1. Install needed tools

- Node.js
- Visual Studio Code
  - Code Runner Extension (View → Extensions)
  - Node.js extension pack - Includes ESLint

And run "Hello World": `console.log('hello');`

and in command prompt:

```
node hello.js
```

2. Check first node.js version! `node --version` it should be 8.x!

   Create new Node.js project folder. In this folder give command `npm init` to create `package.json` - file. Create `index.js` - file which will be main source file of your app (can be empty for now)

   Now create application that generates passwords, use and install separate module for this locally. See the dependency in `package.json` after installing. Install ESLint also locally (use `--save-dev`). See the dependency in `package.json`. See slides for commands!

   Create preset for ESLint where you decide your code style (`eslint.js --init`).

```
node ./node_modules/eslint/bin/eslint.js --init
```

   Try out also standard JS style code. Install also Visual Studio Code ESLint plugin to use ESLint directly in the editor **if not already.**

   Your app should output random passwords and it should follow some style guide.

   Solution (code)

```
var randomstring = require('randomstring')
console.log(randomstring.generate(7))
```

Variables

Return exercises:

https://www.dropbox.com/request/yIOAXmq4zKn2mnkAGEDF

3. Create application where you demonstrate the differences between `var`, `let` and `const`

Solution

```
function doIt () {
  const X = 1
  // Won't work, you cannot assing const
  //   X = 9
  // Won't work, let is defined in if scope
  //   console.log(number1)
  // Works, var is defined in function scope!
  console.log(number2)
  if (x == 1) {
    let number1 = 12
    var number2 = 99
  }
}
doIt()
```

4. Create application where you demonstrate the differences between *pass by value* and *pass by reference*. In EcmaScript, objects are passed by reference and primitive types pass by value.

   Example of an object:

```
var object = {'key': 'value'}
console.log(object.key)
```

Solution

```
var object = {'key': 'value'}
// Pass by reference!
// When changing o, also object changes!
var o = object
o.key = 'jack'
console.log(object.key) // 'jack'
var num = 4
```

```
// Pass by value!
// When changing num2 it does NOT influence num!
var num2 = num
num2 = 9
console.log(num)  // 4
```

Create application where you demonstrate the differences when using string variables with ", ' or `

Solution

```
console.log("Tässä on merkkijono käyttäen  \"-merkkiä")
console.log('Tässä on merkkijono käyttäen  \'-merkkiä, jolloin voi käyttää helposti myös "-merkkejä sisällä');
console.log(`tällä tavoin voi tehdä monirivisiä merkkijojnoja
    <h1>otsikko</h1>
      <p>
        tekstiä
      </p>`);
```

6. Create application where you have array full of names. Output the first and the last name from the array.

Solution

```
var nimiTaulukko = ['Aapo', 'Elias', 'Akseli', 'Juha', 'Pekka'];

console.log("Ensimmäinen nimi on: "+nimiTaulukko[0]);
console.log("Viimeinen nimi on: "+nimiTaulukko[nimiTaulukko.length - 1]);
```

7. Create application where you output random**floating point value** between `[0,1[`

Solution

```
console.log(Math.random());
```

8. Create application where you output random **integer value** between `[0,9]`. You can use Math -methods for this.

Solution

```
console.log(Math.floor(Math.random()*10));
```

9. Create application where you output a random name from an array that contains names

Solution

```
var names = ["Pekka", "Tiina"]
console.log(names[Math.floor(Math.random()*2)]);
```

10. Node.js (and EcmaScript) contains built-in objects that you can use. One of these built-in objects is called `process`. By using this object try to output one given command line argument to console:

```
> node lab10.js hello
hello
```

Solution

```
console.log(process.argv[2])
```

11. Now modify your app so that user can give two integers and the output is the sum of those integers:

```
> node lab11.js 7 7
14
```

Solution

```
console.log(Number(process.argv[2]) + Number(process.argv[3]))
```