Unit Testing

Table of Contents

Unit Testing

The test plan for "A Blocky Start" is meticulously designed to guarantee the software's quality, focusing on ensuring that every component and functionality adheres to the highest standards. The core objective of this plan is to validate both the functional and non-functional requirements of the game, ensuring that it not only performs the tasks it was designed to do but also performs them efficiently and effectively under various conditions.

Unit testing is the foundation of our approach, examining each unit of the software as defined in the source code. This includes individual components, classes, and methods that drive the game's logic and functionalities. By isolating these units, we can carefully examine their behaviour and reliability, ensuring that each part functions correctly within the system. Our testing strategy is two-pronged, encompassing both white box and black box testing methodologies. White box testing allows us to examine the game's internal structures, focusing on the control flow, decision paths, loops, and boundary cases. This method is instrumental in ensuring that all logical paths are thoroughly tested and helps in identifying any errors in the logic that could lead to unexpected behaviours. Black box testing is used to assess whether the game meets its predetermined functional and non-functional requirements without considering its internal workings. This method is crucial for identifying any discrepancies between the intended functionality of the game and its actual performance, ensuring that the end product meets the users' expectations and requirements.

For components developed in Java, JUnit serves as our primary testing tool, providing a robust framework for conducting unit tests. This choice allows for a systematic and efficient testing process, enabling us to cover all major functions of the software's code comprehensively. The use of JUnit facilitates the creation of a structured testing environment where tests are not only repeatable but also automated to a significant extent. This automation is particularly valuable for testing the logic of the program, streamlining the testing process while ensuring thorough coverage of the software's functionalities. However, it's noteworthy that components requiring direct human interaction, such as the GUI, are not subjected to automated testing, acknowledging the intrinsic value of manual testing in assessing user-centric features.

Overall, this testing plan is crafted to be both sufficient and complete, ensuring that "A Blocky Start" is rigorously tested for quality, functionality, and performance. By employing a blend of white box and black box testing, alongside using the capabilities of JUnit, this plan lays a solid foundation for delivering a software product that is not only bug-free but also aligns with the educational and engagement goals set forth by the our team.