**WEB REPORT 6**

**Group: SE-2429**

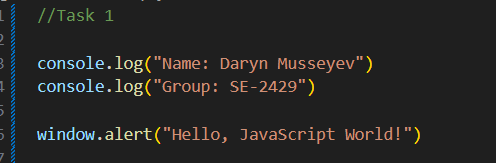
**Name: Daryn Musseyev**

**Task 0. First Script**

**Objective:**  
To create a simple HTML page that includes an external JavaScript file and displays a greeting message.

**Implementation Details:**

* An HTML page named index.html was created.
* The page includes a JavaScript file called script.js using the <script src="script.js"></script> tag.
* Inside script.js, the following actions were performed:
  + The student’s name and group were printed in the browser console using console.log().
  + A pop-up message was displayed using window.alert() with the text **“Hello, JavaScript World!”**.

****

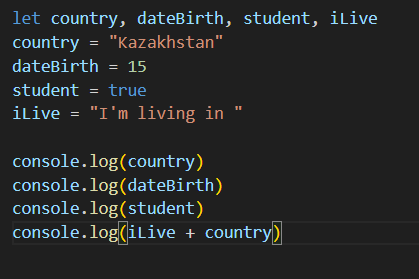
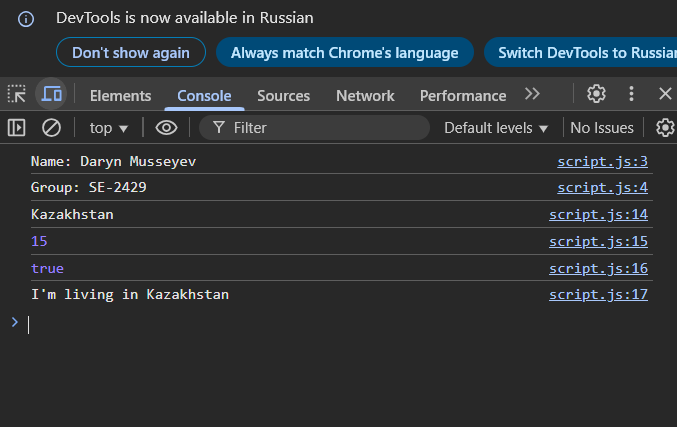
**Task 1. Variables & Operators**

**Objective:**  
To practice declaring variables of different data types and using operators and string concatenation in JavaScript.

**Implementation Details:**

* Variables were declared for different data types:
  + country → String
  + dateBirth → Number
  + student → Boolean
  + iLive → String
* Values were assigned and displayed in the console using console.log().
* String concatenation was demonstrated by combining two strings with the + operator.

**Code Example:**

****

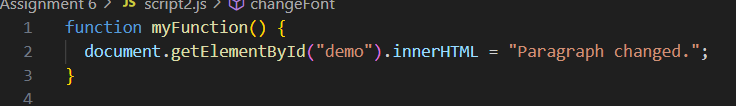
**Task 2. Changing Content**

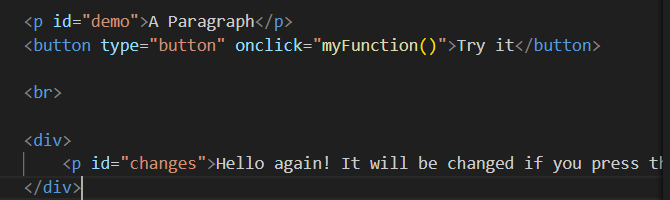
**Objective:**  
To create a paragraph and a button that changes the paragraph’s text when clicked.

**Implementation Details:**

* A paragraph was created with the id="demo".
* A button with an onclick event was added to call the function myFunction().
* The JavaScript function uses document.getElementById() to locate the paragraph and changes its innerHTML to a new message.

**Code Example (HTML + JavaScript):**



****

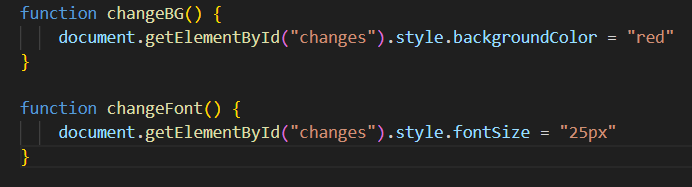
**Task 3. Changing Styles**

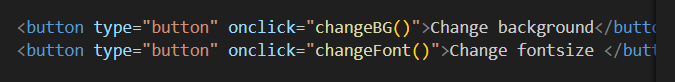
**Objective:**  
To dynamically change the style of an HTML element (background color and font size) using JavaScript.

**Implementation Details:**

* A <div> container was created containing a paragraph with the id="changes".
* Two buttons were added:
  + One to change the background color (changeBG()).
  + Another to change the font size (changeFont()).
* JavaScript functions modify the CSS styles of the element using the .style property.

**Code Example (HTML + JavaScript):**





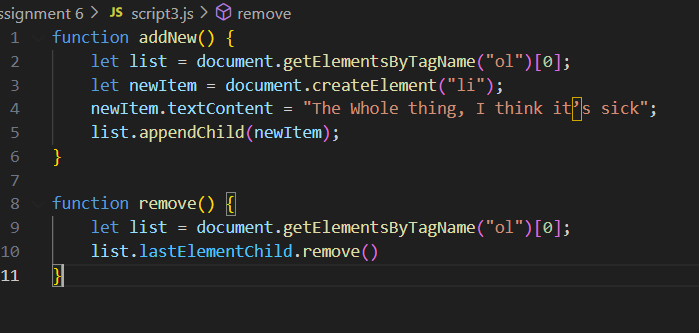
**Task 4. Creating & Removing Elements**

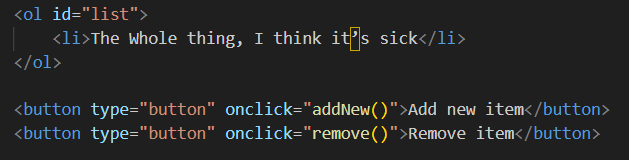
**Objective:**  
To create and remove list items dynamically using JavaScript.

**Implementation Details:**

* An ordered list (<ol>) was created with one default list item.
* Two buttons were added:
  + **“Add new item”** → adds a new <li> element to the list.
  + **“Remove item”** → removes the last <li> element from the list.
* The JavaScript functions use:
  + document.createElement() and .appendChild() to add new items.
  + .lastElementChild.remove() to remove the last item.

**Code Example (HTML + JavaScript):**





**Part 3. Events**

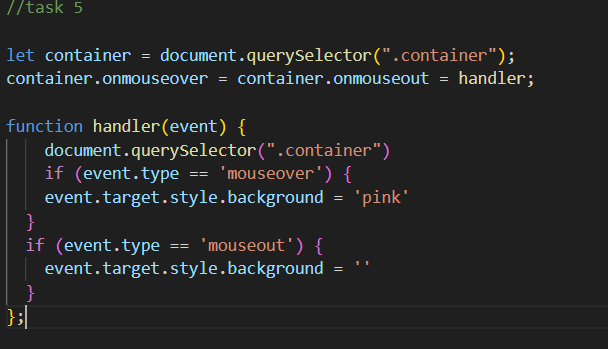
**Task 5. Mouse Events**

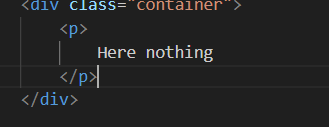
**Objective:**  
To create a box that changes color when the mouse pointer hovers over it and restores the original color when the mouse leaves.

**Implementation Details:**

* A <div> element with the class "container" was created.
* The script uses the onmouseover and onmouseout events to detect when the cursor enters or leaves the box.
* The background color is changed using the .style.background property.
* When the mouse leaves the area, the background color resets to its default.

**Code Example (HTML + JavaScript):**





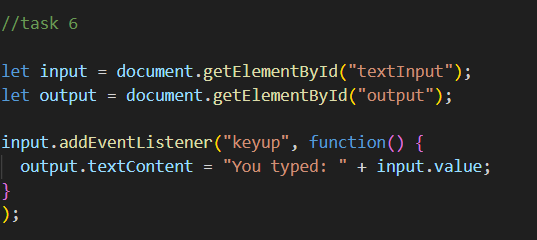
**Task 6. Keyboard Events**

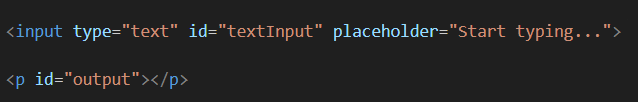
**Objective:**  
To display the current value of an input field in real time as the user types.

**Implementation Details:**

* An <input> field was created with the ID "textInput".
* A <p> element with the ID "output" was used to show the live text.
* A keyup event listener was added to detect every key press release.
* The displayed text updates dynamically with the content of the input field.

**Code Example (HTML + JavaScript):**





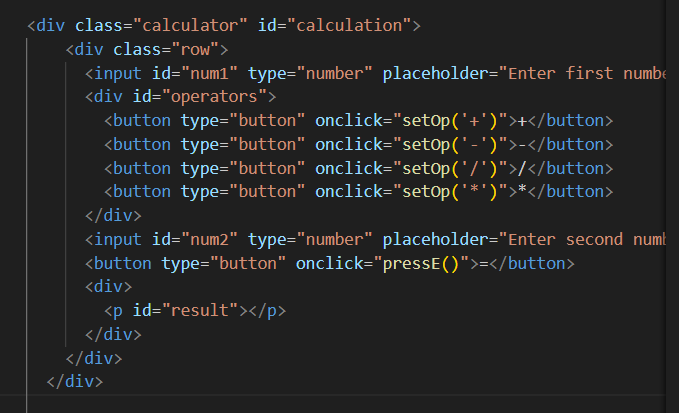
**Task 7. Calculator**

**Objective:**  
To create a simple calculator that performs addition, subtraction, multiplication, and division.

**Implementation Details:**

* Two input fields were created for the user to enter numbers.
* Operator buttons (+, −, ×, ÷) call the function setOp() to set the operation.
* The pressE() function retrieves input values, performs the calculation based on the selected operator, and displays the result.
* Basic input validation was added:
  + Displays a message if numbers are missing.
  + Prevents division by zero.

**Code Example (HTML + JavaScript):**

**Part 4. Mini Project — Interactive To-Do List**

**Task 8. Build a To-Do App**

**Objective:**  
To create an interactive **To-Do List application** using JavaScript and DOM manipulation.  
The app allows users to:

* Add new tasks with a title and description.
* Delete tasks from the list.
* Mark tasks as completed (with a strike-through effect).
* Store tasks temporarily in an array (no database or local storage required).

**Implementation Details**

**1. HTML Structure**

* A responsive layout was created using **Bootstrap**.
* The interface contains two main sections:
  + **To-Do List** — displays all created tasks.
  + **New Task Form** — contains input fields for the task title and description, along with an **“Add Task”** button.
* Each task includes two buttons:
  + **Delete** (to remove a task).
  + **Complete** (to mark it as finished with a strike-through).

**Code Example (HTML):**



**2. JavaScript Functionality**

* An empty array task is defined to hold all created tasks temporarily.
* When the **“Add Task”** button is clicked, the addNew() function:
  + Reads values from input fields.
  + Creates a new <li> element dynamically.
  + Inserts task title, description, and two buttons — **Delete** and **Complete**.
  + Appends the new task to the <ul> list.
  + Clears the input fields after submission.
* The **removeTask()** function deletes a specific list item (<li>) from the DOM.
* The **acceptTask()** function visually marks the selected task as completed by applying a **line-through** text decoration to both the title and description.

**Code Example (JavaScript):**



