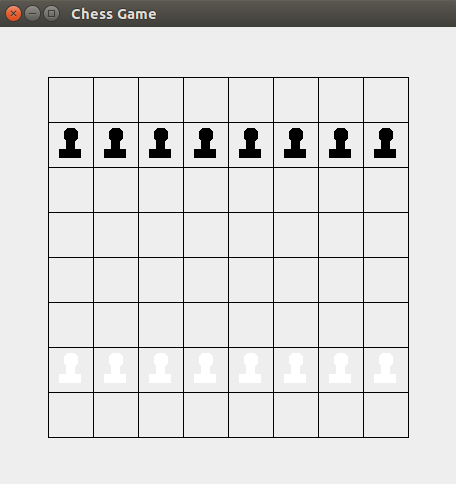
**CMPE 160 Project #1 (Spring 2017)**

**Deadline: April 14, 2017; 23:59**

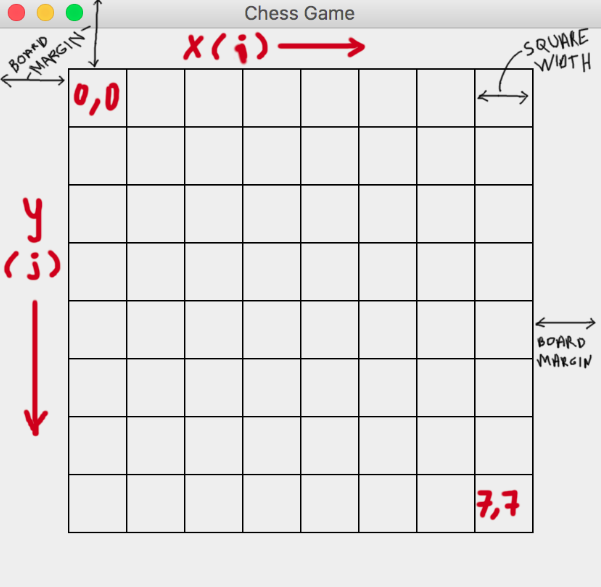
**1. Description**

In this project, you are expected to develop a chess game along with its Graphical User Interface (GUI). For the GUI as well as the game itself, you are required to use the Object Oriented Programming (OOP) principles you have learned so far. The GUI has to be completed using *java.swing* library. You are given an incomplete version of the project. Build and run the code you are given. You should see a chessboard with eight black and eight white pawns. Try to move the pawns by drag & drop. Click on a pawn and drag the mouse to the square in front. In this game the player will be able to move and capture the opponent's pieces using such mouse movements.



**2. Code**

The source code that is provided to you consists of four classes. The **ChessFrame** class is a JFrame subclass and draws the chessboard to the frame using the logic shown in the following image. Examine the **ChessFrame.java** code. It is self-explanatory and has sufficient comments.



**3. Requirements**

You are expected to:

1. Implement the other Pieces:

There is an abstract class, **Piece**, which should contain all the common methods and variables of a chess piece. Its code is commented so that you can understand the purpose and workflow of each method. **Pawn** is a sample subclass of Piece. **ChessFrame** creates 16 instances of **Pawn** and puts them in their corresponding squares on the chessboard using **initializeChessBoard()** method. You should create a subclass of Piece for every other type of chess piece like Knight, Bishop, etc. Then, update the **initializeChessBoard()** method properly to put all the pieces in place.

Note: The shapes of the pieces do not need to be fine arts, but at least they should be recognizable. Take the **drawYourself** code of **Pawn** and make some small changes to it if you have to. Use basic drawings like: triangle, rectangle, oval, etc.

All the pieces should be able to move and capture properly according to the rules of Chess.

1. Implement the Turn-Based Logic:

Currently the player (with the mouse) can move a white piece as much as (s)he wants repeatedly. Besides, a white pawn can capture a white pawn, and a black pawn can capture a black pawn. This is wrong. You should fix this. The player should be able to play one black one white piece in turn, and a piece should only be able to capture the opponent's piece.

Hint: To do this, you might want to add some methods to Piece.java, like getColor, etc. You might also want to add some class variables into the ChessFrame.java to record and update the turns. You must certainly modify the mouseReleased() method. How to achieve the goal is up to you.

1. Fix the Pawn Behavior:

A pawn could move forward by two squares in its first movement. Currently this is not possible. You should fix this.

Hint: You might want to add some class variables and methods to Piece.java to achieve that.

1. Some pieces can not jump over the others:

In fact, only a Knight can jump over other pieces. A rook cannot do that, for instance. You should also take this into account.

1. Give us an Advice:

A chess game could be designed-implemented in many ways/styles. According to your experience, write at least one suggestion/advice/comment to indicate how could we design this game code better (from OOP perspective). Write your suggestion as a comment to the top of MainClass.java code.