YOUR FREEDOM IN LEARNING

PROGRAMMING STUDIO
COMP 204
PROJECT - 2
TETRIS 2048
REPORT

Instructor: Prof. Dr. Muhittin Gökmen

DOĞA KÖKEN - 041901034
VOLKAN ÖZTOKLU - 041901037
İREM EKER - 041901022

ABSTRACT

Tetris and 2048 are two games that have maintained their popularity since their first release. These two games have algorithmic similarities. Hence, we combined these two games called "Tetris 2048" in Python using the stddraw library. This report aims to demonstrate the methods utilized to create the game, explain them in detail, and show the accomplishments that this project has provided.

Table of Contents

1.DESCRIPTION OF THE PROJECT

The goal of this project is to integrate two games: Tetris, founded in 1984 by Alexey Pajitnov, Dmitry Pavlovsky, and Vadim Gerasimov, and 2048, created by Gabriele Cirulli. The game is similar to Tetris in that the player can move and rotate 7 different shapes on the 2D playing surface, each consisting of four square tiles. The distinction is that tetrominoes have numbers on their tiles, and the number on each tile determines their color. Each approaching tetromino is made up of four tiles numbered 2 or 4 at random.

2.DESCRIPTION OF THE SOLUTION

2.1.UML Class Diagram:

2.2.Modules and Methods:

2.2.1. Tetris_2048.py

create_canvas(): This method sets the game platform size and scale.

start(): This part is where the game flows. Most of the methods are called here and the main operations are performed there.

settings_game_menu(): This method is used to create the settings menu and from this screen, the user can set game difficulty (as easy, medium, hard).

display_game_menu(): This method is used to create the main display menu and from this screen, ”Tetris 2048” can start, or if desired, settings are changed from this menu.

display_game_over(): This method is used to create the game over menu. When the game is over, or the user presses ‘E’ this method is called. On this screen, the score is displayed, and from this screen, ”Tetris 2048” can start again or if desired, settings are changed.

display_pause(): This method is used to create a pause menu.

create_tetromino(): This method is used to create random tetromino from the tetromino_types array, that includes the shapes *'I', 'O', 'Z', 'S', 'T', 'L', 'J'*.

2.2.2. game_grid.py

show_next_tetromino(): This method is used to show the next two tetrominoes from the tetromino array.
show_options(): This method is used to show shortcut keys and print to the screen.
merge(): This method is used to merge two tiles standing on top of each other. The logic of the method is to delete the upper tile and sum two tiles numbers. After that, define the sum number to the bottom tile.
print_score(): This method is used to show the scoreboard on the game screen.
display(): This method is used for displaying the game grid.
draw_grid(): This method is for drawing the lines and the cells of the game grid.
draw_boundaries(): This method is used to draw the boundaries around the game grid.

is_occupied(): This method checks whether the grid cell with given row and column indexes is occupied by a tile or empty

is_inside() ,is_inside_for_tile(): These methods are used to check whether the cell with given row and column indexes is inside the game grid or not. If it is inside, it returns True, returns False otherwise.

update_grid(): This method locks the tiles of the landed tetromino on the game grid while checking if the game is over due to having tiles above the topmost grid row. . When the game is over, the method returns True; otherwise, it returns False.

after_merge_col_drop(): After the merge, it ensures that all the tiles in the column where the transaction takes place go down one unit at a time.

tile_array_to_binary(): Preparation for labeling. So, this method is to determine all of the tile positions from "tile_matrix" and if there are tiles in the current position, determine '1' if there is a tile, and otherwise, determine '0'.

label_array(): This method defines independent tile communities using other numbers from binary arrays. All connected components have the same label at the end of this method. Also, it creates and returns a new array.

update_labeled_array(): This method updates the labeled array to fix labels.

drop_labeling_tiles(): This method is used to drop all independent tile collections into one unit and does so with the values it receives from the output of label_array().

piece_drop(): This method used to allows you to drop the tetrominoes directly to the bottom.

clear_full_lines(): This method is used to clear all row elements when all of the row elements are full.

drop_tiles_tetris() This method is used to drop the tiles in the rows above the entered row value by one unit.

find_all_tiles_position(): This method is written to find the position of all the tiles in the array.

2.2.3. tetromino.py

draw _next_tetromino(): This method allows the upcoming tetrominoes to be displayed at the bottom right of the screen.

rotate_tetromino(): This method serves to rotate the tetrominoes ninety degrees clockwise or ninety degrees counterclockwise

can_be_rotate(): This method is for checking if the current tetromino can be rotated inside the game grid. If it can be rotated, returns True, returns False otherwise.

2.2.4. tile.py

tile_value_for_merge(): This method creates the tile numbers, which are *4, 8 ,16 ,32 ,64 ,128 ,256 ,512 ,1024 ,2048* , and the colors of the tiles for merge.

draw(): This method is for drawing the tile.

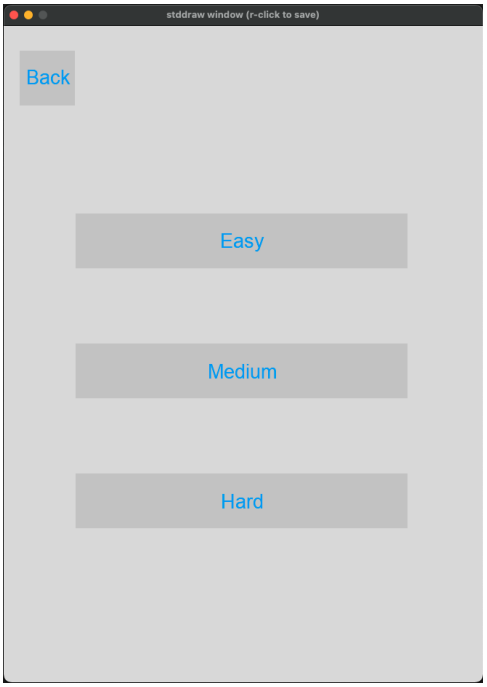
2.2.5. point.py

A class for modeling points as a location in 2D space.

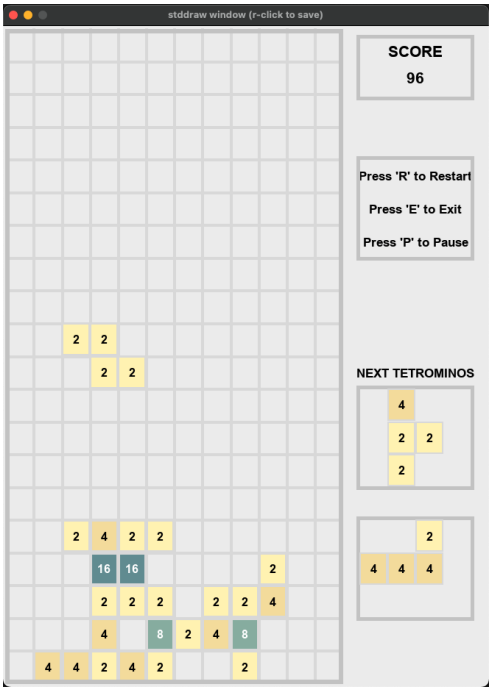
EXAMPLES OF OUTPUT:

Game Menu

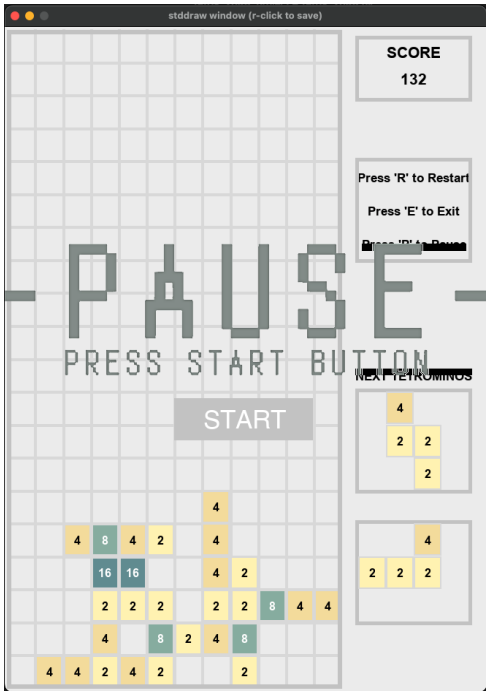
Game Settings Menu



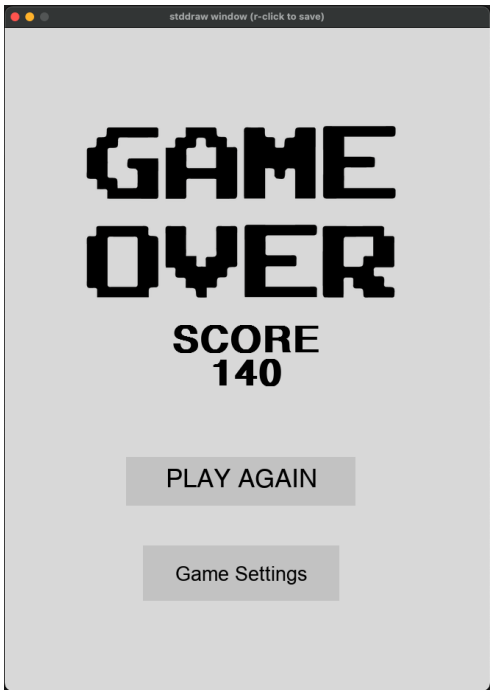
Game Grid



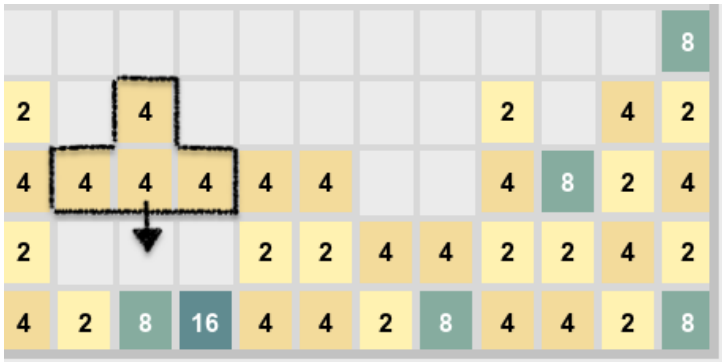
Pause Menu

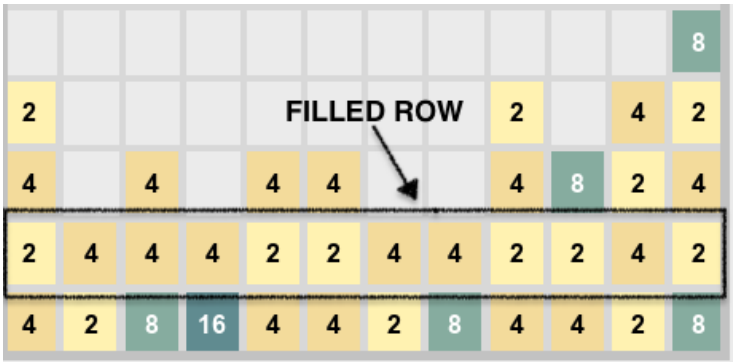


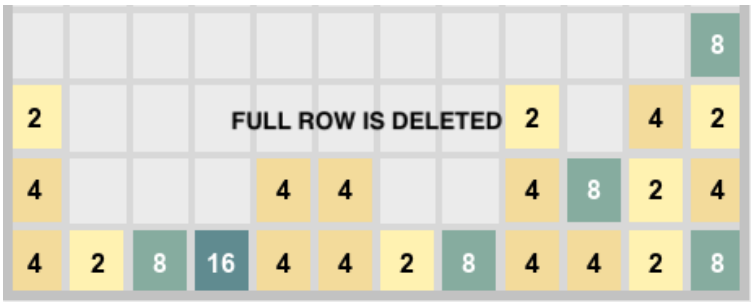
Game Over Menu



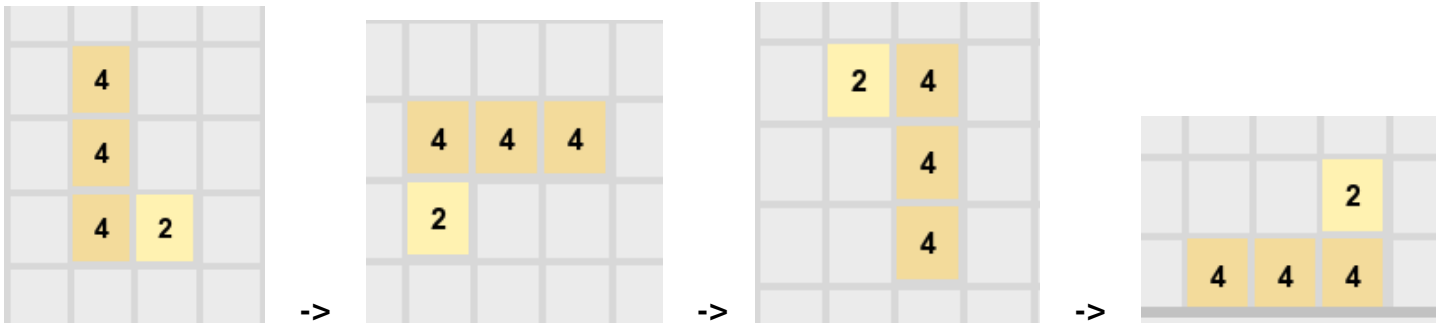
Clearing a full row:



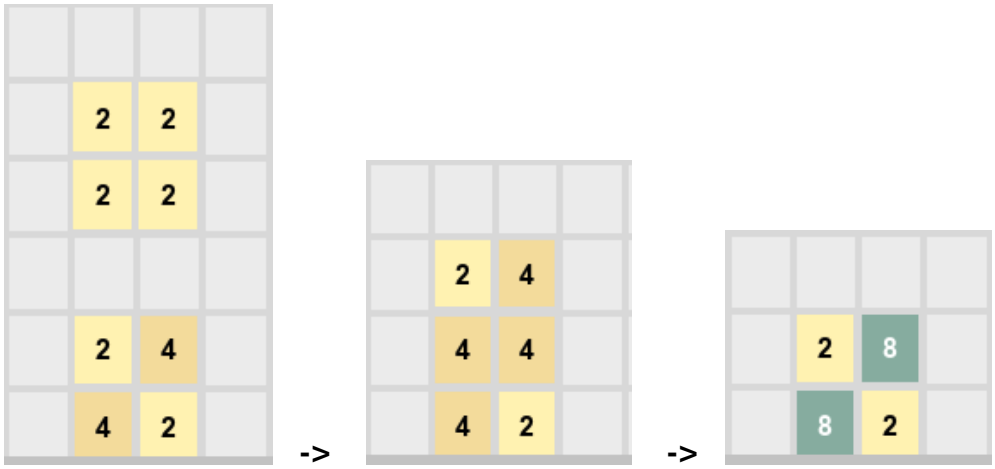




Rotation of a tetromino:



Merging example:



ACHIEVEMENTS

With this project, we learned how to proceed with in-group code projects. By using the Stddraw library in detail, we learned how to use the library better. Although a friend of ours could not use GitHub due to technical problems, we had experience using GitHub together. We have a better understanding of object-oriented coding in Python. Apart from the object-oriented part, we learned more about writing code in Python.

REFERENCES

Weber, X. (2022, January 25). Implementing a connected component labeling algorithm from scratch. Medium. Retrieved April 9, 2022, from https://towardsdatascience.com/implementing-a-connected-component-labeling-algorithm-from-scratch-94e1636554f

Gökmen, M. (2022). Project 2: Tetris 2048. Lecture.

Bakibayev, T. (2020, December 30). How to write tetris in python. Medium. Retrieved April 9, 2022, from https://levelup.gitconnected.com/writing-tetris-in-python-2a16bddb5318

2048 game in Python. GeeksforGeeks. (2022, February 21). Retrieved April 9, 2022, from https://www.geeksforgeeks.org/2048-game-in-python/

Python 2048 game - create 2048 game in 3 minutes. Project Gurukul. (2021, September 11). Retrieved April 9, 2022, from https://projectgurukul.org/python-2048-game/