The members of your team: Kenneth Harlley kdh62, Julia Trahan jat329, Aidan Cutie awc82

Meeting Times: Tuesday, Thursday (~430); Saturday (Whenever).

System Proposal: The core goal of our project is to build a functional version of tetris in Ocaml which is played through the terminal. This includes the core gameplay mechanic of ocaml (creating rows, tracking score, randomly generating blocks, spinning blocks).

- creating rows,
- tracking score,
- randomly generating blocks,
- spinning blocks

The design for the game state would be at its most basic form a two dimensional array of values of either 1 or 0 tied together to represent the different shapes. Each time the game state updates these would be moved one down, with each cycle checking for rows, or collision while part of the shape is not visible, resulting in the player losing the game.

MS1

<u>Developing the Game State structure:</u>

Board! Make her exist.

Satisfactory: the board exists in black and white and is drawn properly

Good: titles and such

Excellent: colors! Very exciting

Shapes

Satisfactory: 3 Shapes, Moving shapes

Good: 5 Shapes

Excellent: All Shapes, Make new shapes (Lots of extra time)

MS2

Player Input.

Satisfactory: able to drop pieces, shift left and right

Good: able to rotate pieces

Excellent: Tetris battle?

Collisions/ Scoring!

Satisfactory: blocks accumulate, Clearing Rows!

Good: Score

Excellent: Tspins, Powerup, a bonus row is cleared for every x cleared rows (?)

Modules Game State: This will contain update(), which will read the player input (or lack of) and update the state accordingly. , Shapes:contain the different shapes, as well as rotate(), drop()

Data: Current game state (board, score, etc.)?? || Storage & Communication - keys to interact || Data structures - Arrays, Queue (for upcoming blocks?)

Third party libraries: We currently don't plan to use any

Testing: We will test our game at each step. So first testing for proper board representation of a blank board, testing drawing shapes. Then testing for downward movement, then collision, player control, clearing lines, then finally scoring.