POLYTECHNIC UNIVERSITY OF THE PHILIPPINES

STRATEGIC 2048 (3x3 GRID)

A Project Proposal Submitted to the Polytechnic University of the Philippines – Parañaque City Campus
Parañaque City

In Partial Requirement for the Subject
COMP 004: Discrete Structures 1

Submitted to:

KENNETH ABAJA

Submitted by:

Bartolay, Asshley Shane A.
Buenaflor, Jericka S.
Cordis, Ricca Jane D.
Dela Torre, Jane Cristal C.
Maningo, Stephanie Vianne M.
Rarugal, Norielle B.
Siervo, Dyril Justin
Soriano, Trina Mae A.
Tamayo, Ellise James B.
Taruc, Yukalhey R.

JUNE 2024

Introduction

Research by Redick & Engle, 2006; Dowker, 2005 has shown that most of the puzzle games can improve Cognitive Flexibility and Strategic Thinking. Among these, 2048, stands out as a prominent example. 2048 as a single player sliding block game which is created by Gabriele Cirulli in 2014, involves merging same value numbered tiles on a 4x4 grid to reach the coveted 2048 tile, if the players reaches 2048 without running out of legal moves, then the player won but if the player didn't reach the number 2048 and runs out of legal moves then the player loses. Its simplicity yet brain challenging scheme led to rapid reaction from gamers. Despite its popularity and merits, The original iteration of 2048 presents certain limitations due to its fixed grid size and repetitive gameplay, so to address these limitations. Our project proposes a newer version of 2048 that brings forth a more complex strategic layers by making transforming the 4x4 grid to 3x3, increasing it's difficulty and making the players puts prioritization on core strategy to win the game and puts more emphasis on importance of moves as there are limited tiles and there is a lot of chances of losing. By this proposed version, we aim to create a more competitive and challenging experience for the players, enhancing their cognitive, mathematical and logical thinking, while also keeping in mind the player's enjoyment and satisfaction.

Goals and Objectives

This project proposes making a playable and more challenging version of the popular puzzle game, 2048. The goal of the project is to serve as a learning tool for fundamental programming concepts such as game logic, user interaction, and data

structures. These are the following measurable objectives that will indicate the project's success:

- Create a visually appealing theme built particularly for the 3x3 grid to match the smaller space. This can improve the overall user experience.
- Implement different modes including Classic Modes, Timer Mode, and Move Mode.
- **Classic Mode:** Automatically save and set the highest score to encourage players to strategize and maximize their plays.
- **Timer Mode:** Record the player's fastest time to reach the target tile (2048). This encourages the players to improve their speed and efficiency.
- **Move Mode:** Display the current move count and record the player's least move count to reach the target tile (2048). This allows players to track their progress and strategize for even fewer moves in future attempts.
- Add sound effects to improve gameplay, and allow users to turn them off for a more relaxed experience.
- Create a user interface (UI) that is simple, with tile colors, move count display, best move display and visually appealing.
- Create game-over, you win conditions, and logic for pausing and restarting the game.

These objectives can help in creating a 3x3 2048 game that is not only challenging but also distinctive, accessible, and encourages strategic thinking and learning.

Project Scope

Inclusions:

Scope of work:

- Development of a 3x3 version of the popular game 2048.
- Implementation of 3 game modes: Classic Mode, Timer Mode and Move Mode.
- Implementation of features: Classic Mode has a score and high score, Timer Mode has a timer, recorded time, pause and resume button, Move Mode has move count and least move count.
- Implementation of core game mechanics using python pygame: merging tiles, movement controls(left, right, up, and down).
- Creation of a user interface (UI) that is simple, with tile colors, and visually appealing.
- Implementation of game-over, win conditions.
- Implementation of logic for pausing, quitting , and resetting the game.
- Implementation of background music and basic sound effects for tile movement and merging

Exclusions:

Out of Scope Activities:

- Development of multiplayer functionality and power up bonuses.

- Advanced feature and animations or graphic effects of the game.

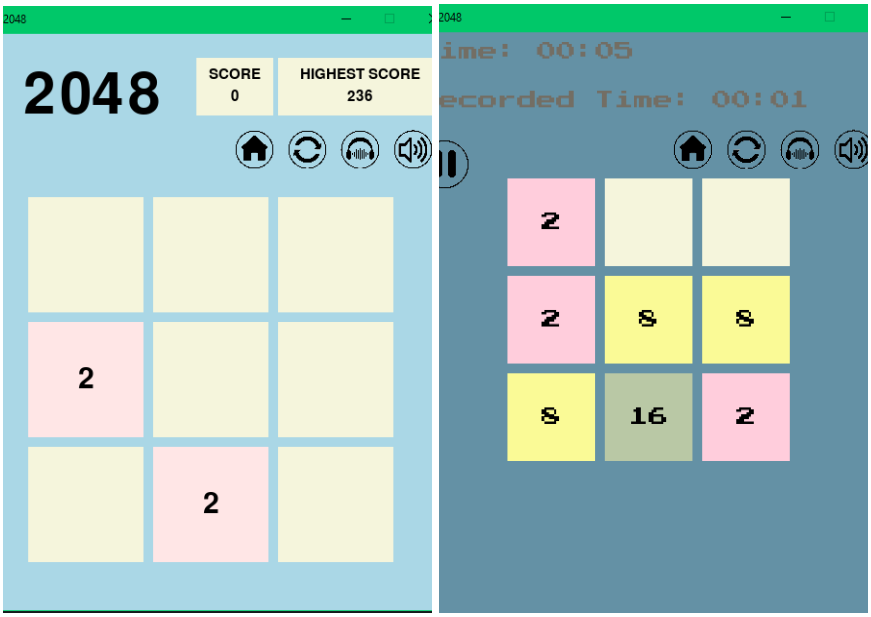
- Playable on mobile phones/ creating mobile versions of the game.

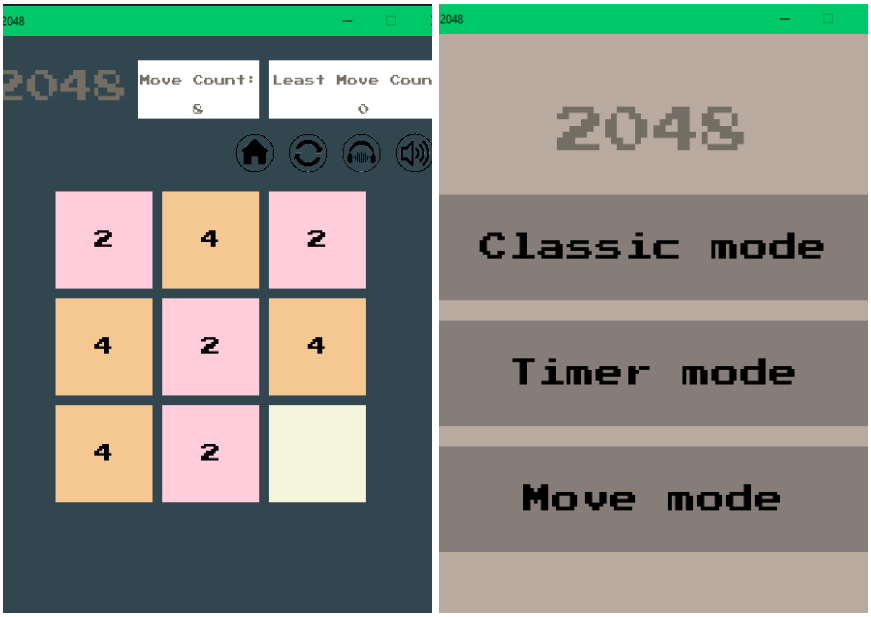
- Monetization Strategies in terms of in app purchases and virtual currency.

Limitations:

- Relying only on free resources and personal devices available for development.
- Limited knowledge and experience with game development.
- End Conditions: fixed win conditions and no moves left.
- No player engagement like limited narrative and short play sessions.

Project Design





Solution Description

Describe the proposed solution in detail. For a computer programming project, this includes:

- **Technical Specifications:**

  Languages and Frameworks

  Programming Language: Python

  Framework: Pygame for the graphical interface and user interactions

- **Features:**

Common Features Across All Modes

- **3×3 Grid:** The primary game board consists of a 3×3 grid where the game's activities occur.

- **Home Button:** This button allows players to return to the main menu or home screen from any mode.
- **Restart Button:** This button resets the current game mode, allowing players to start over without returning to the home screen.
- **Background Music:** Music that plays in the background during the game to enhance the gaming experience.
- **Sound Effects:** Auditory feedback for in-game actions, like making a move or pressing a button.

Classic Mode

- **Score:** A numerical representation of the player's performance in the game. This could be based on points accumulated through specific actions or achievements within the game.
- **Highest Score:** The highest score a player has achieved in Classic Mode. This score is saved and displayed to motivate players to beat their previous best.

Timer Mode

- **Timer:** Continuously tracks the time elapsed during the game, providing a constant time metric.
- **Recorded Time:** The time taken by the player to complete the game or reach a specific goal. This helps in tracking the speed and efficiency of the player.
- **Play Button:** Starts or resumes the timer, allowing the player to continue the game.

- **Pause Button:** Pauses the timer, allowing the player to take a break without affecting their recorded time.

Move Mode

- **Move Count:** The number of moves the player has made in the game. This helps in tracking the player's progress and efficiency.
- **Least Move Count:** The minimum number of moves the player has made to complete the game in previous attempts. This score is saved to encourage the player to beat their previous record.

- **Logic Implementation:**

CLASSIC MODE

1. **Game Initialization:**
    - Initialize a 3x3 grid with all cells set to 0.
    - Add two initial tiles with values of 2 or 4 at random positions.
2. **Tile Movement Logic:**
    - Left Movement: Traverse each row from left to right. For each non-zero tile, check if the adjacent tile is either zero (empty) or has the same value (mergeable). Shift tiles accordingly and merge if possible.
    - Right Movement: Traverse each row from right to left and apply similar logic.

- Up Movement: Traverse each column from top to bottom and apply similar logic.
- Down Movement: Traverse each column from bottom to top and apply similar logic.

3. **Random Tile Generation:**
   - After each move, identify all empty positions on the board.
   - Randomly select one of these positions and place a new tile with a value of either 2 or 4.
4. **Game Over Detection:**
   - Check if there are no empty positions and no adjacent tiles with the same value.
5. **Score Tracking:**
   - Maintain a score variable.
   - Each time two tiles merge, add the value of the merged tile to the score.
6. **User Interface:**
   - Use Pygame to render the 3x3 grid.
   - Draw tiles with their respective values.
   - Update the display after each move.
   - Show the current score on the screen.
7. **User Input Handling:**
   - Use Pygame to capture keyboard events.
   - Map arrow key inputs to corresponding tile movements.
8. **Restart Game:**
   - Reset the game board and score to initial values.

- Generate two initial random tiles.

TIMER MODE

1. **Initialize the Timer**:
    - Start a timer when the game begins.
2. **Track Game State**:
    - Continuously monitor the game state to detect when the player reaches 2048.
3. **Stop the Timer**:
    - When the player reaches 2048, stop the timer and record the time.
4. **Store Best Time**:
    - Compare the recorded time with the previous best time and update if it's a new record.
5. **Display Best Time**:
    - Show the best time on the game interface.

MOVE MODE

1. **Game State Initialization:**
    - Define and initialize variables for tracking the grid, score, current moves taken (moves taken), and the least moves (least move) to reach 2048.
2. **Grid Management Functions:**
    - Initialize a 3x3 grid with zeros.

- add_new_tile(grid): Randomly add a new tile (2 or 4) to an empty spot on the grid.
- merge(row): Merge tiles in a row according to 2048 game rules.
- Movement Functions (move_left(), move_right(), move_up(), move_down()): Implement movement logic for each direction, handling tile merging and updating the grid.

3. **Game Mechanics:**
   - check_2048_tile(grid): Check if the 2048 tile value (or higher) is present on the grid.
   - game_over(grid): Determine if the game is over by checking if no more moves are possible (no empty cells and no mergeable adjacent tiles).
4. **User Interface and Display:**
   - draw_board(screen, grid, score, moves_taken, least_moves): Draw the current game state on the screen, including the grid, score, current moves, and the least moves recorded.
5. **Main Game Loop:**
   - Start the main game loop where the game logic and user input handling occur.
   - Continuously update the screen based on user actions (arrow key presses).
   - Update the grid, check game-over conditions, and update the moves taken.
   - Update least_moves whenever the player reaches the 2048 tile with fewer moves.
6. **Event Handling:**
   - Capture and process keyboard events (arrow key presses) to move tiles on the grid.

7. **Ending the Game:**
    - When the game is over (no more moves or 2048 tile reached), display a message and allow the player to restart or quit.

Methodology

The methodology outlines the structured approach to transform the project from concept to reality. It includes a step-by-step plan covering Planning, Development, Testing, and Deployment, ensuring every phase is meticulously executed. With a clear timeline and the right tools and technologies, the methodology is designed to deliver a high-quality game efficiently and effectively. This roadmap ensures that the team stays on track, meets the goals, and exceeds expectations.

Development Phases:

1. Planning Phase

Objectives:

To ensure the successful creation of a 2048 game with unique features, it is essential to define clear objectives, gather essential requirements, identify stakeholders, and establish a comprehensive project plan.

Activities:

- Planning unique features of the game.
- Identify game logic and features for Classic Mode, Timer Mode, and Move Mode.
- Prepare initial game design and architecture.

- Study Pygame library to understand how to use it for game development.

Deliverables: Detailed game plan, requirements document, initial game design, and a clear roadmap.

2. Development Phase

Objectives:

Develop the core functionality and features of the 2048 game based on the gathered requirements. Create a user-friendly interface that enhances the gaming experience and ensures smooth and responsive game mechanics for all modes.

Activities:

- Implementing the core features and functionalities of the game including the buttons.
- Implementation of the tile movement logic of the directional buttons (Up, Down, Left, Right).
- Implementation of the game logic of the three modes
- Design and develop the user interface.
- Integrate a scoring system to track player progress in Classic Mode.
- Implementation of time recording for Timer Mode to track how fast players reach 2048.
- Implementation of move counting for Move Mode to track the number of moves til it reaches 2048.
- Implementation of win and lose conditions.
- Implementation of sound effects and background music to enhance the gameplay experience.

Deliverables: Fully functional game logic for the 3x3 grid, Implementation of Classic Mode with recording scoring system, Timer Mode with time recording functionality, Move Mode with move counting functionality and Sound effects and background music integrated into the game.

3. Testing Phase

Objectives:

It ensures that the game is enjoyable and engaging for players. Identify and fix any bugs or issues in the game. Validate that the game meets all player expectations and requirements

Activities:

- Perform thorough testing to identify and fix bugs and issues.
- Verify the accuracy of the scoring, timing, and move-counting features.
- Conduct user testing to gather feedback.
- Finalizing the system code.

Deliverables: A polished, bug-free game ready for launch.

4. Deployment Phase

Objectives:

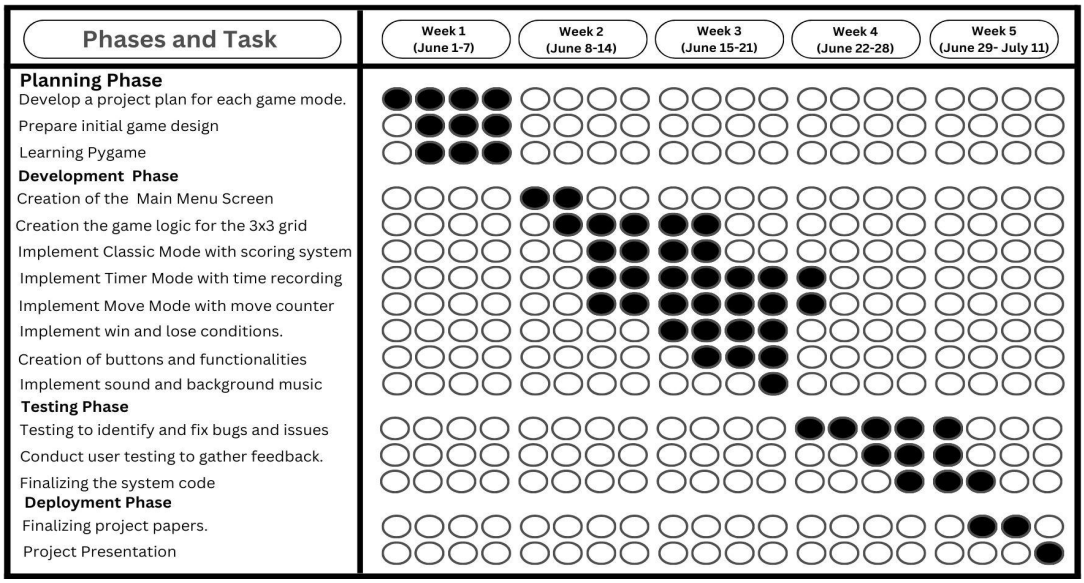
Ensure a smooth and flawless game launch so players can start playing immediately. Verify that all features are fully operational.

Activities:

- Finalizing project papers.
- Presentation of the final project.

Deliverables: A polished, project papers and a bug-free game ready for launch.

- **Timeline:**



- **Tools and Technologies:**

In developing the 2048 game, our team requires careful planning, collaboration, and the utilization of various tools and resources. To support our development process, we employ a wide range of software, hardware, and online resources.

- **Software**

To develop our game, we rely on both PyCharm and Replit IDE. These integrated development environments offer powerful tools that enhance our coding efficiency as we use Python programming language. Within these IDEs, we have installed Pygame, a popular package for game development. Pygame provides various modules that assist in creating engaging graphics and handling user input. This combination of tools and resources significantly improves our development workflow and game quality.

- **Hardware**

The team used a compatible laptop and phones to provide better technologies to develop our game. This device ensures the smoothness and efficiency performance of the game. This allows us to test the game responsiveness and functionality across these devices.

- · **Websites**

As a part of the development of the game, the team watched youtube tutorials as the basis of our code that guide us through the development process of the game. AI resources like Chatgpt and Blackbox, offered us immediate assistance to prove the effective collaborators throughout our project.

Conclusion

In conclusion, the development of an advanced 3x3 grid version of 2048 offers an additional gameplay experience. By delving into research on cognitive benefits and leveraging in significant enhancement to the trights from puzzle games, this project aims to strengthen players' strategic thinking and decision-making skills. The shift from a 4x4 grid to a 3x3 grid introduces a new level of complexity and challenge, elevating the game's attractions.

The key objectives of the project focus on implementing Classic, Timer, and Move modes, along with features like score tracking, enhancing the user interface, and maintaining the game's engaging nature by streamlining the core gameplay mechanics and incorporating user-friendly features, the project aims to create a seamless and immersive gaming experience for players.

While the project scope includes essential elements like core gameplay development and user interface design, certain exclusions such as multiplayer functionality and larger grid sizes ensure a concentrated effort on delivering a polished and refined gaming experience. These limitations are offset by the project's commitment to quality and user satisfaction. Our structured methodology encompasses detailed planning, development, testing, and deployment phases to deliver a polished and bug-free game.

The technical specifications and tools chosen for the project, including the use of Python and Pygame framework, set a solid foundation for efficient development. These choices, coupled with resources like PyCharm, Replit, YouTube tutorials, and AI tools, demonstrate a comprehensive and structured approach to game development.

This proposed 3x3 2048 goes beyond mere entertainment. It offers a rewarding experience that simultaneously challenges and stimulates the mind. The strategic depth, cognitive benefits, streamlined user experience, targeted features, and efficient development approach make this project a compelling choice for players seeking a fun and intellectually enriching gaming experience.