Searching and Sorting

The program starts with the import statements, importing the necessary classes (Arrays and Scanner) from the java.util package.

The SortingAndSearching class is the main class that contains the main method. This is where the program execution starts.

Inside the main method, a Scanner object is created to read input from the user.

The user is prompted to enter the size of the array.

An array of integers, array, is created with the specified size.

The user is then prompted to enter the elements of the array.

The user is also asked to enter the element to search.

The program displays a list of sorting algorithms and prompts the user to choose one.

Similarly, the program displays a list of searching algorithms and prompts the user to choose one.

The original array is printed using Arrays.toString().

A switch statement is used to perform the selected sorting algorithm on the array. Depending on the choice, the corresponding sorting method (bubbleSort, selectionSort, insertionSort, mergeSort, or quickSort) is called.

After sorting the array, the sorted array is printed along with the name of the sorting algorithm.

Another switch statement is used to perform the selected searching algorithm on the array. Depending on the choice, the corresponding searching method (linearSearch or binarySearch) is called.

After performing the search, the program displays whether the element was found or not, along with the index of the element (if found).

Finally, the Scanner is closed to release system resources.

The code then defines several sorting algorithm methods (bubbleSort, selectionSort, insertionSort, mergeSort, and quickSort) and searching algorithm methods (linearSearch and binarySearch).

Each sorting algorithm method takes an array of integers as input and sorts it using the specific algorithm.

Each searching algorithm method takes an array of integers and a target element as input and searches for the target element using the specific algorithm. It returns the index of the element if found, or -1 if not found.

In addition to the sorting and searching algorithms, the code defines abstract classes SortingAlgorithm and SearchingAlgorithm. These classes serve as a base for the individual sorting and searching algorithm classes.

The sorting algorithm classes (BubbleSort, SelectionSort, InsertionSort, MergeSort, and QuickSort) extend the SortingAlgorithm class and override the sort method. They provide the implementation for their specific sorting algorithm.

The searching algorithm classes (LinearSearch and BinarySearch) extend the Searching Algorithm class and override the search method. They provide the implementation for their specific searching algorithm.

Note that there’s an execution time in nanoseconds for each sorting and searching algorithm. This helps the user know how long it took for the execution to be carried.

By organizing the code in this way, each sorting and searching algorithm is encapsulated in its own class, making the code modular and easier to understand and maintain. The main class handles the user interaction and orchestrates the sorting and searching operations.