CSCI 1130A/B Introduction to Computing Using Java 2021-2022 First Term
Department of Computer Science and Engineering
The Chinese University of Hong Kong

Due date: 13 Dec 2021 (Mon) **Assignment 6: BoardGame** Full mark: 100
Expected normal time spent: 10 hours

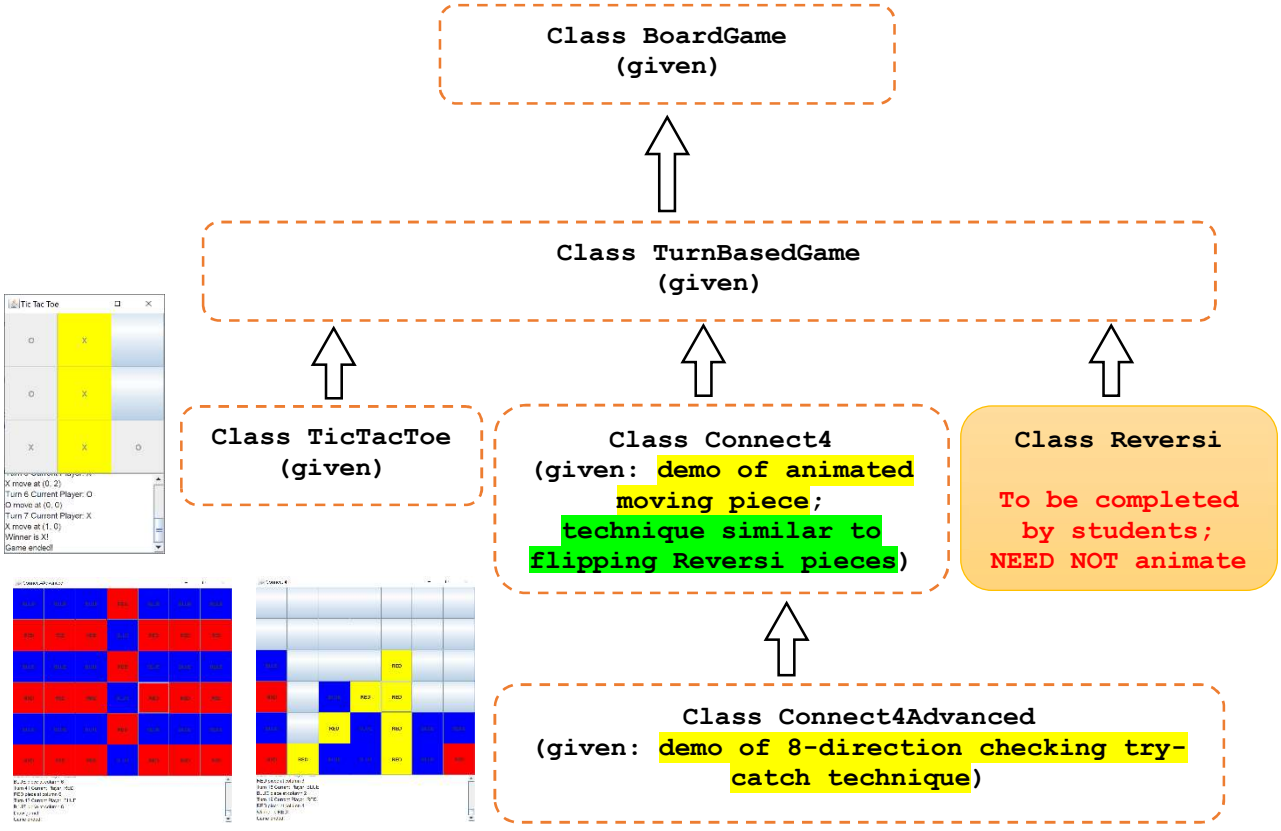
Aims: 1. Practice defining a subclass, as well as overriding methods.
2. Practice creating a GUI application.

We are going to build a Reversi BoardGame with GUI. There are some given classes and methods for your comprehension, reference and use.

Procedure:

1. Create a new NetBeans project **`BoardGame`** with a new class **`boardgame.Reversi`**. Download and copy some given classes into the package folder **`boardgame`**.
You just have to complete the class **`Reversi`** which is a subclass of **`TurnBasedGame`**.

2. Below is the given inheritance hierarchy. All classes are "main" classes, i.e., can be run.



3. **First of all, leave your newly created class Reversi alone. Run all other given classes one-by-one. Try to enjoy and understand their behaviours and the coordinates systems.**

4. **Read the comments and code** in the given source code files. **`TicTacToe`** is a good reference for you to begin with.

5. Remember that due to inheritance, subclasses can enjoy most of the features, fields and methods of their super classes. Subclasses may also override some of the inherited methods. Subclasses may also introduce new fields and new methods.

6. The board size of Reversi (also known as Othello or Black-White Chess) is always **8 columns (xCount) by 8 rows (yCount)**. The two players, **BLACK and WHITE**, place one piece at a time on an initial board (as shown below) in turns. **BLACK always plays first**.

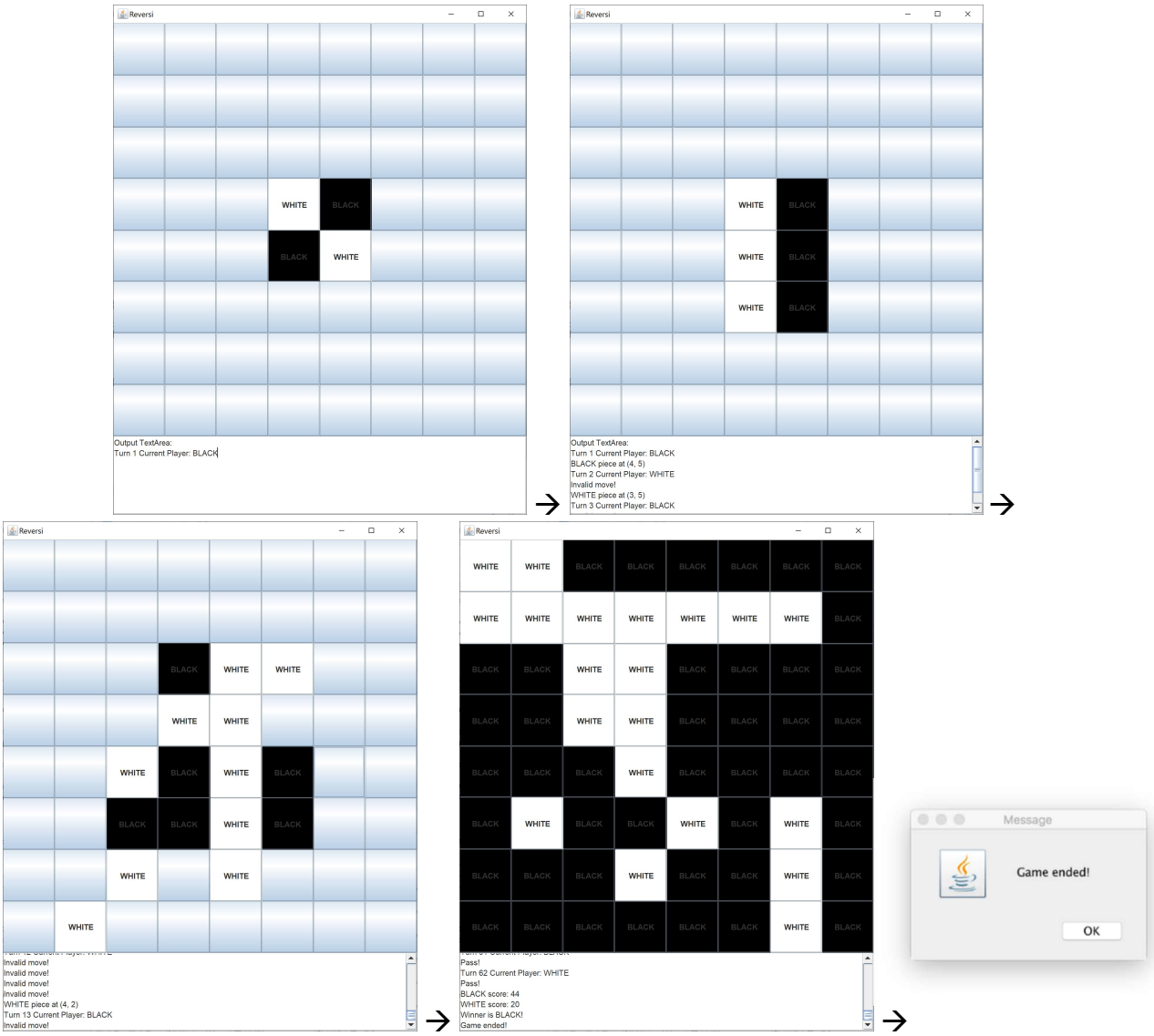
   - Current player must make a **valid move** at a slot such that it is empty AND such move can capture some opponent piece(s).
   - To **capture opponent piece(s)**: in any of the 8 directions, there must be some immediately neighbouring consecutive opponent pieces on a straight line AND terminated with a friend piece at the far end.
   - If the current player cannot make a valid move, the player shall pass.
   - If there are **two consecutive passes**, i.e., double-pass in two consecutive turns, **the game shall end**.
   - To simplify your task, we need not consider *board-full condition*. Such condition will *trigger double-pass* anyway. Thus, *turn number may exceed 60* or even 64 because pass is also counted a turn!
   - When game ends, count and display number of BLACK and WHITE pieces, as well as the winner, whoever got more pieces. If the counts are the same, output "**Draw game!**"
   - After game end, the application shall output "**Game ended!**" After dismissing the game-end dialog, let players review the game until clicking the close button to terminate.

7. Read given source code, pay attention to:

```
/*******************************************************************************
 * Students are expected to inherit and override the following methods in subclasses
 *******************************************************************************/
```

8. Your focus in class **`Reversi`**:
   - Constructor **`Reversi`**
   - Overridable methods: **`initGame()`, `gameAction()`, `checkEndGame()`**
   - Suggested new methods such as: **`isValidMove()`, `mustPass()`, `countPieces()`**
   - You may add other new helper methods.
   - Animation IS NOT REQUIRED! Although the demo may be animated for easy viewing.
   - Hint:
     i. The logic of **`isValidMove()`** is related to **`flipCapturedPieces`**
     ii. You may refer to class **`Connect4Advanced`** source code for idea on **`isValidMove()`**
     iii. You may refer to class **`Connect4`** source code for idea on flipping pieces

Screen shots:



9. **If you have many opened projects, close others or click menu [Run] → [Set Main Project]. You shall set Reversi as the Main class of the project.**

10. Build the project (press the function key [F11] on the keyboard). If there are errors, don't panic. Double-click on the first error message in the Output window. Check the error, correct it and re-compile. Feel tired? Take a rest.

11. You may insert println() statements in your work to inspect variables and intermediate results.

12. Run the application (press the function key [F6] on the keyboard). Enjoy your work.

Your Task:

1. **Locate** your NetBeans project folder, e.g. **`H:\Assignment6\`**.

2. ZIP the project folder **`Assignment6`** and Submit the file **`Assignment6.zip`** via our Online Assignment Collection Box on Blackboard **`https://blackboard.cuhk.edu.hk`**

Marking Scheme and Notes:

1. The submitted program should be free of any typing mistakes, compilation errors and warnings.

2. Comment/remark, indentation, style is under assessment in every programming assignments unless specified otherwise. Variable naming, proper indentation for code blocks and adequate comments are important. ***Include also your personal particulars and your academic honesty declaration statement in a header comment block of each source file.***

3. Remember to do your submission before 6:00 p.m. of the due date. No late submission would be accepted.

4. If you submit multiple times, **ONLY** the content and time-stamp of the **latest** one would be counted. You may delete (i.e. take back) your attached file and re-submit. We ONLY take into account the last submission.

University Guideline for Plagiarism

Attention is drawn to University policy and regulations on honesty in academic work, and to the disciplinary guidelines and procedures applicable to breaches of such policy and regulations. Details may be found at http://www.cuhk.edu.hk/policy/academichonesty/. With each assignment, students are required to submit a statement that they are aware of these policies, regulations, guidelines and procedures.

Faculty of Engineering Guidelines to Academic Honesty

MUST read: https://www.erg.cuhk.edu.hk/erg/AcademicHonesty
(you may need to access via CUHK campus network/ CUHK1x/ CUHK VPN)

Extended Learning Outcomes

Feel free creating more BoardGame subclasses, such as Sudoku, Go, etc.