APCS1 Final Project Proposal

BrownSquad — Period 5
Sachal Malick, Shantanu Jha
January 10, 2016

Overview

For our AP Computer Science Semester 1 final project, we have decided to make an ASCII version of the classic game of Tetris. We will create multiple versions which the user can choose between at their discretion. Our game will include the original Tetris game represented by ASCII characters and colors. We anticipate that user input and possible multithreading will be the most difficult aspect of this project. Although we will have to learn and utilize some outside information, we plan to implement all of the ideas that we learned in class.

1 Extending the Classics

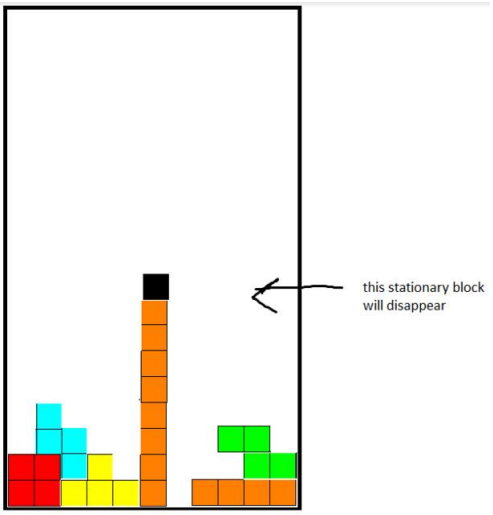
Our game will be adapted from Tetris, a classic game released in 1984. We will recreate the original game using only ASCII characters. We will use ASCII characters and different colors to recreate the game. The main challenge of this project will once again be real-time user input processing.

After fully recreating the basic game of Tetris we will extend the game by adding several features. These features include: maps that the user can start with, possible obstacles in the path of a falling block piece, added user game functionality, and more. We will provide the user to choose what type of tetris game they wish to play. We will include various maps that can be chosen randomly or by the user. We will include obstacle blocks which will be fixed to the screen and prevent pieces from falling until they are cleared (disappear). We also plan to include different levels of difficulty from which the user can choose. The chosen level of difficulty will determine how the obstacle blocks can be cleared. We will elaborate on this later in the proposal.

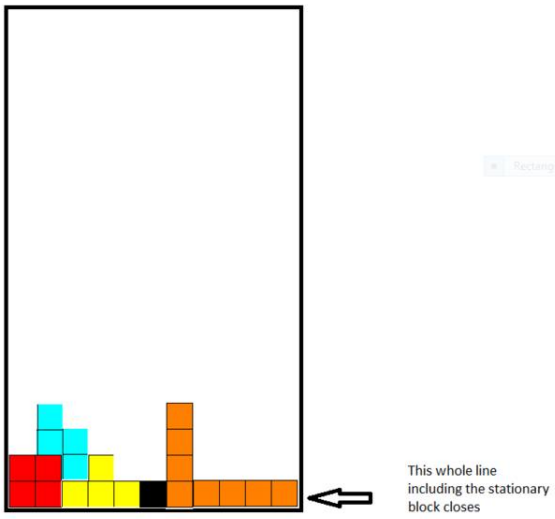
In our project, we will include all the major ideas that we learned in our AP Computer Science class this semester. We will implement a useful class hierarchy, which will include inheritance, implementation, and abstract classes, that will help organize our code, and our work flow. We will use sorting to help determine the shapes that drop for our user.

2 Difficulty Levels

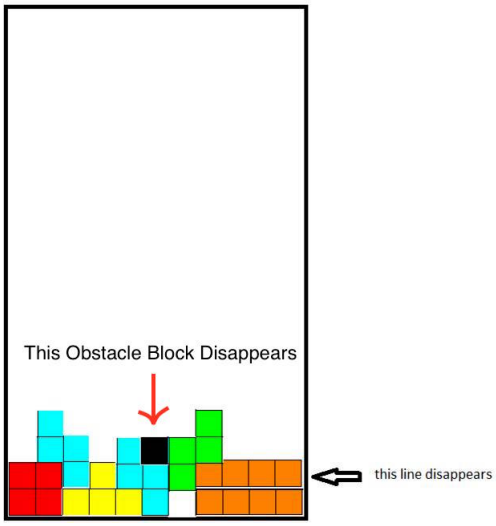
Difficulty levels will determine the method by which the obstacles block, if present, will be cleared. The easiest difficulty will allow obstacle blocks to be cleared if any stationary block is directly under it, or if the line with the stationary block is completely filled. The medium difficulty will allow obstacle block to be cleared if the line under the obstacle block is completely filled, or if the line with the obstacle block is completely filled. Finally, the hardest difficulty will only allow obstacle blocks to be cleared by completely filling the line below the obstacle block.



This method of obstacle clearing is only valid for easy difficulty.



This method of obstacle clearing is valid for easy, and medium difficulties.



This method of obstacle clearing is valid for easy, medium and hard difficulties.

3 Features

Main Features:

- Original Tetris Game
- Real time user input
- Point System
- Line clearing, and pieces falling

Additional Features:

- maps - user chooses or random
- obstacle blocks
- levels of difficulty
- size of tetris field
- other features maybe?

4 Development Tools

Some tools we will use in our website:

- Github
- Java
  - JCurses
  - Multithreading

5 Roadmap - Stages of Development

1. Stage 1: Stage 1 will be creating the foundation for the game. This will include:
    - The random shape generator that will create a shape that will appear at the top of the window and begin falling.
    - The system that allows the user to change the orientation and position of the shape. (this will probably require the scanner class so the user will have to press enter after typing every key)
    - the system that stacks the shapes once they reach the bottom
    - scoring
    - recognizing when the game is over
2. Stage 2 developments:
    - Allow key press detection so the user does not have to press enter after hitting a key, we will use multithreading + jcurses to accomplish this.
    - Different maps: A map is like a regular blank tetris window only it has certain fixed shapes on it. We have different implementations for how we will handle the fixed blocks
        - Implementation 1: Once the row containing the fixed block is filled everything including the obstacle block on that row will disappear.
        - Implementation 2: Once the row directly underneath the obstacle, the fixed block disappears.
        - Implementation 3: Once there is a stationary block right under the obstacle block, the obstacle block will disappear.
3. Stage 3 developments:
    - Levels of difficulty: For each instant in the game, different blocks will be harder to work with than others. Will have progressive difficulty so at the start the game will supply shapes that are easy for the user to work with, and then over time the user will supply shapes that are harder for the user to work with. We can sort each shape based on ascending difficulty with different sorting methods.