Given: Wednesday, March 27, 2013
Electronic Copy Due: Wednesday, April 10, 2013 by 11:59 p.m.

Objectives

- to practice using procedural abstraction when thinking about methods
- to practice incremental program design, development, and testing
- to develop several inherited classes
- to work within a given framework
- to implement abstract methodsto practice using two-dimensional arrays

Introduction

In this assignment, you will be writing a simplified two-player chess program. Both players will be human, so you do not need to write a computer player. When the program begins it will set up a partial chess board with white at the bottom. It will then allow the human users to take turns moving, with white going first until the game ends.

The Game

Chess is a game played on a board with an 8 by 8 grid. On the board are two sets of pieces. The sets are traditionally white and black. There are 16 pieces in each set. Each player is in control of all of the pieces of the same colour. The players take turns as the game progresses, with white always taking the first move. The game is over when one player loses their king. The player who still has their king at the end of the game wins.

The Peices

Here is a summary of the standard chess pieces:

Piece Class	Symbol	Description
`King`	`'K'` or `'k'`	Can move one square in any direction (horizontal, vertical, diagonal; forward or backward).
`Queen`	`'Q'` or `'q'`	Can move multiple squares in any direction.
`Bishop`	`'B'` or `'b'`	Can move multiple squares diagonally.
`Knight`	`'N'` or `'n'`	Can move two squares horizontally and then one vertically, or can move one square horizontally and then two vertically. **Special note**: this is the only piece that can jump over other pieces when moving to a new square.
`Rook`	`'R'` or `'r'`	Can move multiple squares horizontally, or can move multiple squares vertically

Piece Class	Symbol	Description
`Pawn`	`'P'` or `'p'`	Can move one square forward vertically if not capturing, or can move one square forward diagonally if capturing. **Special note**: can move two squares forward on its very first move if not capturing.

The symbol for a white piece will be uppercase. Black pieces will be lower case.

The Board

The game board in chess is an 8 by 8 grid. The standard setup for a chess board is shown below. Each player has 8 pawns, 2 rooks, 2 knights, 2 bishops, a queen and a king.

r	n	b	q	k	b	n	r
p	p	p	p	p	p	p	p
P	P	P	P	P	P	P	P
R	N	B	Q	K	B	N	R

Design

The focus of this assignment is object oriented design using inheritance. To facilitate this, you are required to use an abstract base class representing a chess piece. This class must have all methods needed by a game or board class to facilitate running the game. All of the logic about how the various pieces move and take opponents pieces must be held in the piece classes.

Incremental Construction

Because of the size of this project and the modularity inherent in its design, you may choose to complete only a portion of the assignment with a lower maximum score. A table of required components for each score is given below.

Maximum Score	Required Peices	Required Game Interactions
60	King Simplified Pawn Knight	Initial Board Setup (skipping incomplete pieces) Piece Movement Movement Validation (follow piece rules) Ability to Quit
80	Rook Bishop	Turn Taking (white, black, white, …) Movement Validation (no jumping except knight)

Maximum Score	Required Peices	Required Game Interactions
100	Queen Pawn (complete)	Movement Validation (only take opponents pieces) Automatic Game Ending (<2 kings) Determine Winner

The simplified pawn will always move exactly one space forward, even if taking an opponent's piece.

User Interface

The user interface for your program should look something like the following example. MNote the use of a special coordinate -1, -1 to indicate a choice to quit. In the example, the user's input is in **bold**.

```
********** Welcome to the Super Duper Chess Game **********
*                                                         *
* Upper case pieces are White's and lower case            *
*    pieces are Black's.                                  *
*                                                         *
* To specify a move:                                      *
*    enter row and column for piece to move               *
*    then enter row and column of square to move to       *
*                                                         *
* To quit before game is finished:                        *
*    enter "-1 -1" for piece to move                      *
*                                                         *
***********************************************************

          x
    0 1 2 3 4 5 6 7
  0 r n b q k b n r
  1 p p p p p p p p
  2
  3
y 4
  5
  6 P P P P P P P P
  7 R N B Q K B N R

WHITE's turn to move.
Enter start point x followed by y (e.g. "1 2"):
2 1
Enter end point x followed by y (e.g. "2 7"):
2 2
**** Error: Piece at 2, 1 is not the right colour. ****
```

```
           x
    0 1 2 3 4 5 6 7
  0 r n b q k b n r
  1 p p p p p p p p
  2
  3
y 4
  5
  6 P P P P P P P P
  7 R N B Q K B N R

WHITE's turn to move.
Enter start point x followed by y (e.g. "1 2"):
3 6
Enter end point x followed by y (e.g. "2 7"):
3 4
           x
    0 1 2 3 4 5 6 7
  0 r n b q k b n r
  1 p p p p p p p p
  2
  3
y 4       P
  5
  6 P P P   P P P P
  7 R N B Q K B N R

BLACK's turn to move.
Enter start point x followed by y (e.g. "1 2"):
-1 -1
Enter 'y' to confirm quit: y
Goodbye.
```

Rules to Ignore

In a real game of chess, if a king can be captured by an opposing piece on that player's next turn, the king is said to be in check and players inform one another to "check" their king. A player cannot make a move which places her king in check. When in check, a player must make a move to get out of check. If that is impossible, then s/he has lost. This is called "checkmate".

Your program does not have to handle check detection. In this simplified game, players will not be informed that their king is in check. Furthermore, a player may make any move s/he wishes at his/her turn, even putting the king in check or leaving the king in check.

Your program is not to handle the special chess moves called capturing en passant and castling.

Your program does not have to handle pawn promotion. In chess, that is a special situation that occurs when a pawn reaches the far end of the board. It becomes a queen!

Your program does not have to detect stalemate.

Additional chess information can be found in many books and online resources. Here's the Wikipedia article:

```
http://en.wikipedia.org/wiki/Chess
```

If you have any questions about the rules of chess, please consult with your instructor well before the due date.

Documentation

Each method and class must have a comment describing what it does. In addition, inline documentation should be used as needed. Indentation and the use of constants will also be marked.

Submission

All of your source code should be submitted to the submit drive in a folder with a name in the format *Last_First*_A4 (where *First* and *Last* are replaced with your first and last names). The date and time stamp on your submission will determine if the assignment is on time.

Marking

Your work will be graded heavily on the design of your classes as well as the correctness of your program, documentation and how much of the game components you have completed. All programs submitted will be tested using a set of test data. Programs that do not compile will receive at **most** 40%.