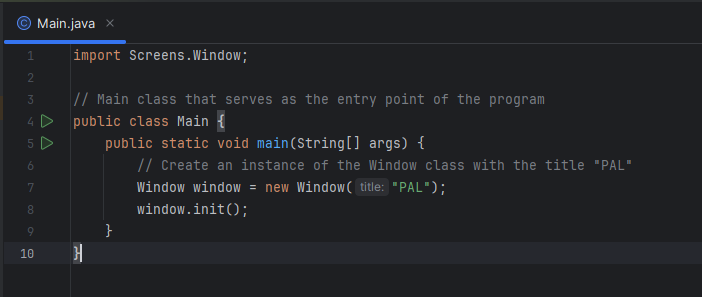
**Game Development Report**

**Overview**

This report summarizes the development of a Java-based game. The game features a variety of components including fruits, pizza slices, baskets, and obstacles. The main gameplay involves catching falling fruits and pizza slices into baskets while avoiding obstacles.

**Main Class:**



It is the entry point of the application. It creates a customized window object and initializes all the required objects/components.

**Key Features and Implementations**

1. **Component System**

* **Base Component Class**: Created an abstract base class, **Component**, to serve as a foundation for all moving objects in the game, such as fruits and pizza slices. This class includes common properties like position, speed, and image frames.
* **Fruit and Pizza Slice Implementation**: Implemented specific classes for **Fruit** and **PizzaSlice**, extending from **Component**. These classes handle the specific behaviors and animations of fruits and pizza slices.
* **Polymorphism**: Utilized polymorphism for efficient handling of different game objects. Both fruits and pizza slices are treated as **Component** objects, allowing for simplified management in the main game loop.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**2. Game Panel and Control**

* **Game Panel**: Developed the **GamePanel** class as the main canvas for the game, responsible for game rendering and updates.
* **Collision Detection**: Implemented collision detection for handling interactions between components (fruits, pizza slices) and baskets.
* **Mouse Input Handling**: Created a **MouseInputHandler** to manage drag-and-drop interactions, allowing players to catch and drag fruits and pizza slices.

A screenshot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated

**3. Obstacles**

* **Dynamic Obstacle Generation**: Introduced obstacles that spawn randomly on the screen. Obstacles are represented by various colored images and are positioned away from the screen edges to ensure fair play.
* **Collision with Obstacles**: Programmed logic to remove fruits or pizza slices upon collision with obstacles, adding a layer of challenge to the game.

**4. Scoring and Lives**

* **Scoring System**: Built a scoring system where players earn points by successfully catching fruits or pizza slices in baskets.
* **Lives Mechanics**: Implemented a lives system, where players start with a set number of lives, losing one each time a basket moves off-screen without being filled.

A screenshot of a computer program

Description automatically generated

**5. Start Screen and User Interface**

* **Start Screen**: Designed a start screen with options to play the game, view high scores, or exit. The screen includes a simple and intuitive layout with button hover effects.
* **Player Name Input**: Added functionality for players to enter their names, enhancing the personalized experience.
* **High Score Management**: Developed a system to save and display high scores, retaining player names and scores in a text file.

A screen shot of a computer program

Description automatically generated

**6. Game Over Handling**

* A computer screen shot of a program

  Description automatically generated**Game Over Screen**: Implemented a game over screen that displays when all lives are lost. It shows the player's score and prompts for exiting or restarting the game.

**7. Graphics and Animation**

* **Sprite Handling**: Managed sprite sheets for fruits and pizza slices, extracting individual frames for animations.
* **Obstacle Graphics**: Loaded and displayed different colored obstacle images.

**Conclusion**

The game's development involved a range of Java programming techniques, including object-oriented programming, event handling, collision detection, and graphics rendering. The use of polymorphism and abstract classes provided a robust and extendable architecture, allowing for easy additions and modifications to the game's components. The final product is a visually appealing and interactive game that challenges players to skillfully catch falling items while avoiding obstacles.

ThankYou!