



CST2550

**Software Engineering Management and Development**

**Coursework – Library Management System**

**Autumn/Winter term**

**2024/2025**

**Date of Submission: 01/07/2025**

**Student Name: Teeluckdharry Harsh Sharma Student ID Number: M01014018**

**Lab Tutor: Aisha Idoo**

Library Management System

[Introduction and description of the project 3](#_Toc202000342)

[Justification of selected data structures and algorithms 3](#_Toc202000343)

[Time complexity analysis with pseudo-code 3](#_Toc202000344)

[Testing approach and test cases 4](#_Toc202000345)

[Conclusion 4](#_Toc202000346)

[References 4](#_Toc202000347)

## Introduction and description of the project

This project involves the design and implementation of a Library Management System (LMS) for a local community library transitioning from paper-based records to a digital platform. The primary goal is to develop an efficient and user-friendly system using C# with Entity Framework for database operations. The system is designed to manage library resources including books, authors, and customers through relational database architecture.

## Justification of selected data structures and algorithms

This project utilises Entity Framework's DbContext which internally implements several optimised data structures such as Hash table and Collection. It provides automatic optimisation for database operations, memory management, and relationship handling. All entities use Guid as primary keys instead of integer auto-increment values as it ensures global uniqueness across the system. The BookAuthorCrossReference entity implements a many-to-many relationship pattern to allow efficient querying of relationships.

Binary search is used because it's a efficient way to find a specific element within a sorted list or array, significantly faster than linear search, especially for large datasets. It works by repeatedly dividing the search interval in half, effectively eliminating half of the remaining elements in each step, leading to a logarithmic time complexity (O(log n)).

## Time complexity analysis with pseudo-code

**Binary Search Algorithm**

**Pseudo-code**:

ALGORITHM BinarySearch(sortedArray, target)

INPUT: sortedArray[0..n-1], target

OUTPUT: index of target or -1 if not found

BEGIN

left ← 0

right ← length(sortedArray) - 1

WHILE left ≤ right DO

mid ← left + (right - left) / 2

IF sortedArray[mid] = target THEN

RETURN mid

ELSE IF sortedArray[mid] < target THEN

left ← mid + 1

ELSE

right ← mid - 1

END IF

END WHILE

RETURN -1

END

**Time Complexity Analysis**:

* **Best Case**: O(1) - Target found at first comparison
* **Average Case**: O(log n) - Target found after log₂(n) comparisons
* **Worst Case**: O(log n) - Target not found, complete binary division

## Testing approach and test cases

## Conclusion

## References

Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C. (2022) Introduction to algorithms. 4th edn. Cambridge, MA: MIT Press.

Freeman, A., Ivanova, E. and Japikse, P. (2021) Pro C# 9 with .NET 5: foundational principles and practices in programming. 10th edn. Berkeley, CA: Apress.

https://www.researchgate.net/profile/Kumaresan-Durvas-Jayaraman/publication/392863342\_Harnessing\_the\_Power\_of\_Entity\_Framework\_Core\_for\_Scalable\_Database\_Solutions/links/6855836393040b17338c456b/Harnessing-the-Power-of-Entity-Framework-Core-for-Scalable-Database-Solutions.pdf