cct College Dublin
Computing • IT • Business

Student Full Name	Luis Medina Burgoa
Student Number	2024283
Module Title	Algorithms & Constructs
Assessment Title	System Modelling & Build
Lecturer/Supervisor	Muhammad Iqbal
Assessment Due Date	7th April 2025
Date of Submission	10th May 2025

Use of AI Tools

I have not used any AI tools or technologies in the preparation of this assessment.

Declaration

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

Contents

Introduction

For this assignment, it was asked to simulate how a Tech Company system works using a command-line interface on Java Netbeans, where we need to be able to input data, adding new employees for example. This data includes employee details such as, name, surname, position, salary, department, entity, which it was organized properly to be able to retrieve it easily when it is needed. As it was required, parent classes were implemented along with their child classes using an object-oriented structure. Also, to avoid the system to crash, it was necessary to implement input validations.

This report is going to focus on the decisions of implementing Merge sort and Binary Search algorithms, and why they were chosen.

Sorting Algorithm

Merge Sort Recursive

This assignment requires to sort the list of employees alphabetically read from the file name Application_Form.txt. As this list contains a large number of employees stored it is necessary to find a method that can have a good performance.

For this reason, the Merge sort was chosen as the sorting algorithm due to the reasons bellow:

- Merge sort always maintains in all cases its time complexity O(n log n), which provides a stable sort and performance, making this algorithm reliable even in the worst case.
- It uses the divide and conquer technique which means it divides the array into subarrays until it has only one element.
- It is very useful to manage a large amount of dataset with reliability.

Comparing to other algorithms, like Quick sort that can have a poor performance or bubble sort that it can be easy to implement, but it is not efficient with large datasets. Meanwhile, Merge sort provides a good balance between performance and stability.

Searching Algorithm

Binary Search Recursive

For the next part of the assignment, it was required to implement a search algorithm. As we have a large list of employees and sorted, performance is crucial to have a good experience.

That is why, the Binary sort was selected as the searching algorithm because of the following facts:

- It works effectively when the list is already sorted.
- It is ideal for large datasets due to it works dividing it in half and comparing it until finds the target.

Unlike other methods, that are easier to implement, they are inefficient for large datasets and consume more resources. Binary Search recursive in a sorted list, along with the already implemented merge sorting recursive algorithm, works efficiently for this assignment.

Conclusion

To conclude, as Merge Sort and Binary Search provide an efficient performance for large amount of dataset and our Tech Company can store a large list of employees, it is crucial to use a reliable algorithm to organize properly the data. By implementing those algorithms, the program aim to have a stable performance, scalable solution easy to maintain and provide a user-friendly experience which is important in this type of organization.

GitHub Link

https://github.com/lubum593/CA2_2024283.git

References

Board Infinity, 2023. Time Complexity of Sorting Algorithms. Available: https://www.boardinfinity.com/blog/time-complexity-of-sorting-algorithms/. [Accessed 24 April 2025]

Kumari, Yukti, 2024. Time & Space Complexity of Searching Algorithms. Available at: https://www.naukri.com/code360/library/time-space-complexity-of-searching-algorithms. [Accessed 26 April 2025]

Parekh, Dhanam, 2022. Sorting Algorithms: Slowest to Fastest. Available at: https://builtin.com/machine-learning/fastest-sorting-algorithm. [Accessed 25 April 2025]

Ranjan, Abhishek, 2024. Time and Space Complexity of Sorting Algorithms. Available at: https://www.naukri.com/code360/library/time-and-space-complexities-of-sorting-algorithms-explained. [Accessed 29 April 2025]

WsCubetech, 2024. Time and Space Complexity of Searching Algorithms. Available at: https://www.wscubetech.com/resources/dsa/time-space-complexity-searching-algorithms. [Accessed 25 April 2025]