**Project Report: Game Menu and AVL Tree Implementation in C++ using SFML**

***Contributors***

***22I-1214(Muhammad Haziq Naeem)***

***22I-1289(Muneeb ul Islam)***

***22I-1186(Abdullah Azeem)***

***Overview:***

The provided code implements a maze game in C++ using SFML graphics library. It encompasses various aspects such as maze generation, shortest path finding, combat mechanics, enemy encounters, efficient data structure operations, integration with AVL tree for managing inventory, reward system, user interface design, and file handling. Each aspect is evaluated based on specific criteria outlined in the rubric.

***Maze Generation:***

The maze generation in the provided code dynamically creates maze layouts using a depth-first search algorithm. This approach ensures that each maze generated is unique, adding variety and challenge to the gameplay. The maze complexity and variety are commendable, contributing to an immersive gaming experience.

***Efficiency in Finding Paths Through the Maze:***

The implementation of Dijkstra's algorithm for finding the shortest path through the maze is correct and efficient. It allows players to navigate through the maze while considering obstacles and dead-ends, ensuring they reach their destination optimally.

***Handling of Obstacles and Dead-Ends:***

The code effectively handles obstacles and dead-ends within the maze, preventing players from progressing through blocked paths. This adds strategic depth to the gameplay, requiring players to explore alternative routes to reach their objectives.

***Combat Mechanics and Enemy Encounters:***

The combat system implemented in the code provides players with engaging battles against enemies encountered throughout the maze. It offers strategic depth, allowing players to use various tactics and items acquired from treasure chests stored in AVL tree inventory.

***Integration with AVL Tree:***

The integration of AVL tree for storing treasure information is a key aspect of the game. It efficiently handles insertion, deletion, and retrieval operations, ensuring optimal management of player inventory. Additionally, proper balancing of the tree is maintained to prevent performance degradation.

***Reward System:***

The reward system includes valuable treasures and artifacts scattered throughout the maze, providing players with incentives to explore and engage in combat encounters. These items offer unique abilities and bonuses, enhancing the gameplay experience.

***User Interface Design:***

The user interface design features real-time feedback on player progress, inventory management, and combat encounters. It offers ease of use and navigation, enhancing the overall gaming experience. The inclusion of sound effects adds immersion to the gameplay, providing auditory feedback during key events such as combat encounters and treasure discoveries.

***Enhancement***:

The random generation of the maze layout ensures that each playthrough offers a unique and challenging experience. This adds replay value to the game and keeps players engaged over multiple sessions.

***Bonus***:

The code demonstrates creativity in its implementation of maze generation, combat mechanics, enemy encounters, reward system, and user interface design. These elements combine to create an engaging and immersive gaming experience for players. Additionally, proper file handling mechanisms for loading and saving game data ensure that player progress is preserved between sessions.