Assessment:	**Web Programming 27(8)1 ST**
Duration:	90 minutes
Total	50

Assessors:

WEB PROGRAMMING 27(8)1 SUMMATIVE TEST

INSTRUCTIONS: GENERAL

- Five minutes have been added for you to read and understand these instructions. You should use this time to ask for clarification from the invigilator(s).
- Time yourself using a watch or your system clock. The start time is calculated from the assessment start time.
- No instructions or directives of the invigilator shall be disregarded. If any of the instructions are disobeyed, candidates shall expose themselves to disqualification from the current and future assessments.
- Invigilation is strictly on HybeFlex. Keep the chat clear of messages; send a direct message to the invigilator if you have queries. Do not use MS Teams, Email, or Moodle. assessmentQ will be used as a secondary messaging tool by the invigilators.
- Any updates to the assessment will be posted on HybeFlex and the shared message board. Keep checking these platforms. Additionally, you must monitor your assessmentQ messages.
- No explanation of the assessment questions shall be given, but you are allowed to ask whether a question is valid. It is possible that there might be an issue with a question e.g., repeated options in a Multiple-Choice question.

INSTRUCTIONS: SUBMITTING YOUR TEST

- Uploading: This assessment must be submitted on AssessmentQ only.
- Naming conventions: When uploading work, ensure naming conventions are followed as instructed. See the saving instructions at the end of this assessment if you are not sure.
- Zipping your work: At the end of this assessment, you should only upload a single zip file containing all your work. See the saving instructions at the end of this assessment if you are not sure.
- File formats: Any work that is uploaded must be in zip format ONLY. Other file types may give problems!
- Emailing (In case of difficulties during submission): Do not use MS Teams or cloud storage for submissions regardless of the reason. In case of technical difficulties, submit your assessment to your lecturer with a proper subject line that follows the naming convention WPR27(8)1 ST: StudentSurname StudentName. Your zip file should also follow the same naming convention.
- Reporting technical issues: Any technical issue related to AssessmentQ and HybeFlex must be reported to the invigilator immediately. You may be asked to provide proof of the error, and to follow reporting procedures that may include using the inbuilt reporting tool in HybeFlex.

INSTRUCTIONS: INTEGRITY AND LIABILITY

- Integrity: Observe the Honor Code1. Each candidate is expected to maintain academic integrity. Any form of plagiarism (including self-plagiarism) or cheating that is detected can lead to disciplinary measures being taken.

- Liability: It is your responsibility to ensure that you have saved your assessment correctly! We will not accept excuses and explanations after the assessment.
- Late submissions: Late assessments will not be admitted for marking.

INSTRUCTIONS: FORMAT OF THE ASSESSMENT

- This assessment does not include any theory.
- Marks may be reserved for best practices including commenting the important parts of your code etc. and error handling.
- Organise your code well as per the instructions.
- The allocated time includes time for uploading/ saving your solution.
- The mark allocation is subject to change.
- Penalties may be applied for not following the instructions.

Assessment:	**Web Programming 27(8)1 ST**
Duration:	90 minutes
Total	50

ANSWER THE FOLLOWING QUESTIONS:

PART 1: LOGIC IMPLEMENTATION (JS ONLY)

You have been provided with an array of objects. See the addenda. Using JS only (No HTML), complete the following tasks:

1. Create a Function to Add a Vehicle [3]:
    - Write a function called addVehicle(make, model, year, type) that takes in the vehicle's details as parameters and adds a new vehicle object to the vehicles array.
2. Create a Function to Find the Total Number of Vehicles [3]:
    - Write a function called getTotalVehicles() that returns the total number of vehicles in the vehicles array.
3. Create a Function to Calculate the Average Year [3]:
    - Write a function called getAverageYear() that calculates and returns the average year of all the vehicles in the vehicles array. Use built-in array methods where necessary.

SCORING CRITERIA PER PART:

- 3 Points: The function is correctly implemented, and the code works flawlessly.
- 2 Points: The function is mostly correct, with minor errors or omissions.
- 1 Point: Attempted, but significant errors in logic or missing parts.
- 0 Points: No attempt or completely incorrect implementation.

PART 2: HTML FORM INTEGRATION

You have been provided with a basic form for input. There are some errors in the form. Complete the following tasks. Remember that missing code/ syntax is also considered an error:

1. Find errors that are in the form [3].
2. Link the HTML to the JavaScript from Part 1 [0]
3. Ensure that the buttons (Add Vehicle, Show Total Vehicles, Show Average Age) are working with the JS. [3]

SCORING CRITERIA NO 1:

- 3 Points: No errors are in the error.
- 2 Points: Not more than 1 error.
- 1 Point: Fewer than 3 errors fixed
- 0 Points: No attempt.

SCORING CRITERIA FOR NO. 3:

- 3 Points: Form is correctly integrated with JavaScript, and all functionalities work as expected.
- 2 Points: Form is mostly functional with minor issues or missing features,
- 1 Point: Attempted but with significant errors or incomplete functionality.
- 0 Points: No attempt or completely incorrect implementation.

PART 3: FORM FEEDBACK MECHANISMS

Implement Input Validation:

- Ensure that all fields are filled out before submission.
- If a required field is empty, change its border color to red, and green otherwise.

SCORING CRITERIA:

- 3 Points: All feedback mechanisms are correctly implemented and provide clear feedback. The form is visually responsive to user input.
- 2 Points: Feedback mechanisms are mostly correct, with minor issues or omissions. The form provides some visual feedback.
- 1 Point: Attempted but with significant errors or incomplete functionality. Minimal or no visual feedback.
- 0 Points: No attempt or completely incorrect implementation.

Please note that you must write your own validation, and not rely on the input's type or the required attribute.

PART 4: NEW ARRAY METHODS: TOTAL VEHICLES BY TYPE CALCULATION

Instructions:

1. Write a Function to Calculate and Display Total Vehicles by Type:
   - Write a function called calculateTotalVehiclesByType(type) that filters the vehicles array by type and returns the total number of vehicles for that type.
2. Display the Results:

- Display the total vehicles for each type in the HTML.

SCORING CRITERIA PER PART:

- 3 Points: Total vehicles by type are correctly calculated and displayed using filter() and reduce(). The output is clearly presented.
- 2 Points: Calculation is mostly correct, with minor issues or omissions. The output is displayed with some clarity.
- 1 Point: Attempted but with significant errors or incomplete functionality. The output is minimal or unclear.
- 0 Points: No attempt or completely incorrect implementation.

*****END OF TEST****