Utilizing Doubly Linked Lists and Sorting Algorithms In Professional Football Transfers

Blake McFarlane (U14766550)
Matthew Sharp (U)
Rima El Brouzi (U25027078)

I. Overview

This coding project simulates the role of a football team manager during the transfer season, allowing the user to experience the strategic considerations of building a team within a fixed budget. The program is built around a core implementation of doubly linked lists, which efficiently manage the roster of available players and ensure dynamic data manipulation as users select or sort players based on various attributes such as price, rating, and position.

Utilizing custom sorting algorithms, the project enhances user interaction by allowing the manager to prioritize players by specific fields, facilitating a more tailored team-building approach. This setup not only replicates the complexities of real-life player selection and team budgeting but also incorporates fundamental computer science concepts like data structures and algorithm optimization to create a practical, educational, yet fun simulation environment.

II. Details of Implementation

In the implementation of the football team management simulation, the core data structure employed is the doubly linked list, utilized through the FootballDList class. This choice ensures efficient management of player data, facilitating quick insertions and deletions from both ends of the list and anywhere in between, which is essential for dynamically updating the team roster during the drafting process. The doubly linked list structure allows for bidirectional traversal, making operations like reordering (based on specific attributes like player ratings or costs) and player look-ups (both forward and backward) more straightforward and time-efficient.

Custom sorting algorithms, such as insertion sort were adapted for doubly linked lists. These functions were sortPlayersByCost and sortPlayersByRating and they organized players by cost or rating in descending order. This method was ideal for small data sets and offered simplicity and effective performance. Moreover, the prioritizePosition function used a targeted sorting approach to move players of a specific position to the front of the list which enhanced the manageability of player data and facilitated strategic decision-making based on urgent positional needs. This combination of data structures and tailored sorting mechanisms created a responsive and intuitive environment for simulating football team management.

III. Challenges

Some of the challenges we faced when building our project ranged in topics from sorting algorithm implementation, logic handling, syntax errors, and even UI/UX design. Starting with sorting algorithm implementation, one of our biggest hurdles was ensuring that the user would be able to effectively sort and filter through the list of available players. We navigated this by coming up with the three essential sorting mechanisms we were determined to incorporate; sorting by price, rating, and position were the three variables we thought were most impactful in a user's decision when participating in the player transfer. From there we were able to separate our requirements into two main algorithms we would implement; one being based off of an insertion sort and the other off of a custom targeted searching approach.

Ensuring that positions were accurately tracked and preventing user input errors proved to be complex. These issues required careful consideration of conditional logic and robust input validation measures to manage user interactions effectively. Additionally, the UI/UX design presented its own set of challenges. Crafting an intuitive and user-friendly interface that accommodated the complexity of player transfers while maintaining aesthetic appeal was particularly demanding. Balancing functionality with simplicity in the UI required multiple iterations and feedback sessions to ensure the system was accessible and straightforward for users.

IV. Contributions

In our team of three, each member contributed significantly to different aspects of the project. Matthew implemented the doubly linked list class and its methods for managing the player list. Blake focused on developing the sorting algorithms and making the project robust against user errors. Rima initiated the project idea, worked on the program design, and developed the presentation. All members supported each other and met outside class hours to ensure cohesive progress and integration of our work.

Presentation Link:
https://www.canva.com/design/DAGDRnylTjs/0Y5hbY0UNdKAv2oxUYZFvA/