# Pseudocode for Eight Queens (Sean)

//The first class, Queen, is relatively small and will be used in the EightQueens class as an //ArrayList<Queen> in the method addQueens

public class Queen {2

int row, col;

//getters and setters for these fields

}

//The next class, ChessSquarePanel, acts as a single panel and will be used in a 2d array to fill the 8x8 //grid within the EightQueens class

public class ChessSquarePanel extends JPanel {

Color backgroundColor;

Boolean hasQueen;

//default and two-args constructors go here

public void paintComponent(Graphics g){

super.paintComponent(g);

this.setBackground(backroundColor);

if(hasQueen){

//drawString “Q” using specific

//graphics methods that aligns it in the center of the panel

}

}

}

//The next class, EightQueens, is the biggest and acts as both the window which will display the //chessboard and related components as well as contain the method addQueens which carries out

//the algorithm that places the Queens in the right positions

public class EightQueens{

JFrame window;

JPanel header, footer, grid;

ChessSquarePanels[][] squares;

ArrayList<Queen> queens;

public EightQueens(){

createFrame();

createHeader();

createFooter();

createGrid();

window.setVisible(true);

}

//These graphics methods are just stubs because of how finnicky Java graphics are but //essentially each will instantiate their respective JFrame or JPanel, give it appropriate size,

//and set it to visible if necessary

public void createFrame();

public void createHeader();

public void createFooter();

public void createGrid(){

//this method is a little longer than the others because it also instantiates the 2d array //of ChessSquarePanels

}

//This method simply displays the example solution demonstrated in the writeup

public void exampleSolution(){

//without going into the actual code, this method sets the ChessSquarePanel at each

//corresponding position as shown in the write up to display a queen. There is little //actual coding going on as this method does not actually solve the problem but just //displays one possible solution

}

//This method is perhaps the most important

}