CSCI 5548 - Object Oriented Analysis and Design
Semester Project - Final Report

1. **Project Title -** Ascii Attack

   Team Members - Prashanth Vamanan Srinivasan, Ashwin Viswamithiran

2. **Final State of System:**

   We were able to implement most of the features that we listed as part of our design phase in Project 5. These features are listed below:

   - Loading screen with delay.

   - Welcome screen with the ability to start a new game, view rules, view high scores and quit the game.

   - Main game loop with the following elements:

     - Randomly falling letter, number and special blocks at regular intervals
     - Fixed number of blocks spawning for each level.
     - Ability to destroy both normal and special blocks.
     - Ability to restart the game.
     - Sound effects and background music in loop
     - Corresponding update of UI elements such as score, blocks left, misses allowed, current level number and high score.
     - A game over screen that allows users to enter their name to track top 10 high scores in the game.

   The only feature we were not able to implement was the back button used to go back to the welcome screen from the view rules and view leaderboard screens. We were not able to achieve this as we did not have a deeper understanding of transitioning between the various screens using pygame.

   As this was a minor feature add, not part of the core game functionality, it did not impede our progress towards implementing the main game logic and design patterns to support the game.

Changes from Project 5 and 6:

We were able to complete the majority of our project as we planned, since we invested significant time in designing our classes and application flow. All of the Project 5 timeline was spent in designing our application and we started development from Project 6. At the end of Project 6, we had completed the main logic of our game that included setting up the loading and welcome screens and enabling random instantiation of blocks in the main game loop.

This base work set us up nicely for Project 7. Since we already had the core game functionality working, it was easier to add further functionality including the ability to destroy blocks, instantiating special blocks that required multiple hits, having a game over screen, ability to track high scores and restart the game. We also implemented **four design patterns** that support our game functionality during Project 7.

- **Singleton** - To ensure single instances of resource intensive classes such as GameManager, LevelManager, UIManager and BlockManager.

- **Factory -** To separate the instantiation of letter and number blocks from their usage.

- **Command -** To encapsulate the actions required to destroy a block on key press.

- **Decorator -** To add the ability to require multiple hits for special blocks at run time.

3. **Final Class Diagram and Comparison Statement:**

- Final Class Diagram - Please find the class diagram linked here

- Initial Class Diagram - Please find our initial class diagram linked here

  The final version of our class diagram included some additional classes as we implemented more design patterns from our initial class diagram in Project 5. Apart from that our design and class diagram stayed pretty consistent, since we spent significant time during our design process.

- **Summary of design changes**:

  The class diagram of Project 5 allowed us to document an MVP of our application mapping out the overall application flow and including the necessary classes, relationships and design patterns most suited to achieve our final product. Since we spent a good amount of time brainstorming this, the class diagram in Project 6 did not change much from Project 5. This was also due to the fact that we had only implemented one pattern which was simple to do (Singleton).

  Transitioning to the last phase of our project, we introduced significant changes to our class diagram as we implemented the factory, command and decorator patterns. We created separate factory classes to produce instances of letter and number blocks.

  Using a command interface, we implemented the DestroyBlock command class that encapsulates the actions of destroying a block on the correct key press. Apart from this, we added additional capabilities to our spawning blocks by decorating it at run time adding multiple hits to destroy them using the decorator pattern. This was accomplished using a combination of BlockInterface, BlockDecorator and BlockHits classes.

4. Third Party vs Original Code Statement:

A majority of our source code used in the project is our own effort. We created the individual classes, methods and attributes through our design activities from Project 5. However, we did refer to tutorials to get up to speed with OOP implementation in Python and the usage of pygame library to develop games.

Apart from this we used stackoverflow to get past the specific issues we were facing during development with pygame and flaticons to gain access to the assets. Links to all the resources we used are listed below:

- OOP in Python
- Pygame Tutorial for Beginners
- Command Pattern in Python
- Singleton Pattern in Python
- Factory Pattern in Python
- Decorator Pattern in Python
- Game Sound Effects and Music
- Game Images

- Searching through a list of objects in python
- Wait for some time in pygame
- Reset timers in pygame
- Check if music is being played in pygame
- Pygame - Toggle music
- Metaclasses in Python

5. OOAD Design Process

- Project 5 dedicated to the design phase of the semester project helped us a lot to streamline our application flow and think about the various classes and the interactions among them to achieve the desired final application.

- We felt that we could have thought a bit more deeper about the classes and interfaces that we would use for the implementation of design patterns. This could have saved time during our development phase as we had to dig a lot deeper into the implementation details during our last sprint to accomplish this.

- In retrospect, we believe that we scoped our project very well including the technologies used to construct the application. Since we had a clear design layout at the end of Project 5 and a deeper understanding of the individual tasks, we worked together effectively referring to our designs frequently to complete our project. Overall it was an enjoyable and fruitful learning experience.