ENSF 338 – Winter 2023

Practical Data Structures and Algorithms

Final Project: Library for Common Data Structures

Instructor: Dr. Lorenzo De Carli

Group 55 Submission for L01/B01

Submitted by: Maged El Habiby

UCID: 30144718

Partner: Mazen El Habiby

UCID: 30144614

Repo Link: https://github.com/maged-elhabiby/ENSF338-Project-Maged-and-Mazen-Group-55

UNIVERSITY OF CALGARY

OUR VIEW OF LIBRARY DESIGN

We created library design with the purpose of it being modular, has reusability and to be organized so it can be easy to use. We made it so we can they find which data structure that user wants to import from the library. For example, "import mylib.datastructures.linear.SLL;". In this library, we decided our common data structures into three folder nodes, linear and tree. In which in nodes folder we have our SNode and DNode which we use in the files in linear folder and TNode for the files in Tree folder. The linear folder contents has data structures for singly Linked List(SLL), doubly Linked List (DLL), Circular Singly Linked (CSLL) which extends SLL, Circular Doubly Linked List(CDLL) which extends to DLL , Stack based on Singly Linked List (StackLL) which extends SLL and Queue based on Singly Linked List (QueueLL) which extends SLL. The trees folder content has data structures for the Binary Search Tree (BST) and the self-balancing AVL tree (AVL). Furthermore, our code is well documented, explaining purpose of the functions. Finally, our library is well tested using Junit tester to ensure that all is working as expect.

Overview of Library Design:

- mylib
    - datastructures
        - nodes
            - SNode
            - DNode
            - SNode
        - linear
            - SLL
            - DLL
            - CSLL -> Extends to SLL
            - CDLL -> Extends to DLL
            - QueueLL -> Extends to SLL
            - StackLL -> Extends to SLL
        - Tree
            - AVL -> Extends to BST
            - BST
- Mylib
    - SNodeTest
    - DNodeTest
    - TNodeTest
    - SLLTest
    - DLLTest
    - CSLLTest
    - CDLLTest
    - StackLLTest
    - QueueLLTest
    - AVLTest
    - BSTTest

CONTRIBUTION BREAKDOWN

We split project evenly between us, where each person has contributed in the each aspect of the library from Linear to Trees data structures. We used GitHub and direct communication to build our data structure library. We each tested and contributed to each library.

ANY BONUS ELEMENTS ATTEMPTED

We attempted:

- The Jar Packaging
- AVL Balancing constructor (not successive insertion)
- AVL delete with balancing updates

ANY SPECIFIC INSTRUCTIONS/REQUIREMENTS TO SUCCESSFULLY USE YOUR YOUR LIBRARY

We Packaged it into Jar file, so once implemented to project structure, it becomes easy to use. To use the library, just import what data you want to use, for example “import mylib.datastructures.tree.BST;” or “import mylib.datastructures.nodes.TNode;” or “import mylib.datastructures.linear.DLL;”.

The Jar file is located in the target Folder under the name “datastructures-1.0V.jar”.