Minigame Trio

Student Name

Paul McGlothlin

Degree

Computer and Information Sciences

Project Advisor Name

Prof. Julie M. Henderson

Expected Graduation Date

Spring 2023

Statement of Purpose

The purpose of this program is to give users entertainment for a period of time, while also having the capacity to be enjoyed with a friend.

Problem Statement

Over the several year period of mandatory leading into heavily advised self-isolation due to the pandemic, the question of what to do with our collective time was brought up repeatedly. Many were not comfortable staying at home with nothing to do, and the unusual circumstance dragged on past the point of comfort. This program seeks to offer a chance to enjoy the moment regardless of your circumstance, allowing the user to play a small minigame of their choosing amongst three options. The games can be played alone, or with a nearby friend, allowing the user to hopefully enjoy their day just a little bit more.

Research and Background

As part of this project, I had to do some research into basic game design in order to make sure the three minigames were playable at a basic level; if the games were not fun in the first place, then the mission statement would be failed immediately. This coincided with some research into how exactly I wanted to create my envisioned product, leading me to SFML. This was the first time I was using SFML in this manner, so I had to learn the mechanics of how to utilize it.

Project Language(s), Software, and Hardware

C++ and SFML.

Requirements

|  |  |
| --- | --- |
| ID Number | 1 |
| Type | Functional |
| Description | Keyboard Control: The ability for the program to properly read input from a Keyboard Device |
| Rationale | Basic requirement for user control on a PC platform |
| Fit Criterion | User can control the program using a Keyboard input device |
| Priority | High |
| Dependencies | None |

|  |  |
| --- | --- |
| ID Number | 2 |
| Type | Functional |
| Description | Menu Handler: Code to handle the main menu of the program, allowing user navigation and selection of options and games. Also controls what music plays on the main menu. |
| Rationale | Basic part of the code structure to allow the user to move into the further, more important handlers |
| Fit Criterion | User can control and navigate the main menu |
| Priority | High |
| Dependencies | 1, 2 |

|  |  |
| --- | --- |
| ID Number | 3 |
| Type | Functional |
| Description | 1P Event Handler 1: Event handler for Game 1 while loaded into the Single Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 1 includes controlling the block spawning, and managing the score system to adapt to player moves. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers. |
| Fit Criterion | User can play the first game solo with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3 |

|  |  |
| --- | --- |
| ID Number | 4 |
| Type | Functional |
| Description | 1P Event Handler 2: Event handler for Game 2 while loaded into the Single Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 2 includes managing the score for the player based on variables. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers. |
| Fit Criterion | User can play the second game solo with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3 |

|  |  |
| --- | --- |
| ID Number | 5 |
| Type | Functional |
| Description | 1P Event Handler 3: Event handler for Game 3 while loaded into the Single Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 3 includes managing dice output, the current hand of dice for the player, and what combo strings have already been taken. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers. |
| Fit Criterion | User can play the third game solo with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3 |

|  |  |
| --- | --- |
| ID Number | 6 |
| Type | Functional |
| Description | 2P Event Handler 1: Event handler for Game 1 while loaded into the Two Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 1 includes controlling the block spawning, and managing the score system to adapt to player moves. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers. |
| Fit Criterion | User can play the first game with two players with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3, 4 |

|  |  |
| --- | --- |
| ID Number | 7 |
| Type | Functional |
| Description | 2P Event Handler 2: Event handler for Game 2 while loaded into the Two Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 2 includes managing the score for the player based on variables. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers, with optional modifiers based on settings. |
| Fit Criterion | User can play the second game with two players with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3, 5 |

|  |  |
| --- | --- |
| ID Number | 8 |
| Type | Functional |
| Description | 2P Event Handler 3: Event handler for Game 3 while loaded into the Two Player option. Handler will control the game’s mechanisms.  Unique functionality for Game 3 includes managing dice output, the current hand of dice for the player, and what combo strings have already been taken. |
| Rationale | Handlers should be unique between games and player numbers to allow for more robust modifications as needed between handlers. |
| Fit Criterion | User can play the third game with two players with no issues. |
| Priority | High |
| Dependencies | 1, 2, 3, 6 |

|  |  |
| --- | --- |
| ID Number | 9 |
| Type | Functional |
| Description | AI System: Optional Vs. Computer mode that utilizes a custom AI for games 2 and 3 to substitute a second player. |
| Rationale | Allows a single player to play the multiplayer versions of each respective game without having access to a friend. |
| Fit Criterion | AI is functional and can be used to substitute a player 2. |
| Priority | High |
| Dependencies | 7, 8, 9 |

|  |  |
| --- | --- |
| ID Number | 10 |
| Type | Look and Feel |
| Description | Readable GUI: User-friendly GUI that is readable at a glance. |
| Rationale | Allows a user to actually interact with the experience instead of relying on command line. |
| Fit Criterion | Program runs with a proper GUI that has fitting visuals. |
| Priority | High |
| Dependencies | None |

|  |  |
| --- | --- |
| ID Number | 11 |
| Type | Look and Feel |
| Description | Sound Output: Game plays background music corresponding to the events onscreen. Controlled via the invidual game and menu handlers. |
| Rationale | Proper music will enhance user immersion. |
| Fit Criterion | Program has background music during play. |
| Priority | Medium |
| Dependencies | 3, 4, 5, 6, 7, 8, 9 |
| Fit Criterion | User is able to change components of the GUI using settings |
| Priority | Low |
| Dependencies | 12, 13 |

|  |  |
| --- | --- |
| ID Number | 12 |
| Type | Look and Feel |
| Description | AI Readability: AI behaves accurately as if it were a player. |
| Rationale | Humanizing the AI will improve user immersion. |
| Fit Criterion | A user in a blind test should ideally not be able to tell if they are competing with another human or the AI. |
| Priority | Medium |
| Dependencies | 10, 11 |

|  |  |
| --- | --- |
| ID Number | 13 |
| Type | Usability |
| Description | Tutorials: Prompts in the program to teach the user about the experience. |
| Rationale | New users might need guidance to properly access the features of the program. |
| Fit Criterion | New users are able to easily get help when needed. |
| Priority | Low |
| Dependencies | 3, 12 |

|  |  |
| --- | --- |
| ID Number | 14 |
| Type | Usability |
| Description | Control Settings: User is able to alter the setup of the current control scheme, rebinding keys at will. Saves a file that keeps track of control preferences in the project folder. |
| Rationale | Users being able to pick what buttons they want to do what will increase immersion and playability |
| Fit Criterion | Users are able to change control schemes at will |
| Priority | Medium |
| Dependencies | 1, 2, 3, 12 |

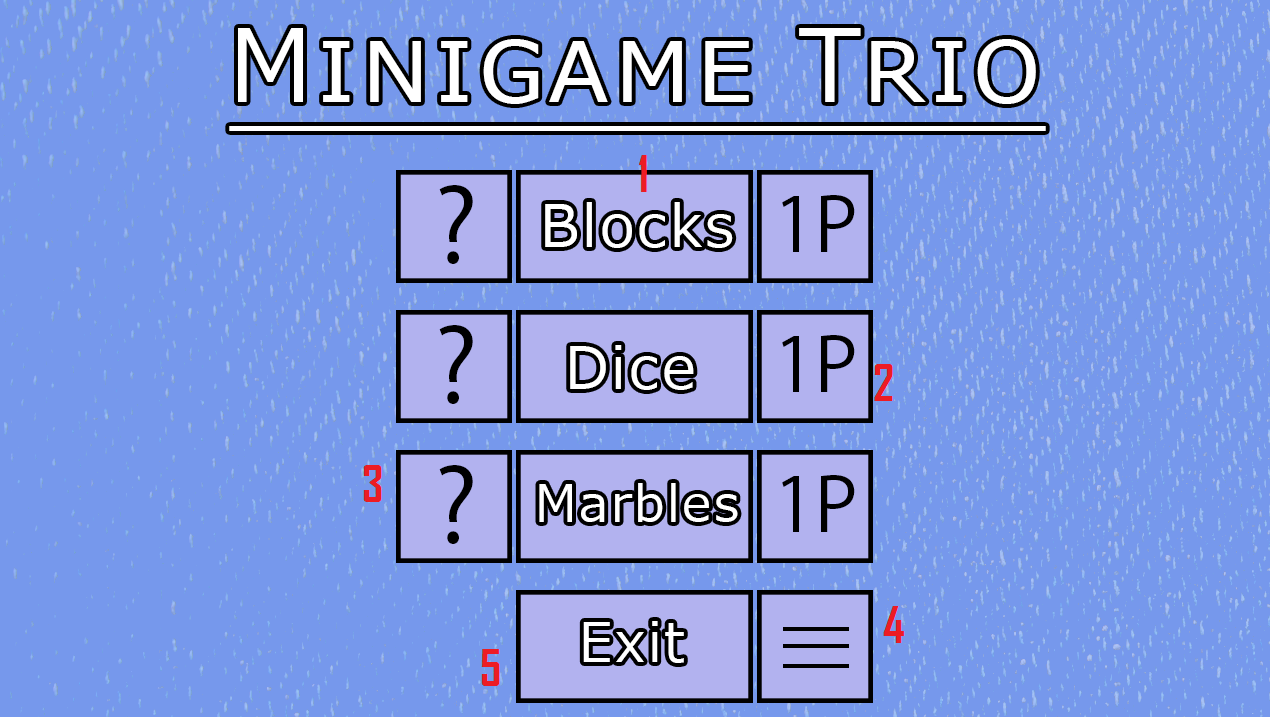
|  |  |
| --- | --- |
| ID Number | 15 |
| Type | Usability |
| Description | Game Settings: User is able to change aspects of each individual game through a settings menu. Saves a file that keeps track of game preferences in the project folder. |
| Rationale | Users being able to alter components of the games will increase user immersion and playability, specifically by making the experiences more accessible. |
| Fit Criterion | Users are able to change the settings of each individual game at will. |
| Priority | Medium |
| Dependencies | 3, 4, 5, 6, 7, 8, 9, 12 |

|  |  |
| --- | --- |
| ID Number | 16 |
| Type | Performance |
| Description | Smooth Input: User is able to control the game via their desired control method without input lag. Applies to both users in multiplayer. |
| Rationale | Makes sure the gameplay experience is enjoyable. |
| Fit Criterion | Regardless of game activity, users are able to control the experience with the expected level of fluidity and command |
| Priority | Medium |
| Dependencies | 1, 2, 13 |

|  |  |
| --- | --- |
| ID Number | 17 |
| Type | Performance |
| Description | Smooth Scaling: The first game’s difficulty scales at a reasonable pace to provide challenge to the player. |
| Rationale | Maintaining a balanced increase of difficulty over time will make the experience more enjoyable |
| Fit Criterion | Difficulty will rise according to user expectations |
| Priority | Medium |
| Dependencies | 4, 5, 6, 7, 8, 9, 12, 18 |

Project Description

<https://github.com/paulryanmc/seniorproject>

The program launches into a title screen where the user can select between a variety of options.

1. Buttons that, when clicked, lead to the three various games.

2. Buttons that toggle between one player (1P) and two player (2P) modes for the games.

3. Buttons that lead to a brief tutorial for each game.

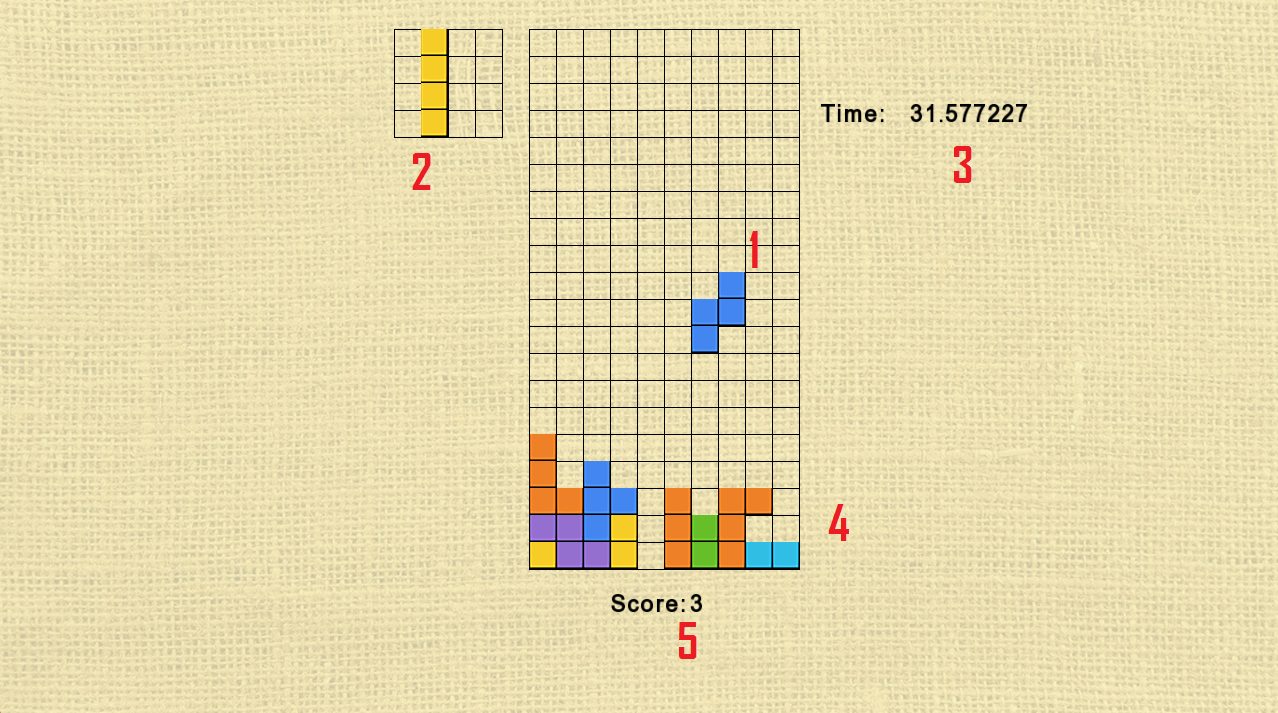
4. Button that leads to a menu with settings that can be changed.

5. Button to exit the program.

Upon clicking one of the game buttons, they will be brought to one of the following:

Blocks

A game about stacking blocks with various shapes in order to make lines.



1. The blocks that the player controls. Perpetually falling at a speed based on the game length.

2. The blocks that will appear after the player drops the current blocks.

3. The timer for the current game. Difficulty rises alongside it.

4. The field of blocks that the player has dropped. When a line is made, the blocks in that line are removed and score is given.

5. The player’s current score. Rises when a line of blocks is cleared.

Dice

A game of chance that involves rolling dice and fitting them into categories.



1. Button to roll the dice. Spawns five randomly generated dice. Every turn the player can roll up to three times.

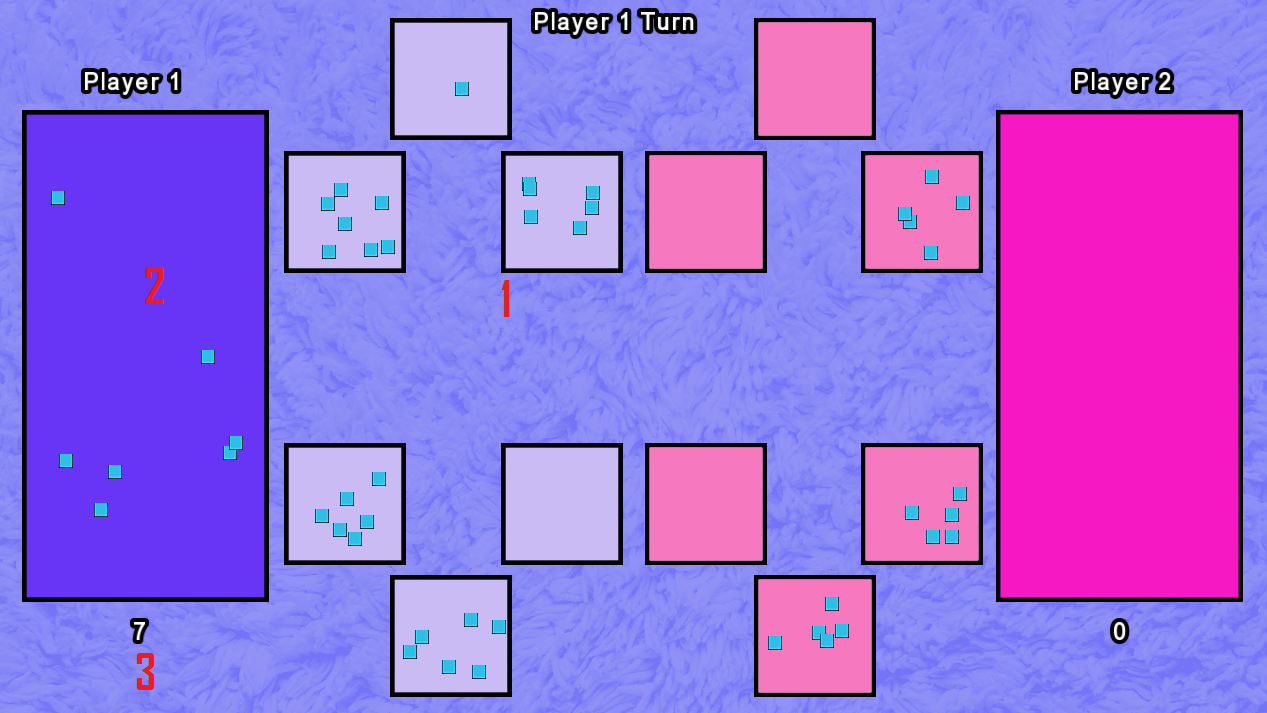
2. The field for the die. Individual dice can be clicked to freeze them, preventing rolls from changing them for that turn.

3. A label showing whose turn it currently is.

4. The categories. The player clicks one in order to fit the current die into it.

Marbles

A tactical game involving eight camps of marbles that players take turn scattering in a counter clockwise rotation. Surprisingly complex rules can make games take a long time to complete.



1. The camps for the marbles. The players take turns grabbing all of the marbles from one of their camps and placing the marbles one at a time in camps in a counter clockwise direction until they run out. Ending the turn with the last marble in hand landing on an empty camp of your opponent will let you claim all of the marbles in that camp and in the camp mirroring it on the field.

2. The goal. Players will deposit their marbles in the goal while scattering marbles if the goal is theirs, but the opponent’s goal is ignored during the rotation. Ending in your own goal gives you an immediate extra turn.

3. The score tracker. A player’s score is the total number of marbles in their goal. The game ends when one player has no more marbles on the field.

Test Plan + Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature | Test Cases | Steps | Expected Result | Actual Result |
| Title Window | Program starts | Executable is launched | Game window launches and remains open during play | Success |
| Title Window | Games start | From the title screen, any particular game is chosen and clicked | Chosen game launches as expected and begins execution | Success |
| Title Window | Games close | While playing any game, clicking the exit button in the top right closes the window and game execution | Game closes and stops all processes on the computer | Success |
| Title Window | Settings  -  Player | From the title menu, the user clicks the Player button and launches the corresponding game | The corresponding game is launched in single player or two player mode | Success |
| Title Window | Settings  -  Volume | From the title menu, the user clicks the settings button and then a volume button and is able to adjust the setting. | The volume of the program is changes to match the setting | Success |
| Blocks Game | Block moving | During play, input commands via keyboard as instructed | Blocks move correspondingly to the keyboard input | Success |
| Blocks Game | Block clearing | During play, place blocks until a straight line occurs | Line of blocks is cleared and score is added to that player | Success |
| Blocks Game | Game Over | During play, allow one player’s board to vertically fill with blocks | Game freezes and one player is declared the victor | Success |
| Dice Game | Dice Roll  -  Basic | During play, the roll button can be clicked with the left mouse button | Dice output is rerolled and displayed | Success |
| Dice Game | Dice Roll  -  Freeze | During play, the roll button can be clicked with the left mouse button after clicking some of the dice scores with the left mouse button, “freezing” them | Dice output is rerolled and displayed with the exception of the “frozen” dice | Success |
| Dice Game | Category Selection  -  Valid | During play, a category corresponding to the player is clicked with the left mouse button after rolling some valid result | That category has its score updated correspondingly to the amount given | Success |
| Dice Game | Category Selection  -  Invalid | During play, a category corresponding to the player is clicked with the left mouse button after rolling some invalid result | That category has its score updated to 0 | Success |
| Dice Game | Category Selection  -  Taken | During play, a category corresponding to the player is clicked with the left mouse button despite already being previously chosen | Nothing happens | Success |
| Dice Game | Category Selection  -  Wrong Player | During play, a category corresponding to the opposite player is clicked with the left mouse button | Nothing happens | Success |
| Dice Game | Category Selection  -  Final | During play, a category corresponding to the player is clicked with the left mouse button while being the final available category between both players | Game freezes and one player is declared the victor | Success |
| Marbles Game | Camp Movement  -  Basic | During the player’s turn, any camp belonging to the player is clicked with the left mouse button while marbles are within it | All of the marbles in that camp are delegated forward in a counter-clockwise direction | Success |
| Marbles Game | Camp Movement  -  Wrong Player | During the player’s turn, any camp belonging to the opposite player is clicked with the left mouse button | Nothing happens | Success |
| Marbles Game | Camp Movement  -  Capture | During the player’s turn, any camp belonging to the player is clicked with the left mouse button while marbles are within it, with the final marble landing in an empty camp of the player’s | All of the marbles in that camp are delegated forward in a counter-clockwise direction, with the final marble being added to the player’s score while the enemy player’s camp on the parallel side of the field is also claimed for the player’s score | Success |
| Marbles Game | Camp Movement  -  Bonus Turn | During the player’s turn, any camp belonging to the player is clicked with the left mouse button while marbles are within it, with the final marble landing in the player’s goal | All of the marbles in that camp are delegated forward in a counter-clockwise direction, and the player gets to go again | Success |
| Marbles Game | Camp Movement  -  Final Marble | During the player’s turn, any camp belonging to the player is clicked with the left mouse button while marbles are within it, with the final outcome being the player’s entire field is empty | Game freezes and one player is declared the victor | Success |

Challenges Overcome

Designing the games turned out to be harder than expected, and a lot more thought had to be put into balancing certain aspects of the gameplay in order to make it fun. During many parts of the development the games operated quite differently until a final direction was taken.

Notably, developing the block game ended up being far more complicated than initially expected, ending up taking multiple times the amount of time I had allotted for it.

Future Enhancements

In early designs of the games, I had envisioned a togglable “mission mode” that would cause variable changes to the gameplay over time in a session. The idea was scrapped due to scope creep, but it would be interesting to re-introduce.

Increasing the number of things the player can tweak, such as colorschemes.

Clearing up the visuals in general, especially in the marble game.