

**ICTICT449 Use version control systems in development environments**

**ICTPRG437 Build a user interface**

**ICTPRG440 Apply introductory programming skills in different languages**

**ICTPRG302 Apply introductory programming techniques**

Learner version

# Assessment task 2

JavaScript Portfolio

**Disclaimer:** This resource was developed by Holmesglen Institute of TAFE in 2024 for the State of Victoria (Department of Education and Training) for educational purposes, is © State of Victoria (Department of Education and Training) and is available under a Creative Commons Attribution Non-commercial 4.0 International [(CC BY-NC 4.0)](https://creativecommons.org/licenses/by-nc/4.0/) license.

**The license does not apply to:** Trademarks or branding, including the Victorian Government logo and any unattributed content supplied by third parties



Learner version

# Assessment task 2

JavaScript Portfolio

|  |  |
| --- | --- |
| Learner information | |
| **Learner name:** Van Minh Le | **Learner ID:** 100693330 |
|  | |

|  |  |
| --- | --- |
| Section A – Program/course details | |
| **Qualification code:** ICT40120 | **Qualification title:**  Certificate IV in Information Technology (Programming) |
| **Unit code:**  **CPRO1 - Javascript**  ICTICT449  ICTPRG437  ICTPRG440  ICTPRG302 | **Unit title:**  **JavaScript**  Use version control systems in development environments  Build a user interface  Apply introductory programming skills in different languages  Apply introductory programming techniques |
| **Pre/co-requisites:** N/A | |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Section B – Assessment task details | | | |
| **Assessment number: 2** of 2 | **Semester/year:** 1/2025 | | **Due date:** 6/06/2025 |
| **Duration of assessment:** 8 weeks | | **Assessment task results:** This assessment task will be marked as:  Ungraded result: satisfactory or not yet satisfactory | |
|  | | | |

|  |
| --- |
| Section C – Instructions to learners |
| Assessment task description:  This assessment has two parts:   * Part A – Knowledge questions * Part B – Portfolio section   For Part A, you are required to answer all questions in the boxes below each question. You should be writing at least 1 paragraph for your answers, unless the question specifies otherwise. Some questions have tables to fill out.  Part B is broken up into 6 different parts.   * Part 1: Data structures and algorithms – simple data types. * Part 2: Using basic data structures – objects * Part 3: String manipulation and files * Part 4: Data structures and algorithms – complex data types * Part 5: Build a graphical User Interface * Part 6: Version control in a team environment   While doing each part the following requirements and guidelines must be followed:  **Documentation Requirement**  You are required to fully comment the codes, you must   1. Comment all functions 2. Comment all variables   **Comply with Coding Guidelines**  You must follow the coding guidelines below when developing the program (Refer to [w3school JS style guide](https://www.w3schools.com/js/js_conventions.asp) for detailed explanation for each of the following rules)   1. Use camelCase for identifier names (variables and functions) 2. All names start with a letter 3. Always put spaces around operators (= + - \* /), and after commas 4. Always use 2 or 4 spaces for indentation of code blocks 5. Statement rules    1. Always end a simple statement with a semicolon    2. Put the opening bracket at the end of the first line.    3. Use one space before the opening bracket.    4. Put the closing bracket on a new line, without leading spaces.    5. Do not end a complex statement with a semicolon.   **Version Control**   * You are required to create a **minimum of 3 repositories** to store your code. * Each part has a section to record your repository for that part. * Suggested grouping is, parts 1 and 2, part 3, parts 4 and 5. * You are required to have at least 3 branches for each repository. * Each question has an area for you to write in your branch name and commit number. * You need to have 3 merge requests as well for each repository.   **Project setup**   * You need to create a root folder called “JSAT2” * Under this folder create the folders part1, part2, part3, part4, part4, part6.      * All work for each part should be placed inside these folders * You can create sub folders in each of these folders if you need to. I.e. one for wireframes for example. * You are required to submit all your code zipped up to Brightspace   For more detailed instructions for each part, refer to the supporting document.  {RTO to insert relevant policy and procedural information for submission of assessments} |
|  |

|  |
| --- |
| Section D – Conditions for assessment |
| ****Unit application:****   * This is an individual task. (except for Part 6 of this assessment) * You must meet all criteria listed in the marking guide to be satisfactory in this task. * You must submit all required working files, documentation, and any other assets that you feel may be required in a zipped file, including the completed and signed coversheet. The assessment must be completed and submitted electronically to Brightspace by the due date. If this is not possible, you must contact your assessor to gain written approval for an alternative arrangement for submitting the assessment. * If not successful within the enrolment period as per Holmesglen assessment procedure, you will be requested to resubmit within 7 days of receiving feedback. You will have the opportunity to resubmit if any part of the assessment is deemed unsatisfactory (you are permitted TWO (2) resubmission per assessment task). Resubmissions must be submitted by the resubmission due date provided by your teacher. * This task is open book. You may use the internet for research purposes only. All answers must be in your own words. Where a quote is used, you must cite the information source. * If you feel you require special allowance or adjustment to this task, please discuss with your assessor within one week of commencing this assessment. Any change to assessment arrangements must be reviewed by the Education Manager and approved by the Head of Department. * You can appeal an assessment decision according to the Holmesglen Assessment Complaints and Appeals Procedure. * You are expected to dedicate time to develop this assessment task both in and out of the classroom. * Leaner must contribute to and abide by organisational standards including intellectual property, privacy laws, and plagiarism and academic honesty. Further information is detailed at: https://holmesglen.edu.au/Students/Student-Resources/ |
|  |

|  |  |
| --- | --- |
| Equipment/resources learners must supply: | Equipment/resources to be provided by the RTO: |
| * **Learners opting to BYOD laptop or intending to learn remotely will require access to:** * A MAC or PC/laptop with the following minimum specification: * Quad Core CPU * • CPU with minimum 4GHz processor or faster * • RAM 16GB * • 200GB of storage * • Windows 10 OS, or virtual machine installed with Windows 10 OS (available free from https://developer.microsoft.com/en-us/windows/downloads/virtual-machines/ or https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise) * • Monitor 24" (PC only, dual monitor optional but preferred) * • Headset with microphone (webcam optional but preferred) * • Access to reliable internet connection (ADSL or cable connection desirable) * Applications are available through Holmesglen MyHorizon and some have free licences which can be downloaded via url below: * • Brightspace (Learning Management System) access - https://holmesglen.brightspace.com/ * • MyHorizon - https://myHorizon.holmesglen.edu.au - free to download * • 365 Microsoft office suite also can be downloaded via https://portal.office.com - free for all Holmesglen students * • LinkedIn Learning - free access * • WebEx - https://holmesglen.webex.com/ - free to download * • Unity - https://unity.com/ - student license free to download * • Maya - http://www.autodesk.com/education/free-software/maya - student license free to download * • Substance Painter https://substance3d.adobe.com/education/ - student license free to download * • Adobe Photoshop - https://www.adobe.com/au * • ZBrush – https://pixologic.com/ * • GitHub desktop - https://desktop.github.com/ - free to download * • 3D Coat - http:/3dcoat.com/download/ * • 7Zip or an equivalent compression utility - free to download * • Google Chrome – recommended web browser * • OneDrive - https://www.microsoft.com/en-ww/microsoft-365/onedrive/online-cloud-storage - free to downloade | **This is a blended learning course and as such a remote learner will access their own computer equipment as per the specification provided. The Institute will provide the following:**   * Holmesglen CAIT computer classroom: * data projector, whiteboard, computer with double screens, mobile Cisco Spark Board, Conference camera * HP Z1 Entry Tower G5 * • Dual Displays 24” HP Monitor * • CPU: Intel Core i7-9700K @ 4GHz * • Motherboard: 8591 * • RAM 32GB * • 1TB SSD * • OS: Windows 10 * This will vary as CAIT upgrades computer rooms. * • Internet connection * Applications available at ZENworks and Holmesglen MyHorizon and free to download via url below: * • Brightspace (Learning Management System) access - https://holmesglen.brightspace.com/ * • 365 Microsoft office suite also can be downloaded via https://portal.office.com - free to Holmesglen students * • LinkedIn Learning - free access * • WebEx - https://holmesglen.webex.com/ - free to download * • Unity - https://unity.com/ - student license free to donwload * • Maya - http://www.autodesk.com/education/free-software/maya - student license free to download * • Substance Painter https://substance3d.adobe.com/education/ - student license free to download * • Adobe Photoshop - https://www.adobe.com/au * • ZBrush – https://pixologic.com/ * • GitHub desktop - https://desktop.github.com/ - free to download * • 3D Coat - http:/3dcoat.com/download/ * • 7Zip or an equivalent compression utility - free to download * • Google Chrome – recommended web browser   • OneDrive - https://www.microsoft.com/en-ww/microsoft-365/onedrive/online-cloud-storage - free to download |

|  |  |  |  |
| --- | --- | --- | --- |
| Section E – Assessment questions/criteria | | | |
| **Assessment number:** 2 | | **Assessment title:** JavaScript Portfolio | |
| **Learner name:** Van Minh Le | | | **Learner ID:** 100693330 |
| **Unit code:**  **CPRO1 - Javascript**  ICTICT449  ICTPRG437  ICTPRG440  ICTPRG302 | **Unit title:**  JavaScript  Use version control systems in development environments  Build a user interface  Apply introductory programming skills in different languages  Apply introductory programming techniques | | |
| **Date:** Insert date |  | | |
|  | | | |

**PART A - KNOWLEDGE QUESTIONS (Written Response)**

**Provide your responses in the boxes below each question.**

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Compare GIT Hub and one other Version Control System. Look at the functionality of each system and how they work. Record your answer below.  Based on your research, which Version Control System would you pick and why? | Yes  No |
| Answer: I’m comparing GitHub and Subversion. Both platforms are Version Control Systems.  <compare>   |  |  | | --- | --- | | GitHub | Subversion (SVN) | | It is a **Distributed** Version Control System, meaning each developer has a full copy of the repository. | It is a **Centralized** Version Control System, mean a single central repository stored on a server. | | It provides a service to **work can be done offline**, commits stored locally | It **requires an internet connection** to commit changes | | It supports pull requests, forks, and **open-source contributions**. | It works primarily **with team-based collaboration** and **restricted access**. | | It **resolves conflicts** more **easily** through merging and is more efficient. | It is more **difficult** **to resolve** because changes need to be committed to the central server first. | | It is **inappropriate** for **large binary files**. | It is better **suited** for handling **large binary files.** | | It is **widely** used and has **strong open-source support**. | It is **mainly** used in **enterprises** with a **smaller** **community**. |   <Which VCS, and why?>  Based on above comparison, I would pick **GitHub**. Because GitHub more **flexible**, **supports** **offline** work. The ability to **easily branch, experiment, and merge** makes it ideal for modern software development, especially for teams. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Define distributed and centralised version control systems and discuss the differences between the two types | Yes  No |
| Answer:   |  |  |  | | --- | --- | --- | | **Feature** | **CVCS (Centralized)** | **DVCS (Distributed)** | | Repository | Single central server | Each user has a full copy | | Offline Work | Not possible | Fully supported | | Speed | Slower (network dependent) | Faster (local commits) | | Failure Risk | High (server crash affects all) | Low (copies exist on multiple devices) | | Branching | Complex and slow | Easy and fast | | Collaboration | Requires constant network | More flexible | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| **Describe the following concepts related to Version Control System (VCS)**  **• repositories and working copies**  **• merging and merge tools**  **• working and indiscriminative commits**  **• resolving conflict and backout changes** | Yes  No |
| Answer:  **Repositories and Working Copies**  Repositories: In a VCS, a repository is a central storage location where all versions of files and their historical changes are kept. It serves as the core database where code and documents are stored and managed.  Working Copies: A working copy is a local version of the repository that a user has on their computer. This local copy can be edited and modified before changes are committed back to the central repository.  **Merging and Merge Tools**  Merging: This is the process of integrating changes from different branches of a repository into a single branch. Merging helps synchronize changes made by different team members or across different features of the project.  Merge Tools: Merge tools are specialized software that help resolve conflicts when merging changes. They provide a visual interface to compare changes side by side, highlight differences, and allow users to choose which changes to keep.  **Working and Indiscriminative Commits**  Working Commits: These are commits that are intended to reflect meaningful progress in the project. They often include specific, well-described changes and are made when a particular feature or bug fix is completed.  Indiscriminative Commits: These commits are often less organized and may include unrelated changes. They might be made hastily or without proper documentation, making it harder to track the purpose and context of the changes. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| **Describe principles and techniques of creating repositories and branch workflows. (Hint: Look at the workflow in Part 6)** | Yes  No |
| Answer:  Creating repositories and branch workflows | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe version control industry standard best practices. | Yes  No |
| Answer: | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe distributed version control industry standard best practices. | Yes  No |
| Answer: | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe workflow processes applicable to using version control systems. You may describe the workflow studied in class. (Hint: look at part6) | Yes  No |
| Answer: | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| **Discuss a range of strategies when developing a user interface with HTML and CSS. Finish the table below.** | Yes  No |
| Answer:   |  |  | | --- | --- | | **Requirement** | **Strategies** | | Introduce a 20 px gap between two vertically aligned element. |  | | Place an element in the centre of its container. |  | | Apply certain colour to all text-based element in a container |  | | Display information in a table manner |  | | Set colour using RGB values and HEX values |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| HTML + CSS + JavaScript languages are used as a combination to program web-based user interface. Discuss the role / functions of each of the languages in building web-based UI. | Yes  No |
| Answer:  HTML  CSS  JavaScript | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe small-size web-based UI development / application process | Yes  No |
| Answer:  Copy here. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Identify application development languages that can be used to build the following application. And briefly describe each of them. | Yes  No |
| Answer:   |  |  |  | | --- | --- | --- | | **Application Type** | **Language** | **Description** | | **Desktop Window Application** |  |  | | **Web Application** |  |  | | **Mobile Application** |  |  | | **Smart TV Application** |  |  | | **Smart Watch Application** |  |  |   . | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| **Describe what is UI (User Interface) prototyping and identify at least two prototyping tools.** | Yes  No |
| Answer:  What is UI (User Interface) Prototyping?   |  |  | | --- | --- | | **Tools Name** | **Website** | |  |  | |  |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| What is UI style guide, and what should a UI style guide contain (with 50+ words)? | Yes  No |
| Answer:  What is UI style guide?  What should a UI style guide contain? | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Use example to explain the following JavaScript syntaxes. | Yes  No |
| Answer:   |  |  | | --- | --- | | **Syntax** | **Example** | | **Define variable** |  | | **Condition structure** |  | | **Loop structure** |  | | **Define a function** |  | | **Define a class** |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe organisational procedures and guidelines that can be used for the following languages. | Yes  No |
| Answer:   |  |  | | --- | --- | | **Language** | **Organisational procedures and guideline examples** | | **JavaScript** |  | | **HTML** |  | | **CSS** |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe at least 3 JavaScript shorthand technique with examples | Yes  No |
| Answer:   |  |  |  | | --- | --- | --- | | **Technique name** | **Description** | **Example** | |  |  |  | |  |  |  | |  |  |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Identify the official standards of the following languages | Yes  No |
| Answer:   |  |  |  |  | | --- | --- | --- | --- | | **Language Name** | **Standard name** | **Description** | **Official link** | | JavaScript |  |  |  | | HTML |  |  |  | | CSS |  |  |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe code conversions that are used to comment JavaScript codes | Yes  No |
| Answer:  Copy here. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| **Describe the following methodologies when programming with HTML/JavaScript/CSS languages.** | Yes  No |
| Answer:   |  |  | | --- | --- | | **Methodology** | **Description** | | Development process |  | | Debugging process |  | | Testing techniques |  | | Using data structures |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Give two example of web-based DevOps lifecycle tools. Describe the tools and identify their official websites. | Yes  No |
| Answer:   |  |  |  | | --- | --- | --- | | **Tool name** | **Description** | **URL** | |  |  |  | |  |  |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Give two example of DevOps automation tools. Describe the tools and identify their official websites | Yes  No |
| Answer:   |  |  |  | | --- | --- | --- | | **Tool name** | **Description** | **URL** | |  |  |  | |  |  |  | | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| List at explain the following concepts for JavaScript   * Datatypes – list and explain at least 3 different datatypes * Operators – list and explain 3 different operators * Expression – explain what an expression is. You may use an example * Variables – list and explain two different ways to create variables | Yes  No |
| Answer:  Datatypes:  Operators:  Expression  Variables: | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| List and explain the following concepts for JavaScript using examples   * What is a sequence * What is selection * What is an iteration construct | Yes  No |
| Answer:  Copy here. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Explain JavaScript debugging techniques. Using an example explain how you would find an error in your code by debugging. | Yes  No |
| Answer:  Copy here. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Explain how you would test a JavaScript application. List the steps you would follow.  Talk about what methods you would use to test your code. | Yes  No |
| Answer:  Copy here. | Comment: |

|  |  |
| --- | --- |
| Question | Satisfactory response |
| Describe how you would document your programming activities. List what documents are created and what their purpose is. (Hint: you may want to look at part 5) | Yes  No |
| Answer:  Copy here. | Comment: |

**PART B - PORTFOLIO**

|  |  |  |
| --- | --- | --- |
| Criteria for assessment | SatisfactoryYes No | Comment |
| **The following has been submitted for assessment:** | | |
| Part 1 |  |  |
| 1A Learner has listed repository details |  |  |
| 1.1 Learner has built the array to specifications |  |  |
| 1.2 Learner has sorted the array |  |  |
| 1.3 Learner has inserted the required numbers into the array and kept the array in ascending order |  |  |
| 1.4 Learner has removed the required numbers from the array and kept the array in ascending order. |  |  |
| 1.5 Learner has implemented sequential search algorithm |  |  |
| 1.6 Learner has implemented binary search algorithm |  |  |
| 1.7 Learner has developed a test plan for parts 1.5 and 1.6 |  |  |
| 1.8 Learner has conducted and recorded the results of the tests created in 1.7 |  |  |
| 1.9 Learner has documented the JavaScript code file as per requirements and organisational requirements |  |  |
| 1.10 Learner has used debugging tools to set a break point, step through code and look at variable contents |  |  |
| 1.11 Learner has followed all guidelines and organisational requirements. |  |  |
| Part 2 |  |  |
| 2A Learner has listed repository details |  |  |
| 2.1 Learner has built the object to specifications |  |  |
| 2.2 Learner has added the new properties to the object |  |  |
| 2.3 Learner has modified the required properties on the object |  |  |
| 2.4 Learn has removed the required property on the object |  |  |
| Part 3 |  |  |
| 3A Learner has listed repository details |  |  |
| 3.1 Learner has created the required strings |  |  |
| 3.2 Learner has used length and charAt to get the required information |  |  |
| 3.3 Learner has used slice and substring |  |  |
| 3.4 Learner has used toUpperCase and toLowerCase |  |  |
| 3.5 Learner has used concat, trim, replace and split as required |  |  |
| 3.6 Learner has created algorithms to save to a text file and read from a text file |  |  |
| 3.7 Learner has written a string to a text file |  |  |
| 3.8 Learner has read text from a text file |  |  |
| Part 4 |  |  |
| 4A Learner has listed repository details |  |  |
| 4.1 Learner has implemented the Movie class |  |  |
| 4.2 Learner has created an array of 10 movies, unordered |  |  |
| 4.3 Learner has sorted the array by Movie ID |  |  |
| 4.4 Learner has implement either sequential or binary search algorithm |  |  |
| Part 5 |  |  |
| 5.1 Learner has listed repository details and initial commit |  |  |
| 5.2 Learner has gathered and understood the UI requirements. The learner has communicated with the client about the requirements and get a sign off. |  |  |
| 5.3 Learner has listed the HTML guidelines the are following and described them |  |  |
| 5.4 Learner has identified the software applications to be used to create the wireframe and UI prototype |  |  |
| 5.5 Learner has created the UI wireframe using the tools listed in 5.4 |  |  |
| 5.6 Learner has met the client to get feedback on the wireframe, made changes due to feedback and gotten client sign off to the changes made. |  |  |
| 5.7 Learner has created a theme for the UI |  |  |
| 5.8 Learner created a UI prototype using the tools listed in 5.4. The learner enhanced the wireframe with the theme colours defined in 5.7 |  |  |
| 5.9 Learner has gotten feedback and made changes to the prototype, then gotten the client sign off. |  |  |
| 5.10 Learner has described the content flow, UI actions and UI events for the application |  |  |
| 5.11 Learner has determined the languages to be used to build the UI. The learner has described the languages and their purpose in building the UI |  |  |
| 5.12 The Learner has built the UI with the languages listed in part 5.11 and to specifications. |  |  |
| 5.13 The learner had developed a test plan to test the UI functionality |  |  |
| 5.14 The learner has conducted the tests in the test plan, recorded the results and made changes where needed. |  |  |
| 5.15 The learner has demonstrated the UI to the client and obtained the final sign off. |  |  |
| Part 6 |  |  |
| 6.1 Learner has created the GitHub account and communicated any difficulties or disruptions while completing this task |  |  |
| 6.2 Learner has cloned the required repository, created their team directory and required files. The learner has recorded the merge request, branch name and commit number |  |  |
| 6.3 The Learner has received and recorded feedback from the teacher about their page. |  |  |
| 6.4 The Learner has actioned changes from feedback and recorded the merge request, branch name and commit number |  |  |
| 6.5 The learner has updated the myscript.js file to add in their details. |  |  |
| 6.6 The learner has obtained the final sign off. |  |  |

|  |
| --- |
| Section F – Feedback to Learner |
| **Has the learner successfully completed this assessment task?** Yes No |
| **Assessor feedback (as appropriate):** Insert feedback |
| **Resubmission allowed:** Yes No  **Resubmission due date:** Insert date |
| **Assessor name:** Insert name |
| **Assessor signature:** |
| Learner acknowledgement: |
| **Learner Signature:** |
| **Date:** Insert date |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Part 1: Data structure and algorithm – simple data type** | | | |
| **Programming Requirement**  In this part, you are required to design and implement a range of algorithms to operate an array with given input. The following operations need to be done using JavaScript programming language and the result need to be verified using browser.   1. Build an array with the following numbers.    * 11, 5, 8, 3, 25, 16, 31, 45, 14, 20 2. Sort the array in ascending order (means to arrange them from smallest to largest) 3. Insert three numbers 19, 23, 30 into the sorted array in step 2, and keep the new array sorted in ascending order. You could decide where to insert. 4. Remove two numbers 8, 31 from the array generated in step 3, and keep the new array sorted in ascending order 5. Create a JavaScript function called “sequentialSearch”    * This function accepts two parameters: an array and a value    * This function searches the input array for the input value using sequential search    * You cannot use the JavaScript build-in “find” function. You must implement the algorithm with you own code.    * This function returns the index of the target value if found and or -1 if not found 6. Create a JavaScript function called “binarySearch”    * This function accepts two parameters: an array and a value    * This function searches the input array for the input value using binary search    * You cannot use the JavaScript build-in “find” function. You must implement the algorithm with you own code.    * This function returns the index of the target array element if found or returns a -1 if not found   **Documentation Requirement**  You are required to fully comment the codes, you must   1. Comment all functions 2. Comment all variables   **Comply with Coding Guidelines**  You must follow the coding guidelines below when developing the program (Refer to [w3school JS style guide](https://www.w3schools.com/js/js_conventions.asp) for detailed explanation for each of the following rules)   1. Use camelCase for identifier names (variables and functions) 2. All names start with a letter 3. Always put spaces around operators (= + - \* /), and after commas 4. Always use 2 or 4 spaces for indentation of code blocks 5. Statement rules    1. Always end a simple statement with a semicolon    2. Put the opening bracket at the end of the first line.    3. Use one space before the opening bracket.    4. Put the closing bracket on a new line, without leading spaces.    5. Do not end a complex statement with a semicolon.   **Debugging**  You are required to use the developer tool in modern web browser (e.g. google chrome) to debug the code. You are required to demonstrate the following skills   * Setup break point to pause the execution * Use code step through functions to step-by-step trace the code execution * Examine variable contents in runtime   **Testing**   * You are required to develop a test plan for this programming task, using required test plan template. * The test case shall have a full coverage of function “sequentialSearch” and “binarySearch” * You are required to test your code according to test plan and record the test result. * If any of the test cases failed, you are required to debug the codes, fix the issue and test again, until all test cases pass.   **Version Control**   * You are required to create a repository to store your code. * You are required to have at least 3 branches for this part. * Each question has an allowance for you to write in your branch name and commit number. * You need to have 3 merge requests as well.   **Submit for review**  You are required to submit the following for review   1. This assessment document with all questions answered 2. All source code files 3. Test plan & test result   You are required to compress all files should in a zip file and submit the zip file for review. | | | |
| **Q1A Record GitHub repository details for this part:** | | | |
| **Record the GitHub repository name:**  **Record the GitHub repository URL:** | | | |
| **Q1.1 Build an array with the following numbers in JavaScript and print the array in the console: 11, 5, 8, 3, 25, 16, 31, 45, 14, 20** | | | |
| < Insert the screenshot JavaScript code that built the array>  <Insert the screenshot of the array printed in the console >  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.2 Sort the array created in Q1.1 above in “ascending” order (ascending is smallest to largest)** | | | |
| < Insert the screenshot JavaScript code that sorts the array>  <Insert the screenshot of the sorted array printed in the console>  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.3 Insert three numbers 19, 23, 30 into the sorted array above, and keep the new array in ascending order** | | | |
| < Insert the screenshot JavaScript code that inserts the new numbers>  (Note: remember that you can choose where you place the numbers or sort the array again.)  <Insert the screenshot of the new array printed in the console >  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.4 Remove two numbers 8, 31 from the array generated in Q1.3, and keep the new array in ascending order.** | | | |
| < Insert the screenshot JavaScript code that removes the numbers from the array>  (Note: remember you can choose what to delete)  <Insert the screenshot of the array printed in the console >  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.5 Implement the function “seqentialSearch” as per part 5 of the task description** | | | |
| <Insert a screenshot of the JavaScript code of the function “sequentialSearch”>  <Insert a screenshot of the console when the target element is found>  <Insert a screenshot of the console when the target element was not found>  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.6 Implement the function “binarySearch” as per part 6 of the task description** | | | |
| <Insert a screenshot of the JavaScript code of the function “binarySearch”>  <Insert a screenshot of the console when the target element is found>  <Insert a screenshot of the console when the target element was not found>  Is your result correct?  Yes / No.  (If No, fix your code and correct the result) | | | |
| Branch name |  | Commit Number: |  |
| **Q1.7 You are required to develop a test plan that covers the functions “sequentialSearch” and “binarySearch” The test plan should be in the submission for review.** | | | |
| Your test plan will need to test the functionality of these two functions.  <Record the name of your test plan file:> | | | |
| Branch name |  | Commit Number: |  |
| **Q1.8 Conduct the tests in your test plan and record the results in your test plan file.** | | | |
| Record the results of your test plans in your test plan document.  <Record the name of your test plan file:> | | | |
| Branch name |  | Commit Number: |  |
| **Q1.9 You are required to properly document your JavaScript code. The JavaScript code files must be submitted for review.** | | | |
| <Record the name of the JavaScript code files for this part:> | | | |
| Branch name |  | Commit Number: |  |
| **Q1.10 While development of the functions in Q1.5 and Q1.6 use the debugging tools to show the following:** | | | |
| <A screenshot of a break point set up>  <Screenshots of a step-by-step flow of one of the functions>  <A screenshot of variable contents> | | | |
| Branch name |  | Commit Number: |  |
| **Q1.11 Confirm that all coding guidelines have been followed** | | | |
| Yes / No | | | |
| Branch name |  | Commit Number: |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Part 2 – Using basic data structures – Objects** | | | |
| **Programming Requirement**  In this part, you are required to design and implement a range of algorithms to operate an object with given input. The following operations need to be done using JavaScript programming language and the result need to be verified using browser.   1. Build an object called “myMovie” with the following properties and data.    * Title: The longest day    * Year: 1964 2. Add in the properties:    * rating: 5    * Summary: World War II movie about the Normandy landings 3. Change the rating to 4 and change the year to 1962 4. Remove the Summary property   As per the last part, each question will have a space for you to record a branch name and commit code. | | | |
| **Q2A Record GitHub repository details for this part:**  **(Note: This repository can be the same as part1)** | | | |
| **Record the GitHub repository name:**  **Record the GitHub repository URL:** | | | |
| **Q2.1 Build an object called “myMovie” with the following properties and data:**   * + **Title: The longest day**   + **Year: 1964** | | | |
| <Insert the screenshot JavaScript code that built the object>  <Insert the screenshot of the object printed in the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q2.2 Add the following properties and data to the “myMovie” object:**   * + **rating: 5**   + **Summary: World War II movie about the Normandy landings** | | | |
| < Insert the screenshot JavaScript code that inserted new properties in the object>  < Insert the screenshot of the new object printed in the console >  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q2.3 Change the data of the properties listed below:**   * + **Rating: 4**   + **Year: 1962** | | | |
| < Insert the screenshot JavaScript code that updated the properties in the object >  <Insert the screenshot of the new object printed in the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q2.4 Remove the summary property from “myMovie”:**   * + **Title: The longest day**   + **Year: 1964** | | | |
| < Insert the screenshot JavaScript code that removed the property in the object >  < Insert the screenshot of the new object printed in the console >  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Part 3 – String Manipulation and Files** | | | |
| **Programming Requirement**  In this part, you are required to design and implement a range of algorithms to manipulate strings. The following operations need to be done using JavaScript programming language and the result need to be verified using node.  You will then need to create an algorithm and implement this to write a string to a text file and read in a text file. Reading and writing to file can be done in either C# or JavaScript. If using JavaScript, you will need to verify the results using Node.js.  As per the pervious parts, there are spaces for you to record your version control. The repository, branch name and commit code.  You should start a new repository for this part.   1. Define the following strings:    * myString = “This is a string”    * anotherString = “ Another string” (note there are 3 blank spaces at the beginning)    * hello = “Hello there!    * myName = your first name 2. Use the utility functions length and chat to find out information    * Find out the length of myString    * Find out the first character of myString    * Find out the 11th character of myString 3. Use the slice and substring functions to create new strings.    * Slice “is a” from myString    * Use substring to get “the” from anotherString 4. Change the case of the myName string    * Print out the myName string in all upper case    * Print out the myName string in all lower case 5. Use concat, trim, replace and split to create new strings    * Concat hello and myName together    * Use the trim function to remove the spaces in anotherString    * Use the replace function to change “is a” to an empty space in myString    * Split myString on the spaces. 6. Create algorithms    * Create algorithm for saving string to a text file    * Create algorithm for reading in the text file 7. Write a text file    * Name the text file output.txt    * Create a string of data to save to a text file    * Write the string to the text file 8. Read text from a text file    * Get the file name output.txt    * Read in the text file    * Prin the text that was read in, to the screen.   **Submit for review**  You are required to submit the following for review   1. This assessment document with all questions answered 2. All source code files   You are required to compress all files should in a zip file and submit the zip file for review. | | | |
| **Q3A Record GitHub repository details for this part:** | | | |
| **Record the GitHub repository name:**  **Record the GitHub repository URL:** | | | |
| **Q3.1 Define the following strings:**   * + **myString = “This is a string”**   + **anotherString = “ Another string” (note there are 3 blank spaces at the beginning)**   + **hello = “Hello there!**   + **myName = your first name** | | | |
| <Insert the screenshot JavaScript code that built the strings>  <Insert the screenshot of the strings printed in the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.2 Use the utility functions to find out the following information**   * + **Find out the length of myString**   + **Find out the first character of myString**   + **Find out the 11th character of myString** | | | |
| <Insert a screenshot of the JavaScript code for the length>  <Insert a screenshot of the JavaScript code for first character>  <Insert a screenshot of the JavaScript code for 11th character>  <Insert a screenshot of length in the console>  <Insert a screenshot of the for the first character shown the console>  <Insert a screenshot of the for the 11th character shown the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.3 Use the functions slice and substring to create new strings**   * + **Slice “is a” from myString**   + **Use substring to get “the” from “notherString** | | | |
| <Insert a screenshot of the JavaScript code for slice function>  <Insert a screenshot of the JavaScript code for substring function>  <Insert a screenshot of the slice shown in the console>  <Insert a screenshot of the substring shown the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.4 Change the case of the myName string**   * + **Change myName string to upper case**   + **Change myName string to lower case** | | | |
| <Insert a screenshot of the JavaScript code>  <Insert a screenshot of the console showing myName in upper case and in lower case>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.5 Use various utility methods to create new strings**   * + **Use the concat method to join the hello string and myName string**   + **Use the trim method to remove the spaces in anotherString**   + **Use the replace method to change “is a” to an empty space in myString**   + **Use the split method on myString using the spaces to break the string** | | | |
| <Insert a screenshot of the JavaScript code>  <Insert a screenshot of the console showing the new strings>  <concat>  <trim>  <replace>  <split>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.6 Create algorithms for file handling**   * + **Create an algorithm for saving a string to a text file**   + **Create an algorithm for reading in a text file** | | | |
| <Saving a text file algorithm>  <Reading a text file algorithm> | | | |
| Branch name |  | Commit Number: |  |
| **Q3.7 Write a text file**   * + **Name the text file output.txt**   + **Create a string to save to the text file**   + **Write the text file**   **Note: For this question you can use JavaScript or C#** | | | |
| <Insert a screenshot of the JavaScript code>  <Insert a screenshot showing the created text file>  <Insert a screenshot showing the contents of the text file>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q3.8 Read the contents of a text file**   * + **Read in the text file output.txt from Q3.7**   + **Read in the contents of the file**   + **Display what was read in from the text file to the screen**   **Note: For this question you can use JavaScript or C#** | | | |
| <Insert a screenshot of the JavaScript code>  <Insert a screenshot showing the console >  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Part 4 – Data structures and algorithms – complex data types** | | | |
| **Programming Requirement**  In this part, you are required to design and implement a range of algorithms to operate an array of objects. The following operations need to be done using JavaScript programming language and the result need to be verified using browser.  You will be required to create a GitHub repository for this part. As per the previous parts, branch names an commit numbers can be recorded in each question below.   1. Define a movie class with the following properties.    * Movie ID    * Title    * Year    * Rating 2. Create an array of at least 10 movies with mock values    * Movie ID must be unique for each movie.    * Movie ID must not be sorted in the array 3. Sort array by Movie ID in ascending order 4. Implement at least one search algorithm (sequential search or binary search)    * This function accepts two parameters: an array and a movie ID    * This function searches the input array for the input movie ID using selected searching algorithm    * You cannot use the JavaScript build-in “find” function. You must implement the algorithm with your own code.    * This function returns the movie object found or return “null” if not found   Note: Students may use <https://www.mockaroo.com/> to help generate mock data for array.  **Submit for review**  You are required to submit the following for review   1. This assessment document with all questions answered 2. All source code files   You are required to compress all files should in a zip file and submit the zip file for review. | | | |
| **Q4A Record GitHub repository details for this part:** | | | |
| **Record the GitHub repository name:**  **Record the GitHub repository URL:** | | | |
| **Q4.1 Define the movie class as required using JavaScript. Insert the screenshot of your code below** | | | |
| <Insert a screenshot of the JavaScript code that built the class>  <Insert a screenshot showing the movie class in the console > | | | |
| Branch name |  | Commit Number: |  |
| **Q4.2 Create an array of at least 10 movies with mock values and print the array in console.**  **Remember: Movie ID must be unique for each movie and not sorted in the array.** | | | |
| <Insert a screenshot of the JavaScript code that built the array>  <Insert a screenshot showing the array in the console >  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q4.3 Sort the array in “ascending” order by Movie ID.**  **Remember: Movie ID must be unique for each movie and not sorted in the array.** | | | |
| <Insert a screenshot of the JavaScript code that sorted the array>  <Insert a screenshot showing the sorted array in the console >  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |
| **Q4.4 Implement at least one search algorithm (either sequential search or binary search) according to the requirements in the task description.** | | | |
| Which search algorithm was chosen:  <Insert a screenshot of the JavaScript code of selected search function>  < Insert a screenshot of the search function when the target ID was found in the console >  < Insert a screenshot of the search function when the target ID was not found in the console>  Is the result correct?  Yes / No  (If no, fix your code till the result is correct) | | | |
| Branch name |  | Commit Number: |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Part 5 – Build a graphical user interface** | | | | | | | | | | | | |
| **UI Requirement**  You are required to create a JavaScript array to **maintain a list of the movie object defined in Part 4** and build a web-based UI to manage the movie list using HTML, CSS and JavaScript. This move list needs to be created as a class.  Also, you are required to create and maintain a repository for this project. You need to upload or push your changes after you complete each question. The repository name is completely up to you, but it must be public.  Remember to use branches and insert your commit numbers in each question  The web-based UI has three sections   1. Add movie section: this UI section allow user to insert one movie to the array. In this UI, the following elements are required    * **A Movie ID input**    * **A Title input**    * **A Year input**    * **A Rating input**    * **A Submit button** – when user click on the Submit button, a movie object will be created using the information get from the input UI above, and this object will be added to the movie JavaScript array 2. Display movie list: this UI section allow user to view the current movie list. In this UI, the following elements are required    * **A list view** – this view is used to display information stored in the movie JavaScript array    * **A Refresh button** – when user click on the Refresh button, all data in the list view will be refreshed with the current data stored in JavaScript 3. Search movie: this UI section allow user to search movie information using movie ID, the following element are required    * **A movie ID input** – user can input a movie ID for searching    * **A Search button** – when user click on Search button, the program will search the movie list using the input ID and return result    * **A search result view** – this view is used to display the search result. The movie details will be displayed if found or a “0 result” information will be displayed if not found.    * **Search by Title** – when this is used search the movie list for something that contains the search string typed in. The list should update with all names shown in the list. 4. Sort movie: this UI section allow user to sort the list in both a-z or z-a fashion and render to the screen,    * **Sort A-Z -** This button should show the movie list sorted by Title A-Z and displayed to the screen    * **Sort Z-A** – This button should show the movie list sorted by Title Z-A and displayed to the screen.    * **Best Movies** - This button should show the movie list sorted by Rating Z – A and displayed to the screen.   **Testing:**  You are required to test both the UI and your JavaScript.   * You need test that your UI shows everything as expected and that it behaves as expected * You also need to test that all your JavaScript functions work as expected.   For the UI defined above you need to style the layout with CSS for better user experience. Pick a theme (colours) to use and stick with it through the page(s).  **Workflow**  You are required to follow the workflow blow to finish the development.   1. Establish UI requirement 2. Build a prototype UI (wireframe or graphic design) 3. Design and implement the UI 4. Test the UI 5. Documentation 6. Obtain client sign-off   **Coding guidelines**  HTML & CSS - <https://www.w3schools.com/html/html5_syntax.asp>  JavaScript - <https://www.w3schools.com/js/js_conventions.asp>  **Mock Data**  Mockaroo: <https://www.mockaroo.com/>  **Submit for review**  You are required to submit the following for review   1. This assessment document with all questions answered 2. All source code files 3. Wireframes in .png or .jpg format | | | | | | | | | | | | |
| **Q5.1 Record GitHub repository details for this part:** | | | | | | | | | | | | |
| **Record the GitHub repository name:**  **Record the GitHub repository URL:**  **The initial commit number:** | | | | | | | | | | | | |
| **Q5.2 Gather the UI requirements from the description above and communicate with the client. Confirm the requirements with your client and obtain client sign off.** | | | | | | | | | | | | |
| List all requirements in the table below:   |  |  | | --- | --- | | **Requirement #** | **Requirement Details** | | 1 |  | | 2 |  | | 3 |  | | 4 |  | | 5 |  | | …… |  | | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Checklist and sign off** | | | | | | | | | | | | |
| **Skills to be observed during this task to the required standard.** Checklist (To be completed by the learner’s facilitator)The following tasks are to be completed in relation to the brief for this project. Each of the skills must be observed on at least one occasion. | | | | | | **Date 1** | | | | **Date 2** | | |
|  | | | |  | | |
| **Satisfactory** | | | | **Satisfactory** | | |
| **Yes** | | | **No** | **Yes** | | **No** |
| **Leaner has expressed UI requirements to the client** | | | | | |  | | |  |  | |  |
| **Learner has confirmed and understood the UI requirements** | | | | | |  | | |  |  | |  |
| **Learner has confirmed the task with the client** | | | | | |  | | |  |  | |  |
| **Learn has confirmed the specifications, standards and guidelines to be followed with the client.** | | | | | |  | | |  |  | |  |
| **Assessor Name** | | *Amberle Seidl* | **Assessor Signature** | | *Signature* | | **Date** | | | |  | | |
| **Q5.3 What are the HTML guideline documents that are to be followed? List the filename or URL of the guidelines. Describe the guidelines you are following (at least 1 paragraph)** | | | | | | | | | | | | |
| <HTML guideline / file name or URL>  <HTML guideline description> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.4 What are the software applications that are to be used to create the prototype UI** (wireframe and prototype) | | | | | | | | | | | | |
| <List tools to use here>  <List which tool you are using > | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.5 Build the wireframe of your UI using the tool you identified in question 5.4** | | | | | | | | | | | | |
| <insert the screenshot of your wireframe> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.6 Review the wireframe with the client and get feedback. Revise and update the wireframe from feedback given.** | | | | | | | | | | | | |
| <List feedback>  <Insert the screenshot of your updated wireframe> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.7 Create the theme for your UI. Record the colours and font below** | | | | | | | | | | | | |
| <Colour 1 >  <Colour 2>  <Colour 3>  <Font> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.8 Create prototype of the application using tools listed in Q5.4. Enhance the wireframe with the theme colours.** | | | | | | | | | | | | |
| <Insert screenshot of the prototype > | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.9 Review the prototype with the client and get feedback. Revise and update the wireframe from feedback given. Obtain signoff** | | | | | | | | | | | | |
| <List feedback>  <Insert the screenshot of your updated prototype> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Client Name** | *Amberle Seidl* | **Client Signature** | *Insert Signature* | **Date** | *Insert Date* |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Q5.10 Describe the content flow / UI Actions / UI Events of each of the UI sections required.**   * **Content flow: Describe the content flow from the user’s perspective** * **UI Actions: describe the UI actions based on the UI design from the user’s perspective** * **UI Events: describe the events triggered in HTML / JavaScript** | | | | | | | | | | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | **UI Section** | **Content flow** | **UI Actions** | **UI Events** | | Add a movie | User input movie id, title, rating and year in the web form, and then click the submit button to add the movie. |  |  | | Display all movies |  | Action 1: user click refresh button  Action 2: list view area display all movies |  | | Search by ID |  |  | Button onclick event is trigger upon clicking the search button | | | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.11 Determine the languages used to build the UI. List all the languages used and describe what they are used for. (at least 1 paragraph for each description.** | | | | | | | | | | | | | |
| |  |  | | --- | --- | | **Language Name** | **Description** | |  |  | |  |  | |  |  | | | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.12 Design and build the UI with the languages listed in Q5.11** | | | | | | | | | | | | |
| List all files and file paths for your UI  <Insert the screenshot of the UI as shown in a browser>  <Get feedback from the client and make changes. List changes made below>  (You may need multiple branches for this question) | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.13 Develop the test cases the verify the UI Functionality** | | | | | | | | | | | | |
| <Insert the name of your test plan file> | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.14 Test the UI according to the test cases created in Q5.13. You must iterate your UI design and build, until the test results meet the requirements.** | | | | | | | | | | | | |
| Record the results of the tests you outlined in 5.13  <Insert the name of your test plan file>  If any tests are not successful, make changes to the code and test again. Record the results of the updates made in your test plan file. | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Q5.15 You are required to demonstrate your application to the client and obtain final sign-off** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| Branch name |  | | | Commit Number: | | | |  | | | | | |
| **Checklist and sign off** | | | | | | | | | | | | |
| **Skills to be observed during this task to the required standard.** Checklist (To be completed by the learner’s facilitator)The following tasks are to be completed in relation to the brief for this project. Each of the skills must be observed on at least one occasion. | | | | | | **Date 1** | | | | **Date 2** | | |
|  | | | |  | | |
| **Satisfactory** | | | | **Satisfactory** | | |
| **Yes** | | | **No** | **Yes** | | **No** |
| **Leaner has used debugging techniques to detect and correct error** | | | | | |  | | |  |  | |  |
| **Learner has developed the application in accordance with client requirements and guidelines** | | | | | |  | | |  |  | |  |
|  | | | | | |  | | |  |  | |  |
| **Assessor Name** | | *Amberle Seidl* | **Assessor Signature** | | *Signature* | | **Date** | | | |  | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Part 6 – Version control in a team environment** | | | | | | | | |
| **Task Requirements**  You are required to contribute to a project and work in a team collaboration environment using GitHub. The project is a website. And the URL of the website is <https://github.com/Amberside/the-programming-team-202510>  The home page of the website has been done. Your task is to create your own page on this website.  Each grid in the home page links to the page of a team member. The link has been defined in the home page (e.g., /team/amberle/index.html), but the member’s page has not been created yet. You need to join the team to get your page created, tested locally and then merge your changes to the project.   * Your teacher’s page (Amberle Seidl) has been created. Click on Amberle’s name to check the detail. * Your page has not been created yet. Find and click on your name to check the detail, which is a 404 file does not exist error.   **1. Requirement of your page**  You shall create a sub-folder, the name of which is made of first name, under the “team” folder. In this folder, you shall create a file called “index.html”. Refer to Amberle’s page as an example.  A screen shot of a computer screen  Description automatically generated  You must include your name.  You need to include some information about yourself.  You can think of this as a portfolio piece about yourself. Include things like experience and projects worked on.  **2. VCS requirement**  You must use GitHub as the VCS system for this task.  You shall provide your GitHub username to your teacher to add you a collaborator of the project, so that you could submit code.  You will also document the use of GitHub and the repositories here as well. Fill out the table provided. You must create at least 3 repositories while completing assessment task 2. These repositories must have at least 3 branches and be updated / merged 3 times.  You are not able to directly push updates to the “main” branch of the repository, instead, a new branch must be created for each push. And a pull request must be created for review. Your submitted code shall be reviewed and approved before merging to the main branch  **3. Edit JavaScript & Feedback**  You will need to update your name in the JavaScript array so that your page can be seen.  Lastly you will need to get feedback and make changes to your page.  **4. Workflow & procedure**  Step 1. Visit <https://github.com/Amberside/the-programming-team-202510>  Step 2. Click on Amberle’s name (the first grid) to view Amberle’s page (page can be viewed)  Step 3. Find your name, click on your name to check your page (page not found)  Step 4. Provide your GitHub username to your teacher to be added to the repository  Step 5. After your teacher invited you to the repository, accept the invitation in your email.  Step 6. Clone the repository to your local environment  Step 7. Create a new branch  Step 8. In folder “team”, create a sub-folder and name it using your first name  Step 9. In the sub-folder created above, create a file named “index.html”.  Step 10. Develop the content  Step 11. Test the result – Your page shall be displayed after clicking your name from the home page  Step 12. You may ask your teacher to confirm the test result before committing the codes  Step 13. Commit the codes  Step 14. Push/publish the commit to the server  Step 15. In GitHub website, create a pull request then notify your teacher to review the codes  Step 16 Respond to feedback.  Step 17. If your code is approved, your teacher will merge the code. Then you could view the website again to confirm the final result.  Step 18. Update the JavaScript file  Step 19. Obtain final sign-off from a supervisor  **5, Upload your pages and this document to Brightpspace** | | | | | | | | |
| **Q6.1 GitHub Details and config** | | | | | | | | |
| <Record your GitHub username>  <record any difficulties with using GitHub>  <Let the teacher know if there are any disruptions while you are doing this part> | | | | | | | | |
| **Q6.2 Create your team page** | | | | | | | | |
| Clone <https://github.com/Amberside/the-programming-team-202510>  <insert a screenshot of GitHub desktop showing the cloned repository>  Add in your directory under the team directory  Add in your name, information and portfolio details  Send the merge request to the teacher.  Merge request number: | | | | | | | | |
| Branch name |  | | | Commit Number: | | |  | |
| **Q6.3 Get feedback from your teacher** | | | | | | | | |
| <Insert feedback from the teacher> | | | | | | | | |
| **Q6.4 Make changes to your page based on the feedback.** | | | | | | | | |
| Make changes and create a new merge request  Merge request number: | | | | | | | | |
| Branch name |  | | | Commit Number: | | |  | |
| **Q6.5 Edit myscipt.js to add your details.** | | | | | | | | |
| Edit the required JS file and create a merge request.  Merge request number: | | | | | | | | |
| Branch name |  | | | Commit Number: | | |  | |
| **Q6.6 Obtain final sign off.** | | | | | | | | |
| <Project has been signed off and approval given>  Yes / no | | | | | | | | |
| **Assessor Name** | | *Amberle Seidl* | **Assessor Signature** | | *Signature* | **Date** | |  |