TETRIS 2.0

Your dream assignment. Trust us, you will love this.

Introduction

Tetris game is the childhood memory of many, but not Johnny's. He has been so focused on programming that he has no friends to play a 2v2 tetris battle with. He is sad, very sad, to the extent of losing his interest in programming. You, as a computer science student, need to bring this poor soul's passion back, by showing him that with good programming skills, he can make one tetris game of his own, modified and tailored to be played even without any companion!

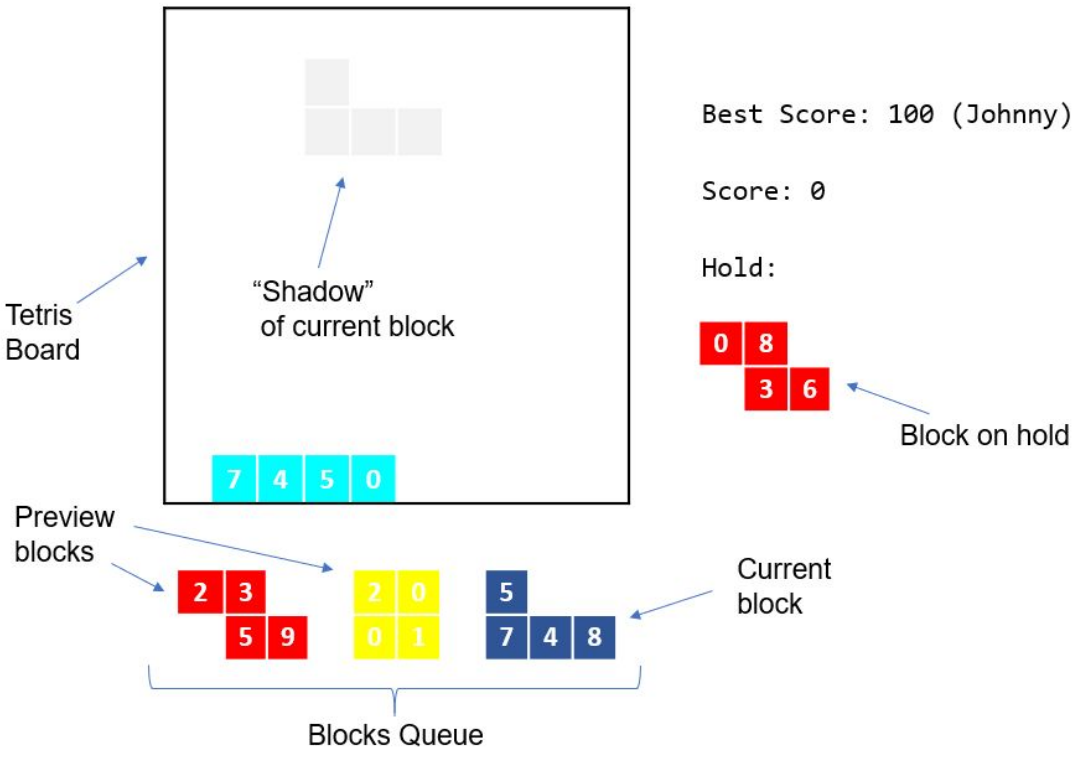
Imagine, Tetris blocks are no longer falling from the top because it's boring and annoying as hell. In Tetris 2.0, Tetris blocks can be placed by yourself anywhere you want on the Tetris board (of course it must be able to fit in and not overlapped), without any time constraint and gravity. No more pressure of Tetris blocks falling! The scoring rules are about the same but it's more interesting this time - you get points for eliminating not only fully-filled horizontal rows but also vertical columns as well if and only if the numbers inside the squares of the row or column sums up to even. Extra points will be awarded for combos! Each turn, you will have to place a random-generated Tetris block in the empty spaces on the Tetris board. When you cannot place the block on the board anymore, it's game over and the score will be recorded.

Problem statement

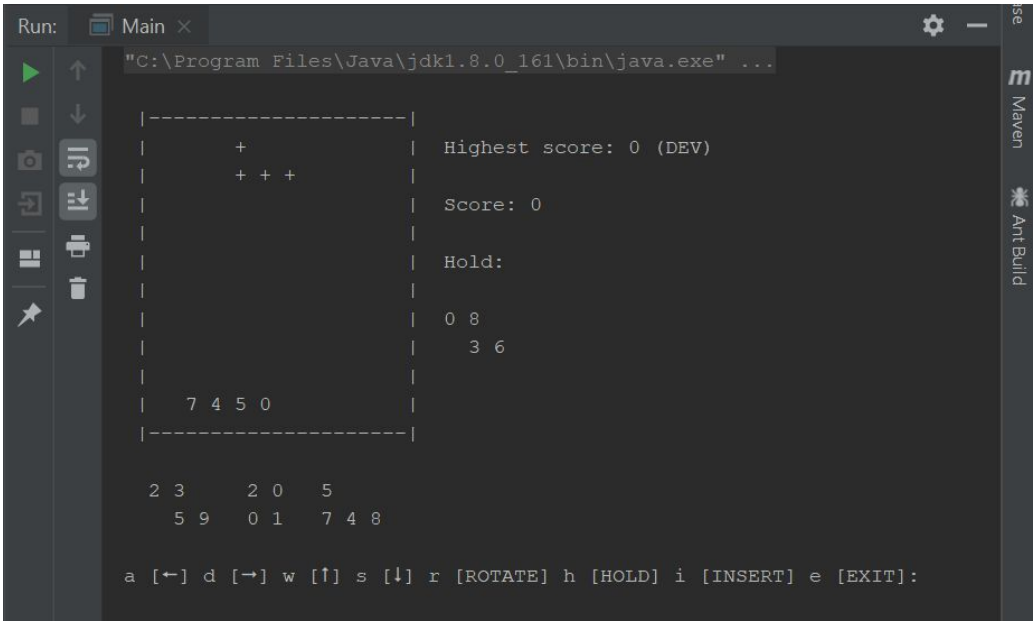
You are required to write a CLI program in JAVA to simulate the Tetris game based on the game specifications below. Programming languages other than JAVA will NOT be accepted. This assignment will test you on basic programming concepts like basic data types, if-else, loop, arithmetic, string manipulation, standard IO, file IO, OOP and logical thinking. Apply everything you've learnt in the class. Everything that is taught is sufficient to yield you a good grade. You are free to apply any advanced programming knowledge as long as you can impress us. Plagiarism is strictly PROHIBITED. You will be penalised if any of your codes is found online.

Game specifications

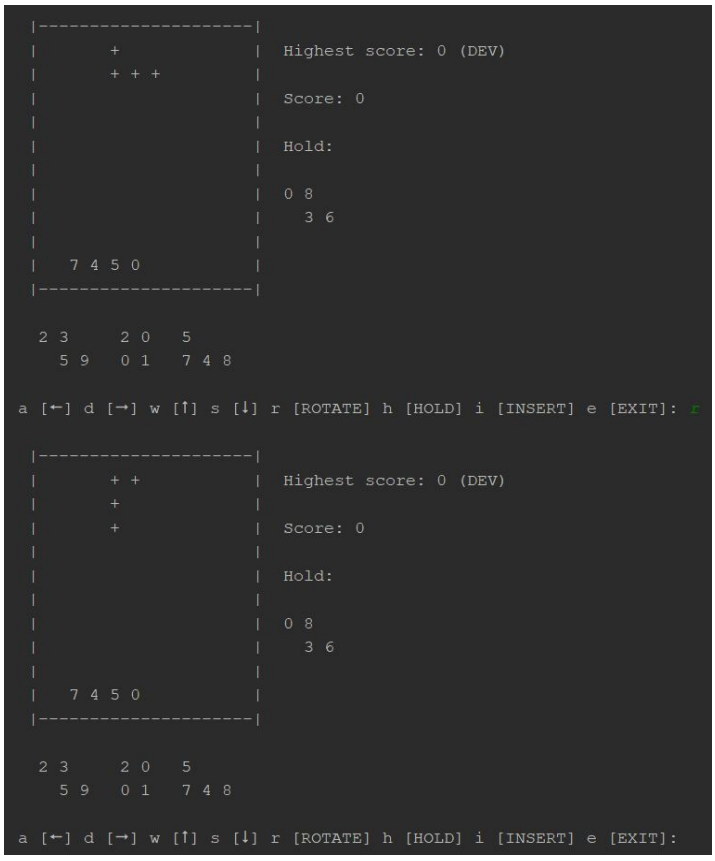
Game interface:



Idea Sketch



CLI Implementation



Rotate block

```
 |--------------------|
 | + +                |  Highest score: 0 (DEV)
 | +                  |
 | +                  |  Score: 0
 |                    |
 |                    |  Hold:
 |                    |
 |                    |  0 8
 |                    |    3 6
 |                    |
 |   7 4 5 0          |
 |--------------------|

  2 3     2 0   5
    5 9   0 1   7 4 8

a [←] d [→] w [↑] s [↓] r [ROTATE] h [HOLD] i [INSERT] e [EXIT]: i

 |--------------------|
 | 7 5      + +       |  Highest score: 0 (DEV)
 | 4        + +       |
 | 8                  |  Score: 0
 |                    |
 |                    |  Hold:
 |                    |
 |                    |  0 8
 |                    |    3 6
 |                    |
 |   7 4 5 0          |
 |--------------------|

    0     2 3     2 0
  1 4 7     5 9   0 1

a [←] d [→] w [↑] s [↓] r [ROTATE] h [HOLD] i [INSERT] e [EXIT]:
```

Insert block

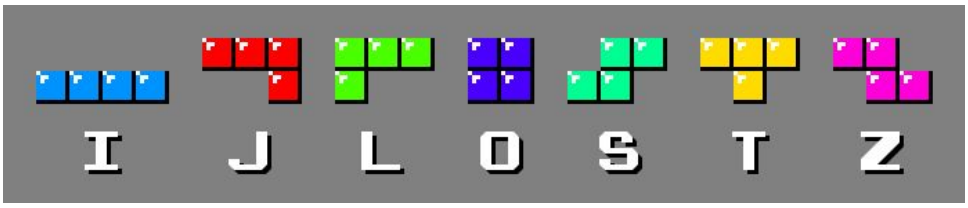
```
 |--------------------|
 | 7 5      + +       |  Highest score: 0 (DEV)
 | 4        + +       |
 | 8                  |  Score: 0
 |                    |
 |                    |  Hold:
 |                    |
 |                    |  0 8
 |                    |    3 6
 |                    |
 |   7 4 5 0          |
 |--------------------|

    0     2 3     2 0
  1 4 7     5 9   0 1

a [←] d [→] w [↑] s [↓] r [ROTATE] h [HOLD] i [INSERT] e [EXIT]: h

 |--------------------|
 | 7 5      + +       |  Highest score: 0 (DEV)
 | 4          + +     |
 | 8                  |  Score: 0
 |                    |
 |                    |  Hold:
 |                    |
 |                    |  2 0
 |                    |  0 1
 |                    |
 |   7 4 5 0          |
 |--------------------|

    0     2 3     0 8
  1 4 7     5 9     3 6

a [←] d [→] w [↑] s [↓] r [ROTATE] h [HOLD] i [INSERT] e [EXIT]:
```

Hold block

Game rules:

- Player is free to place the random-generated Tetris block anywhere on the Tetris board as long as there's space to fit in, strictly no overlapping of blocks.
- The Tetris block that is placed should stay where it is, it won't fall if there are empty spaces beneath it, it just stay "afloat", unlike the conventional Tetris game, it has no "gravity".
- Player is free to rotate the Tetris block.
- Each round, the player must use the first Tetris block (current block) in the blocks queue or put it on hold.
- If a player decides to put the current Tetris block on hold and there's no blocks on hold, the current will simply be put on hold and a new block will be generated and added to the end of the queue.
- If previously there's a Tetris block that is already on hold and the player wish to put the current one on hold, the current block simply switches with the one on hold.
- To eliminate a row or column:
    - The row or column must be fully filled AND
    - The sum of the number associated with each square of the fully filled row or column must be even
- Scoring:
    - For each row or column eliminated, one point will be given
    - A combo performed will yield additional points, i.e. eliminating one or more row or column after each subsequent insertion of block, including eliminating two or more rows in one insertion
    - new_score = previous_score + current_combo_number
    - You may use your own creativity to create a more sophisticated scoring system
- Once the player managed to place the first Tetris block in the Tetris board, a new Tetris block will be generated and added to the end of the queue.
- The blocks queue will always have 3 Tetris blocks, 1 current and 2 previews (Only the current one will need to be used or put on hold, the other two only serve as previews so that the player can plan for his moves)
- The game ends when the player decides to quit the game or the Tetris block can't fit in the Tetris board anymore.

Tetris Blocks (Tetrominoes):



Here's a list of classic Tetris blocks or Tetrominoes. Each block is made up of four squares that are connected orthogonally. In our very own Tetris 2.0, we will be spicing up these boring blocks. As mentioned in the game rules section, to eliminate a row or column, it must first be fully-filled with Tetris blocks and the numbers associated with each square in that particular fully-filled row or column must sum up to an even number. Where can we find those numbers? Of course it's within the squares of the blocks!

The numbers must range from 0 to 9. It should be randomly generated. For Tetris blocks of the same kind (same shape), the four numbers within it must not be the same.

Basic requirement

High score system:

- The home page of the game will display the scoreboard for top five high scores and the player names. (Consider prompting the player to enter his name before starting)
- You may use a simple file system to store each high score so that you won't lose it when the player quits.

Game interface (refer to the game interface section in game specifications):

- Tetris board (a 10x10 grid)
- High score to beat
- Current score obtained
- Blocks queue (1 current block + 2 preview blocks)
- Hold block

Tetris blocks:

- The Tetris blocks in the blocks queue must be randomly generated, including the number associated with the squares. (refer to the Tetris Blocks section in game specifications)
  Tip: create a TetrisBlock class.

Block "shadow" feature:

- A "shadow" of the current block, denoted by "+" sign, must be shown on the Tetris board so that the player knows exactly where the current block is going to be inserted.
- The shadow must move according to the direction inputted by the player and it can be moved within the Tetris board only, if the player tries to move the shadow out of the board, the shadow shall not be moved. (the player should be informed if they are doing so)

Block insertion feature:

- When a player decides to insert the block at the location previewed by the "shadow", the shadow which is previously denoted by "+" must be changed to the numbers within the squares of the blocks to indicate that the block has been inserted permanently until it is eliminated.
- A block can only be placed within the Tetris board and can NEVER be placed on top of another block that is placed, i.e. no overlapping of blocks

Block rotation feature:

- When a player decides to rotate the current block, the block "shadow" must be rotated as well (clockwise or anti-clockwise direction)

Block holding feature:

- When the player decides to hold the current block, the block must be properly moved to the “hold” area. (refer to the game rules section for the details of holding a block)

Rows & Columns elimination feature:

- Refer to the “To eliminate a row or column:” section above
- Scoring method mentioned above will be applied after each elimination

Extra features

- A fancy JAVA desktop application with fancy GUI, make it colourful please.
- Having Tetris board of different shapes, Octagon maybe?
- Make it multiplayer to let Johnny make friends using it:
    - There could be punishment to the opponent after performing a combo, i.e. generating a block at random location which could not be eliminated by the opponent
    - Crazy idea: use socket programming to play it on different devices
- Different game mode:
    - Time attack:
        - Implement a game loop to achieve this, update the board in real-time
        - After a certain time, the bottom row will be locked and the whole board will be moved upwards by one row. Meaning that the top row will disappear.
        - More and more rows will be locked after time and eventually the player will lose
    - OR any additional interesting game mode
- Any feature that is fancy enough to impress us.
- We DO NOT accept the conventional falling Tetris blocks game as an extra feature. Please do not implement it.

CRAZIEST FEATURE:

- A bot that can versus human player, the bot must be smart enough to beat human (Ai?)
- Randomly inserting Tetris blocks is not considered “smart”, so don’t try to act smart
- Implement the punishment features of multiplayer mode
- Remember, Johnny has no friends, your bot will be his best friend
- You are 100% guaranteed to get an **A+** for successfully implementing this feature, just like Johnny guaranteed to get a friend :)

Generous tips and comments

- Try to implement the CLI version before jumping right into GUI. CLI has its own challenges because you will be mainly dealing with strings and characters to be printed out on the console. There are lots of valuable concepts and tricks that can only be learnt by implementing CLI but not GUI. Completing the CLI version can also prepare you with the game logic you need for the GUI because game logic won't change for both versions! When you are ready with the game logic codes, the next steps towards GUI are just preparing the graphics and wire it up.
- Do you think we can improve the way we provide inputs? Have you ever wondered how other games consume direction inputs via the arrow keys instead of typing in letters or characters (which sounds low-level and not fun). Well fret not, we have a tip for you! Try to Google something called “Java KeyListener”.
- If you find that it is annoying and not that sleek for the console to constantly print the game interface again and again after each move or input that is made, why don’t you clear the console? Here are some starter codes for you:

```
public static void clearScreen() {
    try {
        new ProcessBuilder( ...command: "cmd", "/c", "cls").inheritIO().start().waitFor();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (InterruptedException ex) {
        Logger.getLogger(Main.class
                .getName()).log(Level.SEVERE, msg: null, ex);
    }
}
```

  Make console game cool again!

- Last but not least, don’t limit your potential. Have fun and be creative! You will eventually sit down and be impressed by your own work! All the best!

With love, from Philemon and Yong Keat