2021

James Scott

N9934618

5/10/2021

CAB301 Assignment 1

CAB301 – Algorithms and Complexity

Software Application for a Tool Library

Table of contents

[Introduction 2](#_Toc72783095)

[Design and Analysis of Algorithms 3](#_Toc72783096)

[Algorithm Design 3](#_Toc72783097)

[Algorithm Analysis 4](#_Toc72783098)

[Software Test plan and results 4](#_Toc72783099)

## Introduction

I have attempted to implement each of the required functionalities in both the Staff and Member menus. I made many difficult design choices on how to approach the problems posed by the interfaces, I will explore the most notable choices here.

Firstly, I was not able to come up with an eloquent solution to the problem of MemberCollection storage. To get things working for submission I have used a MemberCollection at the program.cs level and an internal MemberCollection within ToolLibrarySystem.cs. This decision later posed challenges with updating/using the correct MemberCollection, but I could not think of a cleaner alternative within the bounds of the assignment.

Furthermore, as my Jagged Arrays of ToolCollection were stored as private members of my ToolLibrarySystem class, a lot of the user input and console logic has been implemented within the ToolLibrarySystem class. Obviously, it would be ideal to extrapolate this logic out to the program.cs level, but I have tried to mitigate this as best I can with utility functions to clean up the console logging logic within ToolLibrarySystem.cs.

I have made no changes to or deviations from the interfaces, and all discovered bugs have been ironed out before submission.

## Design and Analysis of Algorithms

#### Algorithm Design

The algorithm I chose for the Top 3 borrowed tools functionality was a Heap sort algorithm. I chose this algorithm due to its worst case performance combined with the fact that it is an in-place algorithm. There are no unresolved bugs that were discovered throguhout my implementation and testing.

A heap sort is

Pseudocode:

Introduction - a general discussion on the algorithm chosen. The reason for the choice, why other algorithms rejected,  any problems or challenges implementing the algorithm, declaration of any bugs that may be present.  
2. Description - A comprehensive description of how the algorithm works - if there are multiple components to the algorithm discuss how the elements work together, discuss the data structures used in the algorithm as this can sometime affect the algorithm efficiency or the difficulty in the implementation. This section can probably be broken up to include  
   a. Pseudocode  
   b. Complexity analysis - mathematical  
   c. Empirical analysis - discuss how this was achieved (time/basic operations)  
   d. Summary of the final Big O calculation and justification by the maths and the empirical analysis  
3. Summary - A general discussion on the effectiveness of the algorithm, any potential problems you think should be addressed (e.g. extra space required, difficulty in implementation) and the general performance of the algorithm  
You may need to consult the Text for this unit - or you may need to use Google to find information about the algorithm [https://doc.lagout.org/science/0\_Computer%20Science/2\_Algorithms/Introduction%20to%2[…]Algorithms%20%283rd%20ed.%29%20%5BLevitin%202011-10-09%5D.pdf](https://doc.lagout.org/science/0_Computer%20Science/2_Algorithms/Introduction%20to%20the%20Design%20and%20Analysis%20of%20Algorithms%20%283rd%20ed.%29%20%5BLevitin%202011-10-09%5D.pdf)  
I would reference any texts that you used or any websites that you used in researching the algorithm. A link to Cite Write - <https://www.citewrite.qut.edu.au/>

Show less

#### Algorithm Analysis

The Algorithm analysis should include a theoretical analysis for the determination of the algorithm complexity and I would recommend including an empirical analysis either using time or counts of basic operations. The empirical analysis can support your theoretical analysis and provides a well rounded and complete analysis of your Top 3 algorithm.

## Software Test plan and results

Finally the Software Test plan providing the screenshots of all the required functionality and a single screenshot of how your application appears at the end of each process (e.g. adding a member, displaying tools) - I would also include in the test plan some evidence of input validation to show that you have considered this as part of your implementation so numbers out of range, wrong data type inputs with the appropriate screen shots.

|  |  |
| --- | --- |
| **Task** | **Expected Outcome** |
| **Main Menu** | |
| Access staff menu | When correct credentials are supplied, success. |
| Access member menu | When correct credentials are supplied, success. |
| Exiting the program |  |
| **Staff Menu** | |
| Add a new tool to the system |  |
| Increase the quantity of a tool |  |
| Decrease the quantity of a tool |  |
| Register a new member |  |
| Search for a members contact number |  |
| Exit to main menu | Success |
| **Member Menu** | |
| List all the tools of a tool type |  |
| Borrow a tool |  |
| Return a tool |  |
| Display all of my currently borrowed tools |  |
| Display the three most frequently rented tools |  |
| Exit to main menu | Success. |