Cubiks Project Specification

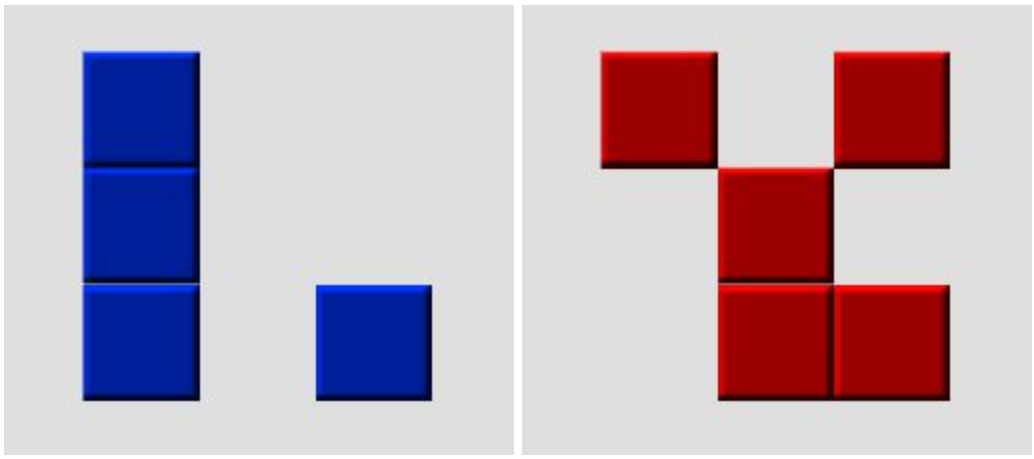
Alex Kastanek (akastanek@cox.net)
Bryce Monaco (bmonaco@nevada.unr.edu)

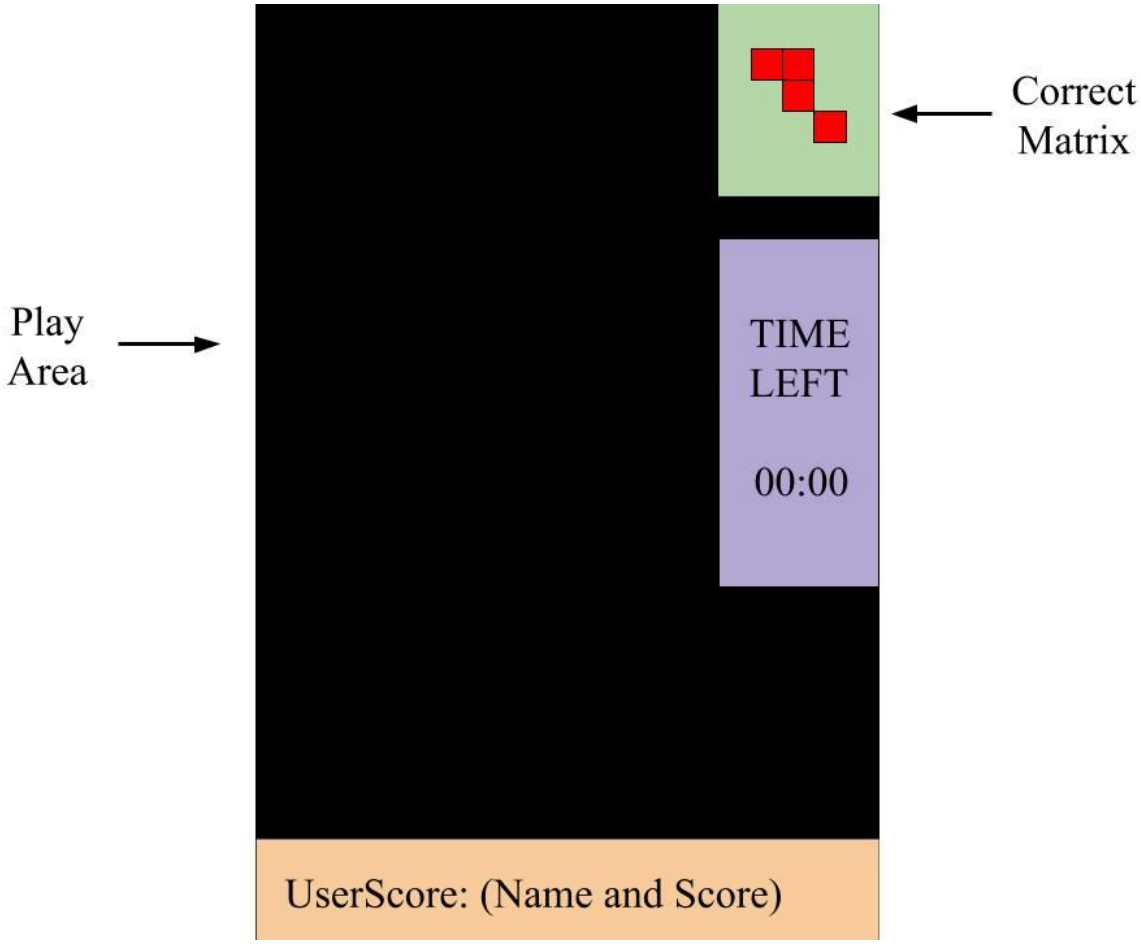
Overview

We propose to build a Tetris-like timed matching game that has the same feel and style of Tetris but a completely different purpose. It is a one-player game in which different, randomly generated patterns of 3x3 matrices of squares fall from the top of the screen. On the right side of the screen is a single 3x3 matrix of squares. The player has to match the correct matrix falling from the top of the screen to the one displayed on the right. The player is prompted for a username at the beginning of the game, and his or her high score is saved to that name. This game will be made in Unity with the C# language and will be available on the Android mobile phone platform. Since the Android OS is similar to that of Linux, it will be simple to run on ECC machines as well.

Screenshot/Mockup

Below is an example of the 3x3 matrix described in the Overview. It was produced using a random number generator within the constraints of a 2-dimensional array in Unity.





Features and Schedule Table

Below is a table that displays who is in charge of doing what and when. Also displayed is how much that portion of the project is worth in terms of percentage. The "*" indicates what features will be able to be presented at the project prototype demonstration on November 30.

Feature	**Implementor**	**%**	**Due**	**Grade**
1. Random 3x3 Matrix Generator*	Bryce	15%	11/24/15	
2. Guided User Interface*	Alex	10%	11/27/15	
3. Touchscreen Implementation and Detectability*	Bryce	10%	11/28/15	
4. Matching with Correct Matrix*	Alex	15%	11/30/15	
5. Scoring System	Bryce	15%	12/02/15	

6. Username and High Score	Alex	15%	12/04/15	
7. Timer and Testing	Bryce	10%	12/05/15	
8. Testing	Alex	10%	12/06/15	
Project Release		100%		

Feature Description

1. The first thing we want to get done is the random matrix generator which will generate a 3x3 matrix of squares. The color and assortment of the squares is where the generator comes in. Once we have this implemented, the overall concept behind our game is done.
2. Next, we want to get a Guided User Interface implemented. This means we want the game to be able to function and able to be operated with user commands. This means that we would need 2D art on the screen to symbolize buttons, information, and stats. Simply the art is all we need for this checkpoint.
3. Once we have GUI, we need it to function. Making the game work with the buttons we've created is essentially what this checkpoint is.
4. After we have our GUI created and functioning properly, we need to be sure that when the user taps on the right matrix, it detects that this was a correct match. The same goes for when the user taps on the wrong matrix.
5. The scoring system is the next step. When the player touches the correct matrix, the system needs to add 100 points to the user's score.
6. Additionally, we want the user to be able to give him/herself a name. When the user gets a score that is higher than the original ones, we want that to be stored to that username.
7. Finally, we need to give our game a challenge and a consequence. The timer will act as both of those things. The user will have a set amount of time to get as many matches as he or she can correct. We will also be using the period for testing and making sure our game works on ECC machines.
8. Finally, the second stage of testing will be testing for minor bugs before release.

Sample Input and Output

The drawing above provides a general idea of what the game is going to look like. The user will simply be prompted for their username when starting the game and will use the left mouse button to choose which matrix is the correct one.

Running the Application

The user simply has to double click Cubix.exe if on a desktop or click once on the Cubix application if on the Android mobile device.