|  |
| --- |
| Functional Requirements |
| The name of the application should be Astronomical Processing. |
| The array is of type integer. |
| The client can use a text box input to search the array. |
| The array has 24 elements to reflect the number of hours per day. |
| The sort method must be coded using the Bubble Sort algorithm. |
| There are buttons that can sort and search the data. |
| There is an input field (text box) so data can be deleted, added and edited. |
| The sort method must be coded using the Bubble Sort algorithm. |
| The search method must be coded using the Binary Search algorithm. |
| A single text box is provided for the search input. |
| The program must generate an error message if the text box is empty. |
| The program must generate an error message if the search is not successful. |
| The program must generate a message if the search is successful. |
| The program must be able to add, edit and delete data values. |
| The array is filled with random integers to simulate the data stream (numbers between 10 and 99) with a button click (as per marking guide) |
| Non-Functional Requirements |
| Error prevention - whenever possible, design systems so that potential errors are kept to a minimum. |
| Aesthetic and minimalist design - keep clutter to a minimum. |
| Help users recognize, diagnose, and recover from errors - use plain language |
| Consistency and standards - interface designers should ensure that both the graphic elements and terminology are maintained across similar platforms. |
| Integer display optimised for two digits |
| User Input for search options and Display Output of 24-hour data should be visible on the same screen |