REPORT

Overall Approach to Implementing the Game

The overall approach to implementing the game is structured across three key milestones, each focusing on specific aspects crucial for the development process.

Milestone 1: This initial milestone focuses on learning Java fundamentals essential for the game development while also in initiating the game's foundation. It involves gaining proficiency in Java programming, with a focus on drawing and animation techniques to lay the groundwork for the game's visual aspects. Dividing the work among team members allows for a focused approach to mastering these fundamental skills.

Milestone 2: Building upon the groundwork laid in Milestone 1, Milestone 2 focuses on implementing core gameplay mechanics and features. Key tasks include enabling player movement, enemy movement, and object collision detection. Additionally, implementing entity interactions and collision detection enhances the game's complexity and immersion. Integrating game states and user interface (UI) screens cater to seamless transitions between different game states.

Milestone 3: The final milestone revolves around testing and refining the game's logic and adding essential features to complete the gaming experience. This includes implementing point systems, victory conditions, and reward collection mechanisms. Ensuring a polished user interface and incorporating feedback loops for player progression enhances the overall gameplay experience.

Adjustments and Modifications to the Initial Design of the Project

The adjustments and modifications to the initial design of the project reflect an ongoing learning process aimed at enhancing efficiency and streamlining the implementation process. One notable adjustment that we made was to involve the extension of the design to incorporate additional features such as rendering, which enriches the visual presentation of the game and contributes to a more immersive player experience. This expansion aligns with the team's objective of creating a polished and engaging gameplay environment.

Another significant modification involves the removal of the position attribute from the Cell class. This decision stems from the realization that there are simpler methods available to handle positioning within the game without the need for aggregation. By eliminating the position attribute from the Cell class, the team simplifies the overall design structure, reducing unnecessary complexity and potential sources of confusion. Furthermore, this adjustment is justified by the recognition that the position of cells on the board is not directly utilized within the game's mechanics, rendering it superfluous to the core functionality.

Overall, these adjustments and modifications represent a proactive approach to refining the project design, ensuring that it remains agile and adaptable to evolving requirements while prioritizing simplicity, efficiency, and clarity in implementation.

Management process of this Phase and the Division of Roles and Responsibilities

During this phase, the team's management process was structured to ensure effective coordination and progress tracking. The team met twice a week to discuss progress, address challenges, and align on next steps. Additionally, to maintain version control and collaboration, a protocol was established where each time a team member pushed code to GitHub, they announced it on Discord. This practice helped ensure that everyone was aware of the latest updates and prompted team members to pull changes to their local repositories.

Communication played a crucial role in managing the project, with discussions primarily taking place on Discord and Zoom meetings. These platforms facilitated real-time collaboration and enabled team members to seek clarification, provide updates, and share ideas. Furthermore, regular meetings helped track progress, identify bottlenecks, and adjust priorities as needed, ensuring that the project remained on track.

The approach to implementing the game follows a structured progression divided into three key milestones. Recognizing the novelty of Java and game development for the team, Milestone 1 prioritizes learning essential Java functionalities, focusing on drawing and animation techniques crucial for game design. Each team member assumes distinct responsibilities tailored to their strengths and interests. Pardeep spearheads the development of the game engine, leveraging abstract classes and utilities to streamline the process, while also leading animation implementation and level construction. Tom takes charge of game states and the creation of the house class, pivotal for game progression. Will concentrates on crafting the Knight class, along with managing reward mechanisms through the Reward, Meat, and Coin classes. Manya undertakes the creation of enemy entities, including the Goblin, Trees, and Dynamite classes, crucial for adding depth and challenge to gameplay, and she also meticulously compiles the project's progress and insights into the comprehensive report, ensuring thorough documentation and reflection on the team's journey and achievements. With this collaborative effort, the team aims to lay the groundwork for an engaging and visually captivating gaming experience, with meticulous attention to detail across all game elements.

With this collaborative approach and a focus on transparent communication and clear division of roles, our team remains poised to lay the groundwork for an engaging and visually captivating gaming experience, marked by meticulous attention to detail across all game elements.

External Libraries

As of now, we haven't incorporated any external libraries into our project. This deliberate choice reflects our intention to maintain simplicity and align with our learning objectives.

Enhance the Quality of Code

To enhance the quality of our code, we adopted several key strategies. Firstly, we prioritized clear and descriptive naming conventions for variables, methods, and classes, ensuring readability and understanding among team members. Extensive use of comments, including simple JavaDoc comments, provided crucial context and explanations for complex algorithms and method functionalities. Consistent formatting and adherence to industry best practices further promoted readability and maintainability. Our modular design approach facilitated code reusability and collaboration, while UML class diagrams visualized class relationships and dependencies, aiding comprehension of the project's architecture. Version control systems like Git enabled organized code collaboration and tracking of changes. Regular code reviews and comprehensive documentation, including project overviews and setup instructions, fostered continuous improvement and accessibility.

Challenges

Indeed, facing a steep learning curve with Java and game development was a significant challenge during Milestone 1. Many team members had limited prior experience with Java, necessitating dedicated time and effort to grasp essential concepts and techniques. This learning process was further compounded by the demands of midterm week, which left team members stretched thin and unable to devote as much time to the project as desired. Balancing academic commitments with project requirements proved to be a delicate juggling act, slowing progress and causing occasional frustration. However, despite these challenges, our team remained resilient and committed to overcoming obstacles, leveraging collaborative efforts and effective time management to make steady progress. The experience served as a valuable lesson in prioritization, adaptability, and perseverance, ultimately strengthening our resolve and deepening our understanding of Java and game development fundamentals.