგარიშებს თითოეული გუნდის გოლების საშუალოს

// Task 2: შექმენით ფუნქცია, რომელიც მოახდენს ამ ორი გუნდის გოლების ჯამის შედარებას; თუ დინამოს გატანილი გოლების ჯამი მეტია ტორპედოს გატანილი გოლების ჯამზე, დაწეროს "დინამოს გუნდის გატანილი გოლების რაოდენობა მეტია ტორპედოს გუნდის გატანილი გოლების რაოდენობაზე თუ არადა პირიქით"

// პირობა: დინამო [2, 1, 1];

// ტორპედო: [3, 2, 1]

HomeWork

გამოიძახეთ prompt ფანჯარა პირველ ფანჯარაში დავწერთ სახელს. მეორე ფანჯარაში დავწერთ დაბადების რიცხვს.გამოიყენეთ ფუნქცია, რომელიც განსაზღვრავს

1) თუ ასაკი არის 17 წელს ქვემოთ დაბეჭდოს. „ ‘სახელი’ არის ‘ასაკი’ თქვენ არ გაქვთ მართვის მოწმობის უფლება“ თუ არის 17 წელს ზემოთ დაბეჭდოს „ ‘სახელი’ არის ‘ასაკი’ თქვენ გაქვთ უფლება გქონდეთ მართვის მოწმობა“