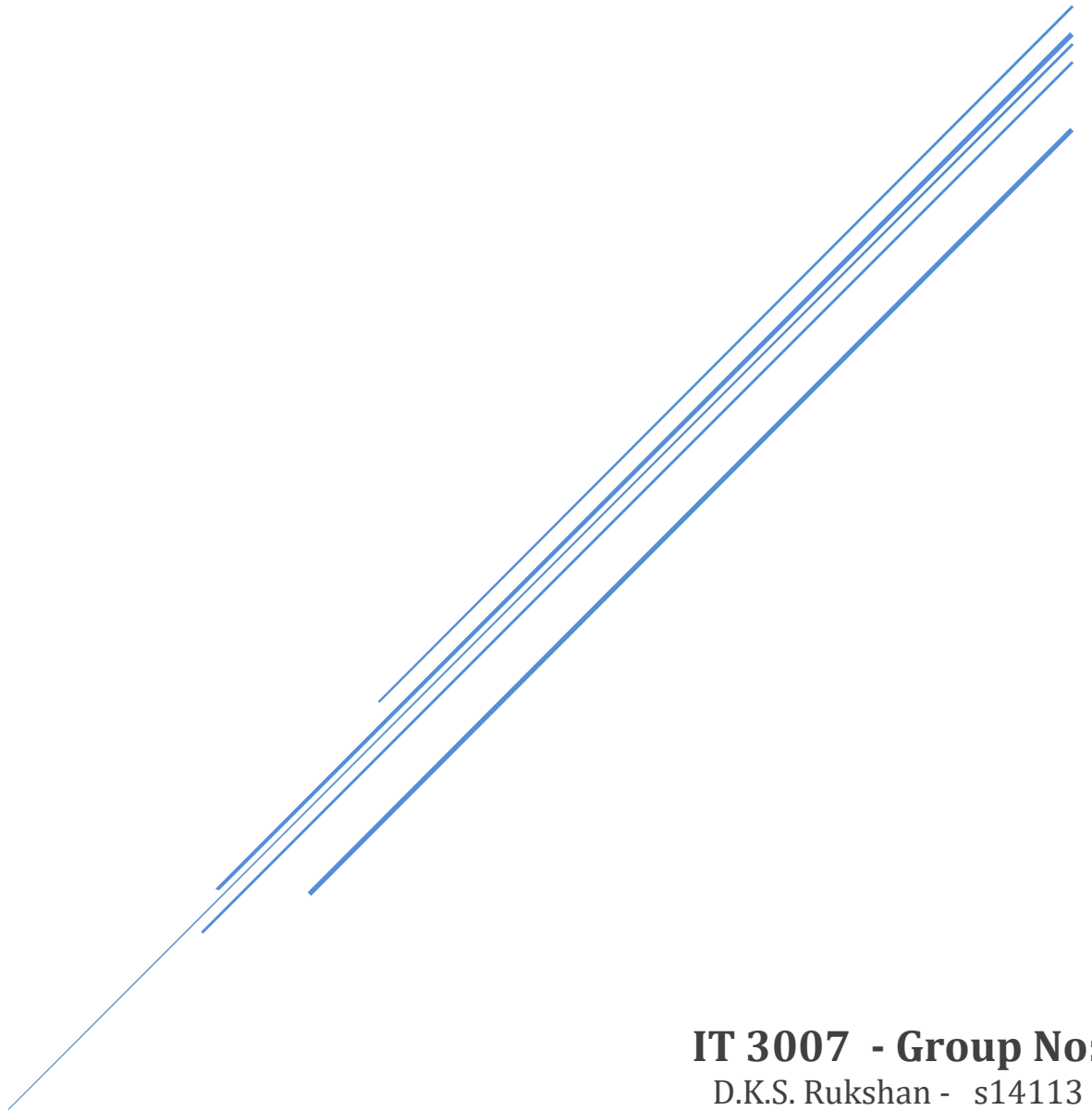


APPLICATION USING B-TREE, MERGE SORT & BINARY SEARCH

Documentation



IT 3007 - Group No: 7

D.K.S. Rukshan - s14113 (L)

A.G.K.M. Ariyaratne - s14043

H.G.S. Sarika - s14119

M.H.D. Maduraarachchi - s14089

L.L.S. Premathilaka - s14105

Application using B- tree, Merge sort and Binary Search

Here are brief explanations about the algorithms used in the application.

B - Tree :

In computer science, a B-tree is a self-balancing tree data structure that maintains sorted data and allows searches, sequential access, insertions, and deletions in logarithmic time. The B-tree generalizes the binary search tree, allowing for nodes with more than two children. Unlike other self-balancing binary search trees, the B-tree is well suited for storage systems that read and write relatively large blocks of data, such as disks. It is commonly used in databases and file systems. Height of the B -tree is $h = O(\log n)$ where n is the number of the keys stored in the tree

Algorithm	Average	Worst case
Search	$O(\log n)$	$O(\log n)$
Insert	$O(\log n)$	$O(\log n)$
Delete	$O(\log n)$	$O(\log n)$

Merge Sort :

Merge Sort is a sorting algorithm, which is commonly used in computer science. Merge Sort is a divide and conquer algorithm. It works by recursively breaking down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem. So, Merge Sort first divides the array into equal halves and then combines them in a sorted manner.

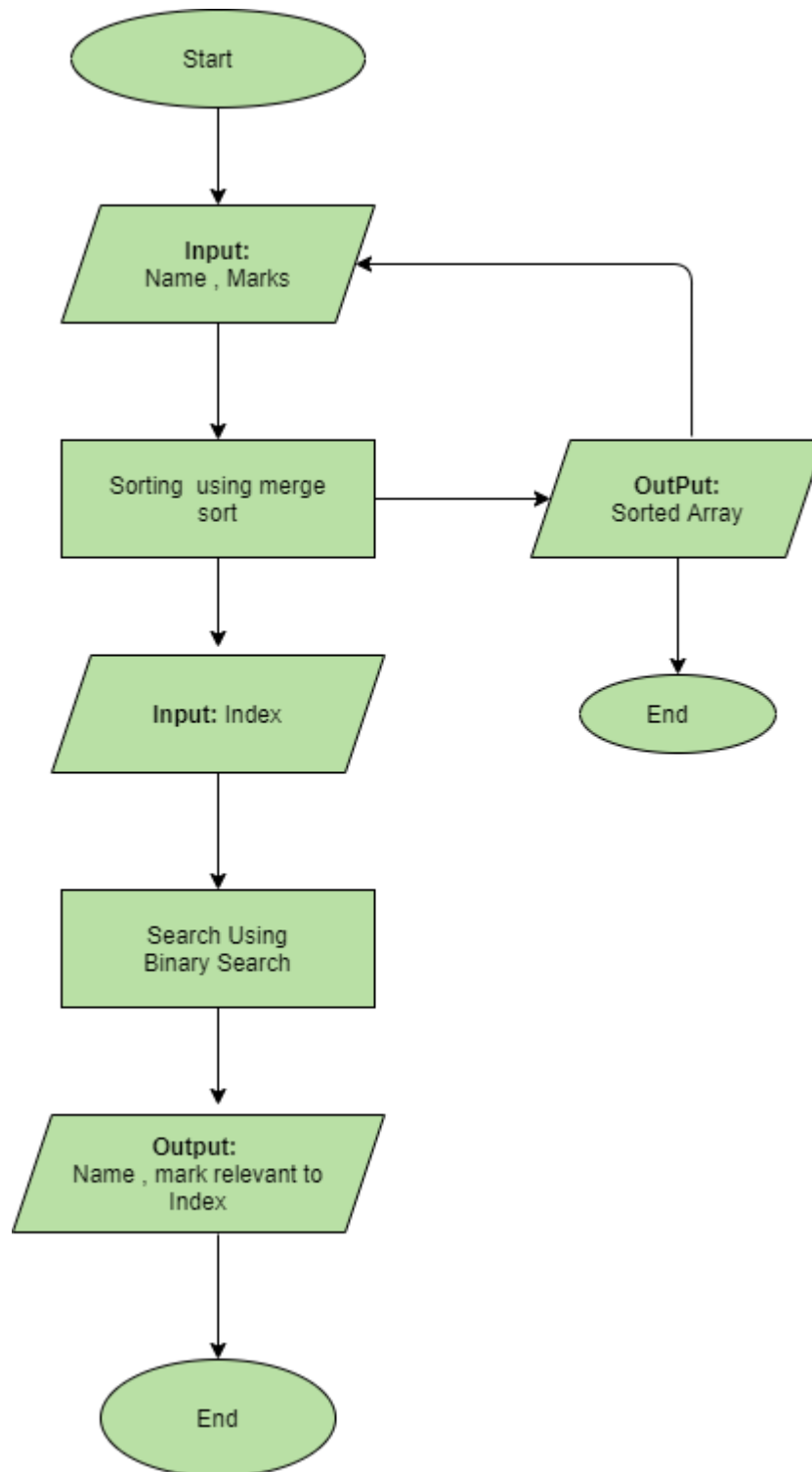
Overall time complexity	$O(n \log n)$
Insert	$O(\log n)$
Average time complexity	$O(n^2)$

Binary Search :

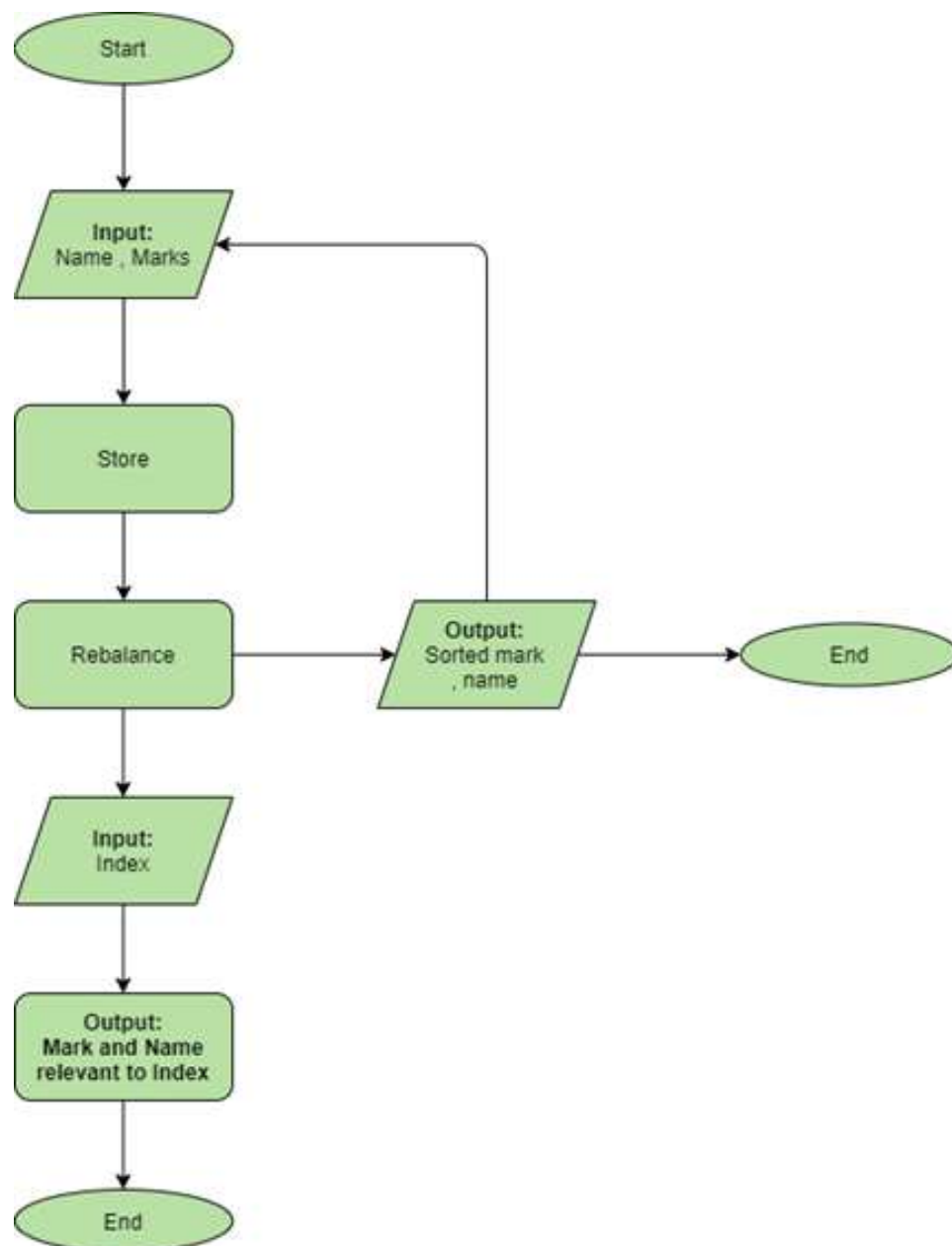
Binary search is a search algorithm that finds the position of a target value within a sorted array. This search algorithm works on the principle of divide and conquer. Binary search compares the target value to the middle element of the array. If they are not equal, the half in which the target cannot lie is eliminated and the search continues the remaining half, again taking the middle element to compare to the target value and repeating this until the target value is found. If the search ends with the remaining half being empty, the target is not in the array.

Algorithm	Average	Worst case
Search	$O(\log n)$	$O(n)$
Insert	$O(\log n)$	$O(n)$
Delete	$O(\log n)$	$O(n)$

Process of Merge sort & Binary search in the application



Process of B - Tree in the application



Application :

The program implemented contains a simple application that maintains a table of names and marks for a particular subject that is ranked according to the marks they have taken.

The screenshot shows a Java Swing application window titled "B-Tree, Merge Sort and Binary Search". The window is divided into two main sections, each with its own title bar: "B-Tree" on the left and "Merge Sort and Binary Search" on the right. Both sections contain a form with the following elements:

- Name:** A text input field. In the "B-Tree" section, it contains "Dilshan".
- Marks:** A text input field. In the "B-Tree" section, it contains "76".
- Insert:** A button to add a new record.
- Search Marks:** A text input field for searching by marks.
- Search:** A button to perform a search.
- Rank:** A text output field showing the result of a search.
- Reset:** A button to clear the form.
- Delete Marks:** A text input field for deleting a record by marks.
- Delete:** A button to delete a record.

The functionality of the B Tree application as follows.

- Insert the **name** of a person and their **marks** for a particular subject.
Enter Name and Marks then click insert

The screenshot shows a window titled "B-Tree, Merge Sort and Binary Search". It contains two panels. The left panel, labeled "B-Tree", has input fields for "Name" (containing "Dilshan") and "Marks" (containing "76"), an "Insert" button, a "Search Marks" field with a "Search" button, a "Rank" field, a "Reset" button, and a "Delete Marks" field with a "Delete" button. The right panel, labeled "Merge Sort and Binary Search", has identical input fields and buttons. A black line originates from the "Insert" button in the B-Tree panel and points downwards towards the second screenshot.

Name and marks according to rank will display under rank label

This screenshot shows the same application window after an insert operation. In the "B-Tree" panel, the "Rank" field now displays "1. Dilshan : 76". The "Insert" button is disabled. The "Merge Sort and Binary Search" panel remains unchanged. The black line from the first screenshot points to the "Rank" field in this screenshot.

- Once the value is inserted the table is updated and the name and marks field are added to the correct place of the rank table.

The screenshot shows a software interface with two main panels. The left panel, titled 'B-Tree', contains a form with 'Name' and 'Marks' input fields, an 'Insert' button, a 'Search Marks' input field, a 'Search' button, a 'Rank' table, a 'Reset' button, and a 'Delete Marks' input field with a 'Delete' button. The 'Rank' table currently lists: 1. Dilini : 90, 2. Nushan : 78, 3. Dilshan : 76, 4. Prasad : 72. The right panel, titled 'Merge Sort and Binary Search', has identical input fields and buttons but is currently empty.

Field adds to the correct rank

This screenshot shows the same interface after an insertion. In the 'B-Tree' panel, the 'Name' field is empty, the 'Marks' field contains '85', and the 'Insert' button is highlighted. The 'Search Marks' field is empty, and the 'Search' button is also highlighted. The 'Rank' table has been updated to: 1. Dilini : 90, 2. Hashan : 85, 3. Nushan : 78, 4. Dilshan : 76, 5. Prasad : 72. An arrow points from the 'Hashan : 85' entry in the table to the 'Search Marks' input field. The 'Merge Sort and Binary Search' panel remains unchanged and empty.

- To search the **marks** values and find a field for a given **marks** value. (Search by marks)

The screenshot shows a Java Swing window titled "B-Tree, Merge Sort and Binary Search". It contains two main panels. The left panel, titled "B-Tree", has input fields for "Name" and "Marks", an "Insert" button, a "Search Marks" field with the value "76", a "Search" button, a "Rank" label, a list box showing a ranked list of names and marks, a "Reset" button, and a "Delete Marks" section with an input field and a "Delete" button. The right panel, titled "Merge Sort and Binary Search", has similar input fields and buttons. An arrow points from the "Search" button in the left panel to the "Search Marks" field in the right panel.

Once search button clicked search result will appear under the **search** button

The screenshot shows the same Java Swing window as above, but with the search result "Dilshan : 76" displayed below the "Search" button in the left panel. An arrow points from the "Search" button to the result text.

- To delete a field from the table when given the **marks** value.

The image shows a software interface with two main panels. The left panel, titled 'B-Tree', contains a 'Name' and 'Marks' input section with an 'Insert' button. Below this is a 'Search Marks' section with a 'Search' button. A 'Rank' list displays the following data:

1.	Dilini	: 90
2.	Hashan	: 85
3.	Nushan	: 78
4.	Prasad	: 72

Below the rank list are a 'Reset' button and a 'Delete Marks' section with a text input field containing '90' and a 'Delete' button. The right panel, titled 'Merge Sort and Binary Search', has a similar layout but with a 'Rank' field instead of a list. An arrow points from the 'Delete Marks' field in the right panel to the 'Delete Marks' field in the left panel.

Type the mark value to be deleted. Click **Delete**. Marks will be deleted.

This screenshot shows the same interface after a deletion. In the 'B-Tree' panel, the 'Delete Marks' field is now empty, and the 'Rank' list has been updated to reflect the removal of Dilini (90):

1.	Hashan	: 85
2.	Nushan	: 78
3.	Prasad	: 72

The 'Delete Marks' field in the 'Merge Sort and Binary Search' panel remains empty.

The functionality of the Merge Sort and Binary Search application as follows.

- Insert the **name** of a person and their **marks** for a particular subject.
Enter Name and Marks then click insert

The screenshot shows the 'Merge Sort and Binary Search' application window. On the left is a 'B-Tree' panel with input fields for 'Name' and 'Marks', an 'Insert' button, a 'Search Marks' field with a 'Search' button, a 'Rank' field, a 'Reset' button, and a 'Delete Marks' field with a 'Delete' button. On the right is the main application panel. It has 'Name' and 'Marks' input fields with an 'Insert' button. Below these is a 'Search Marks' field with a 'Search' button. A 'Rank' section displays a list of names and marks: 1. Dilini : 93, 2. Nushan : 89, 3. Prasad : 75, 4. Hasith : 72. There are 'Reset' and 'Delete Marks' buttons at the bottom. An arrow points from the text 'Enter Name and Marks then click insert' to the 'Insert' button in the main panel.

Name and marks according to rank will display under **rank** label

This screenshot shows the same application window after an additional insertion. The 'Rank' list now contains five entries: 1. Dilini : 93, 2. Nushan : 89, 3. Dilshan : 80, 4. Prasad : 75, 5. Hasith : 72. The 'Name' and 'Marks' input fields in the main panel are now empty. The 'B-Tree' panel remains unchanged from the previous screenshot.

- To search the **marks** values and find a field for a given **marks** value. (Search by marks)

The application window is titled "Merge Sort and Binary Search". It contains two main panels: "B-Tree" on the left and "Merge Sort and Binary Search" on the right. Both panels have input fields for "Name" and "Marks", an "Insert" button, a "Search Marks" field, a "Search" button, a "Reset" button, and a "Delete Marks" field with a "Delete" button. In the "Merge Sort and Binary Search" panel, the "Search Marks" field contains the value "75". The "Search" button is highlighted with a black arrow. Below the "Search" button, a list of ranks is displayed:

Rank	Name	Marks
1.	Dilini	93
2.	Nushan	89
3.	Dilshan	80
4.	Prasad	75
5.	Hasith	72

Once search button clicked search result will appear under the search button

The application window is titled "Merge Sort and Binary Search". It contains two main panels: "B-Tree" on the left and "Merge Sort and Binary Search" on the right. Both panels have input fields for "Name" and "Marks", an "Insert" button, a "Search Marks" field, a "Search" button, a "Reset" button, and a "Delete Marks" field with a "Delete" button. In the "Merge Sort and Binary Search" panel, the "Search" button is highlighted with a black arrow. Below the "Search" button, the search results are displayed:

Prasad : 75
Deshan : 75

The "Rank" list is updated to include Deshan: 75 at rank 6:

Rank	Name	Marks
1.	Dilini	93
2.	Nushan	89
3.	Dilshan	80
4.	Prasad	75
5.	Deshan	75
6.	Hasith	72

- To delete a field from the table when given the **marks** value.

The screenshot shows a software interface with two main panels: "B-Tree" on the left and "Merge Sort and Binary Search" on the right. Both panels have input fields for "Name" and "Marks", an "Insert" button, a "Search Marks" input field, a "Search" button, a "Rank" list, a "Reset" button, and a "Delete Marks" input field with a "Delete" button. In the "Merge Sort and Binary Search" panel, the "Delete Marks" input field contains the value "89". An arrow points from this input field to the "Delete" button. The "Rank" list in this panel shows the following data:

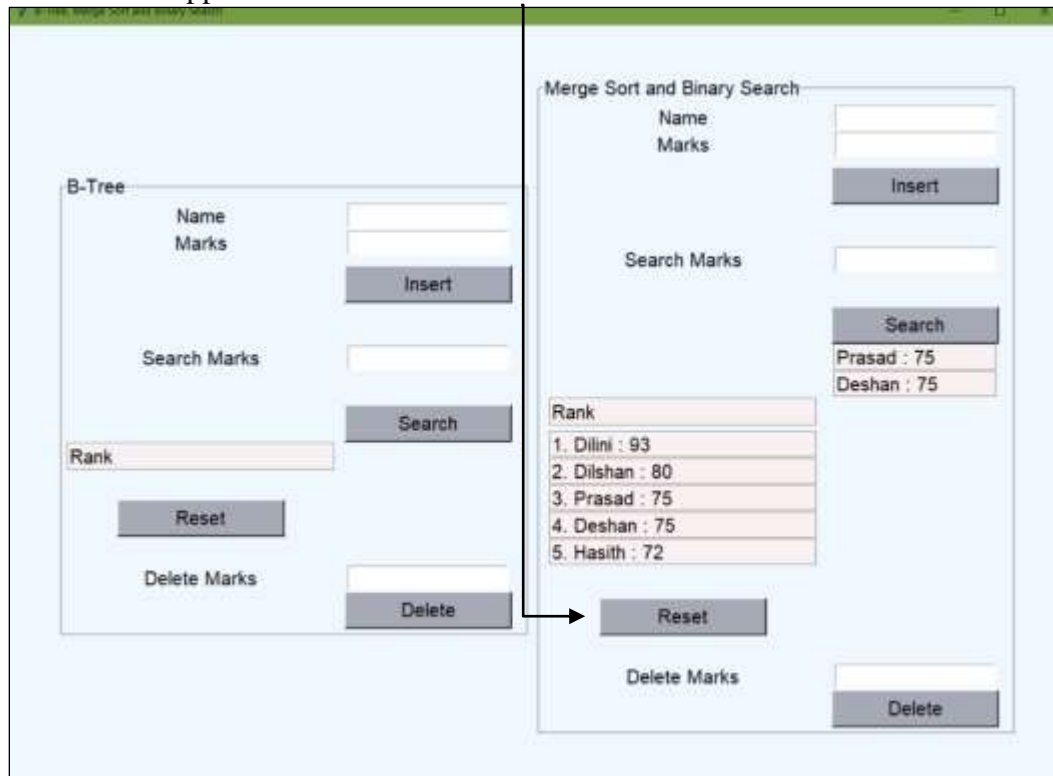
Rank	Name	Marks
1.	Dilini	93
2.	Nushan	89
3.	Dilshan	80
4.	Prasad	75
5.	Deshan	75
6.	Hasith	72

Type the mark value to be deleted. Click **Delete**
The Field will be deleted.

This screenshot shows the same software interface as the previous one, but after the deletion operation. The "Delete Marks" input field in the "Merge Sort and Binary Search" panel is now empty. The "Rank" list now shows only five entries, as the entry with a mark of 89 (Nushan) has been removed:

Rank	Name	Marks
1.	Dilini	93
2.	Dilshan	80
3.	Prasad	75
4.	Deshan	75
5.	Hasith	72

- Both applications have the **Reset** button to reset all the values



Once Clicked each reset button resets the sorted values to be empty

