Name:_____

Senior Design and Development Seminar
Take Home Midterm

Due by midnight Friday March 6th

Question 1_____(20)

Question 2_____(30)

Question 3_____(30)

Question 4_____(20)

Question 5_____(20)

Graduate student must do both 4 &5. Undergraduates pick one of 4 or 5

Undergrad Total_____(100)

Grad Total_____(120)

1. We had two presentations about version control systems in class. We also had some discussions about the prehistory before version control systems.
   (a) Why do we use version control? What does it buy us over other ways of tracking changes?

   (b) Compare and contrast git vs TFVC (Team Foundation Version Control)

(c) Discuss the advantages of using a debugger included with the ide such as the pycharm one that Tom demoed to us over other debugging techniques.

2. Below (and on the following pages) is some code I got off of badprogramming.com. It is bad.
   (a) (30pts) point out through color coded highlighting (when needed) and your own written observations, what things are wrong with this code. Consider in particular the suggestions from the clean code book.

```
1:  //EXTENDED VERSION. CHRISTMAS EDITION. ALSO A SWEDISH VERSION AVAILBLE.
2:
3:  package packageTicTacToe;
4:
5:  import java.awt.BorderLayout;
6:  import java.awt.Color;
7:  import java.awt.FlowLayout;
8:  import java.awt.GridLayout;
9:  import java.awt.event.ActionEvent;
10: import java.awt.event.ActionListener;
11: import java.util.Random;
12:
13: import javax.swing.JButton;
14: import javax.swing.JFrame;
15: import javax.swing.JLabel;
16: //import javax.swing.JMenu;
17: import javax.swing.JMenuBar;
18: import javax.swing.JMenuItem;
19: import javax.swing.JOptionPane;
20: import javax.swing.JPanel;
21: import javax.swing.JTextArea;
22:
23: public class TicTacToeEng implements ActionListener {
24:       final String VERSION = "1.7 Christmas Edition!";
25:
26:       JFrame window = new JFrame("Tic Tac Toe "+ VERSION);
27:       JFrame windowL = new JFrame("Tic Tac Toe Language");
28:
29:       JMenuBar mnuMain = new JMenuBar();
30:       JMenuBar mnuL = new JMenuBar();
31:       JMenuItem mnuNewGame = new JMenuItem("New Game,");
32:       JMenuItem mnuInstruction = new JMenuItem("Instructions,");
33:       JMenuItem mnuExit = new JMenuItem("Exit");
34:       JMenuItem mnuAbout = new JMenuItem("About,");
35:       JMenuItem mnuNews = new JMenuItem("News,");
36:       JMenuItem mnuRestart = new JMenuItem("Restart,");
37:       JMenuItem mnuLanguage = new JMenuItem("Language,");
38:       JMenuItem mnuLanguageS = new JMenuItem("Svenska");
39:       JMenuItem mnuLanguageE = new JMenuItem("English");
40:
41:       JButton btn1v1 = new JButton("Player Versus Player");
42:       JButton btn1vCPU = new JButton("Player Versus CPU");
43:       JButton btnBack = new JButton("Go Back");
44:       JButton btnLanguageE = new JButton("English");
45:       JButton btnLanguageS = new JButton("Svenska");
46:       JButton[] btnEmpty = new JButton[10];
47:
48:       JPanel pnlNewGame = new JPanel();
49:       JPanel pnlNorth = new JPanel();
50:       JPanel pnlSouth = new JPanel();
51:       JPanel pnlTop = new JPanel();
52:       JPanel pnlBottom = new JPanel();
53:       JPanel pnlBottomBottom = new JPanel();
54:       JPanel pnlPlayingField = new JPanel();
```

```
55:         JLabel lblTitle = new JLabel("Tic Tac Toe " + VERSION);
56:         JTextArea txtMessage = new JTextArea();
57:
58:         final int[][] winCombo = { { 1, 2, 3 }, { 1, 4, 7 }, { 1, 5, 9 },
59:                         { 4, 5, 6 }, { 2, 5, 8 }, { 3, 5, 7 }, { 7, 8, 9 }, { 3, 6, 9 }};
60:
61:         final int X = 412;
62:         final int Y = 268;
63:         final int color = 0;
64:         boolean inGame = false;
65:         boolean win = false;
66:         boolean btnEmptyClicked = false;
67:         boolean isPlayingWithCPU = false;
68:         boolean didAnyoneWin = false;
69:
70:         String message;
71:         int turn = 1;
72:         int wonNumber1 = 1;
73:         int wonNumber2 = 1;
74:         int wonNumber3 = 1;
75:
76:         public TicTacToeEng() {
77:                         window.setSize(1000, 550);
78:                         window.setLocation(200, 100);
79:                         window.setResizable(true);
80:                         window.setLayout(new BorderLayout());
81:                         window.setDefaultCloseOperation(3);
82:                         windowL.setSize(800, 325);
83:                         windowL.setLocation(200, 100);
84:                         windowL.setResizable(true);
85:                         windowL.setLayout(new BorderLayout());
86:                         windowL.setDefaultCloseOperation(3);
87:                         windowL.setDefaultCloseOperation(2);
88:                 pnlNewGame.setLayout(new GridLayout(2, 1, 2, 10));
89:                 pnlNorth.setLayout(new FlowLayout(1));
90:                 pnlSouth.setLayout(new FlowLayout(1));
91:
92:                 pnlNorth.setBackground(new Color(220, 20, 30));
93:                 pnlSouth.setBackground(new Color(220, 100, 90));
94:
95:                 pnlTop.setBackground(new Color(220, 100, 90));
96:                 pnlBottom.setBackground(new Color(220, 100, 90));
97:                 pnlBottomBottom.setBackground(new Color(220, 100, 90));
98:
99:                 pnlTop.setLayout(new FlowLayout(1));
100:                pnlBottom.setLayout(new FlowLayout(1));
101:                pnlNewGame.setBackground(Color.pink);
102:
103:                mnuMain.add(mnuNewGame);
104:                mnuMain.add(mnuInstruction);
105:                mnuMain.add(mnuNews);
106:                mnuMain.add(mnuAbout);
107:                mnuMain.add(mnuLanguage);
108:                mnuMain.add(mnuRestart);
109:                mnuMain.add(mnuExit);
110:                mnuL.add(mnuExit);
111:
112:                pnlNewGame.add(btn1v1);
113:                pnlNewGame.add(btn1vCPU);
```

```
114:                //pnlNewGame.add(btnLanguageE);
115:                //pnlNewGame.add(btnLanguageS);
116:
117:                mnuNewGame.addActionListener(this);
118:                mnuExit.addActionListener(this);
119:                mnuNews.addActionListener(this);
120:                mnuInstruction.addActionListener(this);
121:                mnuAbout.addActionListener(this);
122:                mnuRestart.addActionListener(this);
123:                mnuLanguage.addActionListener(this);
124:                mnuLanguageS.addActionListener(this);
125:                mnuLanguageE.addActionListener(this);
126:                btn1v1.addActionListener(this);
127:                btn1vCPU.addActionListener(this);
128:                btnBack.addActionListener(this);
129:                mnuMain.add(mnuExit);
130:                //btnLanguageS.addActionListener(this);
131:                //btnLanguageE.addActionListener(this);
132:
133:
134:                pnlPlayingField.setLayout(new GridLayout(3, 3, 2, 2));
135:                pnlPlayingField.setBackground(Color.pink);
136:                for (int i = 1; i <= 9; ++i) {
137:                        btnEmpty[i] = new JButton();
138:                        btnEmpty[i].setBackground(new Color(220, 100, 90));
139:                        btnEmpty[i].addActionListener(this);
140:                        pnlPlayingField.add(btnEmpty[i]);
141:                }
142:
143:                pnlNorth.add(mnuMain);
144:                pnlSouth.add(lblTitle);
145:
146:                window.add(pnlNorth, "North");
147:                window.add(pnlSouth, "Center");
148:                window.setVisible(true);
149:        }
150:
151:        public void actionPerformed(ActionEvent click) {
152:                Object source = click.getSource();
153:                for (int i = 1; i <= 9; ++i) {
154:                        if ((source == btnEmpty[i]) && (turn < 10)){
155:                                commonClickActions(btnEmpty[i]);
156:
157:                                if(isPlayingWithCPU){
158:                                        btnEmpty[i].setText("X");
159:                                        turn++;
160:                                        checkWin();
161:                                        if(didAnyoneWin)
162:                                                return;
163:
164:                                        int number = 0;
165:                                        while(btnEmpty[number] == null ||
166:                                                        (btnEmpty != null &&
btnEmpty[number].getText() != "") || number < 1){
167:                                                if(turn >= 9){
168:                                                        return;
169:                                                }
170:                                                number = (new Random()).nextInt(10);
171:                                        }
172:
```

```
173:
174:                                         if(btnEmpty[number].getText() == ""){
175:                                                 btnEmpty[number].setText("O");
176:                                                 commonClickActions(btnEmpty[number]);
177:                                                 turn++;
178:                                                 checkWin();
179:                                         }
180:
181:                                 }
182:                                 else{
183:                                         if (turn % 2 != 0)
184:                                                 btnEmpty[i].setText("X");
185:                                         else
186:                                                 btnEmpty[i].setText("O");
187:                                         turn++;
188:                                 }
189:                         }
190:                 }
191:                 if (btnEmptyClicked && !isPlayingWithCPU) {
192:                         checkWin();
193:                         btnEmptyClicked = false;
194:                 }
195:                 if (source == mnuNewGame) {
196:                         clearPanelSouth();
197:                         pnlSouth.setLayout(new GridLayout(2, 1, 2, 5));
198:                         pnlTop.add(pnlNewGame);
199:                         pnlBottom.add(btnBack);
200:                         //pnlBottomBottom.add(btnLanguageS);
201:                         //pnlBottomBottom.add(btnLanguageE);
202:                         pnlSouth.add(pnlTop);
203:                         pnlSouth.add(pnlBottom);
204:                 } else if (source == btn1v1) {
205:                         if (inGame) {
206:                                 int option = JOptionPane
207:                                                 .showConfirmDialog(
208:                                                                 null,
209:                                                                 "If you start a new game, your
current game will be lost.\nAre you sure you want to continue?",
210:                                                                 "New Game?", 0);
211:                                 if (option == 0) {
212:                                         inGame = false;
213:                                 }
214:                         }
215:                         if (!(inGame)) {
216:                                 isPlayingWithCPU = false;
217:                                 clearGame();
218:                                 showGame();
219:                         }
220:
221:                 } else if (source == btn1vCPU) {
222:                         //JOptionPane.showMessageDialog(null, "Coming Soon! Remember that this is a
Beta for Developers!");
223:                         if(inGame)
224:                                 if(JOptionPane.showConfirmDialog(null, "If you start a new game,
your current game will be lost.\nAre you sure you want to continue?", "New Game", 0) == 0)
225:                                         inGame = false;
226:                         if(!inGame){
227:                                 isPlayingWithCPU = true;
228:                                 clearGame();
229:                                 showGame();
```

```
230:                         }
231:                 } else if (source == mnuLanguageS) {
232:                         window.setVisible(false);
233:                         new TicTacToe();
234:                 } else if (source == mnuLanguageE) {
235:                         window.setVisible(false);
236:                         new TicTacToeEng();
237:                 } else if (source == mnuExit) {
238:                         int option = JOptionPane.showConfirmDialog(null,
239:                                         "Do you want to Exit?", "Exit?", 0);
240:                         if (option == 0)
241:                                 System.exit(0);
242:         } else if (source == mnuRestart) {
243:                                 int option = JOptionPane.showConfirmDialog(null,
244:                                                 "Do you really want to restart and reload? Your
current game will be lost.\nAre you sure you want to continue?", "Restart", 0);
245:                                 if (option == 0);
246:                                 new TicTacToe();
247:                                 window.setVisible(false);
248:         } else {
249:                 } if ((source == mnuInstruction) || (source == mnuAbout)) {
250:                         clearPanelSouth();
251:                         String message1 = "";
252:                         txtMessage.setBackground(new Color(220, 100, 90));
253:                         if (source == mnuInstruction) {
254:                                 message1 = "Instructions:\n\nYour goal is to be the first player to
get three X's ot three O's in a row.\n"
255:                                                 + "You can win horizontally, diagonally, or
vertically.";
256:                         } else {
257:                                 message1 = "Version: " + VERSION + "\n\n\n\nSpecial thanks
to:\nOracle, for making Java and JavaScript (We made this game in Java.),\nThe Eclipse team, Making
Eclipse\nAnd\nTo you, for Playing the game, rating and Downloading!\nThank You!";
258:                         }
259:                         txtMessage.setEditable(false);
260:                         txtMessage.setText(message1);
261:                         pnlSouth.setLayout(new GridLayout(2, 1, 2, 5));
262:                         pnlTop.add(txtMessage);
263:                         pnlBottom.add(btnBack);
264:                         pnlSouth.add(pnlTop);
265:                         pnlSouth.add(pnlBottom);
266:                 } else if (source == mnuLanguage) {
267:                         clearPanelSouth();
268:                         //String message2 = "";
269:                         txtMessage.setBackground(new Color(220, 100, 90));
270:                         //message2 = "Choose a Language:\nVälj ett Språk:";
271:                         //txtMessage.setEditable(false);
272:                         //txtMessage.setText(message2);
273:                         //txtMessage.setBackground(new Color(220, 100, 90));
274:                         pnlSouth.setLayout(new GridLayout(2, 1, 2, 5));
275:                         //windowL.setVisible(true);
276:                         pnlTop.add(txtMessage);
277:                         pnlBottom.remove(btnBack);
278:                         //pnlBottomBottom.add(btnLanguageS);
279:                         pnlSouth.remove(pnlTop);
280:                         pnlSouth.remove(pnlBottom);
281:                         pnlSouth.remove(pnlBottomBottom);
282:                         mnuMain.remove(mnuNewGame);
283:                         mnuMain.remove(mnuInstruction);
284:                         mnuMain.remove(mnuNews);
```

```
285:                        mnuMain.remove(mnuAbout);
286:                        mnuMain.remove(mnuLanguage);
287:                        mnuMain.remove(mnuRestart);
288:                        mnuL.add(mnuExit);
289:                        mnuMain.add(mnuLanguageE);
290:                        mnuMain.add(mnuLanguageS);
291:                } else if (source == mnuNews) {
292:                        clearPanelSouth();
293:                        String message = "";
294:                        message = "NEWS:\nGreenPortalGames Has a new member! LEGOlord208, Our Head-
System-Moderator-Developer!\nVERSIONS:\nPLANNED:\n1.8.5.2- Release at Googleplay\n1.8.4- Release at
AppStore\n1.7-\nRELEASED:\n1.6-Added restart button! \n1.5- Added News!\n1.4- Added VS CPU! :D\n1.0 ->
Version 1.3.62_Dev-24- fixed bugs, edited CPU for smoother performance (Biggest Update, version, 1.3,
But all the features came in here)\n";
295:
296:
297:                        txtMessage.setEditable(false);
298:                        txtMessage.setText(message);
299:                        pnlSouth.setLayout(new GridLayout(2, 1, 2, 5));
300:                        pnlTop.add(txtMessage);
301:                        pnlBottom.add(btnBack);
302:                        pnlSouth.add(pnlTop);
303:                        pnlSouth.add(pnlBottom);
304:                //} else if (source == btnLanguageS) {
305:                        //window.setVisible(false);
306:                        //new TicTacToe();
307:
308:                //} else if (source == btnLanguageE) {
309:                        //window.setVisible(false);
310:                        //new TicTacToeEng();
311:
312:                } else if (source == btnBack) {
313:                        if (inGame) {
314:                                showGame();
315:                        } else {
316:                                clearPanelSouth();
317:                                pnlSouth.setLayout(new FlowLayout(1));
318:                                pnlNorth.setVisible(true);
319:                                pnlSouth.add(lblTitle);
320:                        }
321:                }
322:                                pnlSouth.setVisible(false);
323:                                pnlSouth.setVisible(true);
324:          }
325:
326:
327:        public void showGame() {
328:                clearPanelSouth();
329:                inGame = true;
330:                pnlSouth.setLayout(new BorderLayout());
331:                pnlSouth.add(pnlPlayingField, "Center");
332:                pnlPlayingField.requestFocus();
333:        }
334:
335:        public void checkWin() {
336:                for (int i = 0; i <= 7; ++i) {
337:                        if ((btnEmpty[winCombo[i][0]].getText().equals(""))
338:                                        || (!(btnEmpty[winCombo[i][0]].getText()
339:                                                        .equals(btnEmpty[winCombo[i][1]]
340:                                                                        .getText())))
```

```
341:                                        || (!(btnEmpty[winCombo[i][1]].getText()
342:                                                        .equals(btnEmpty[winCombo[i][2]]
343:                                                                        .getText())))) {
344:                                continue;
345:                        }
346:
347:                        win = true;
348:                        wonNumber1 = winCombo[i][0];
349:                        wonNumber2 = winCombo[i][1];
350:                        wonNumber3 = winCombo[i][2];
351:                        btnEmpty[wonNumber1].setBackground(Color.pink);
352:                        btnEmpty[wonNumber2].setBackground(Color.pink);
353:                        btnEmpty[wonNumber3].setBackground(Color.pink);
354:                        break;
355:                }
356:
357:                if ((win) || ((!(win)) && (turn > 9))) {
358:                        if (win) {
359:                                if(isPlayingWithCPU){
360:                                        if (turn % 2 == 0)
361:                                                message = "You won! Congratulations!";
362:
363:                                        else
364:                                                message = "The CPU won! Better luck next time!";
365:                                }
366:                                else{
367:                                        if (turn % 2 == 0)
368:                                                message = "X won! Congratulations!";
369:                                        else
370:                                                message = "O won! Congratulations!";
371:                                }
372:                                didAnyoneWin = true;
373:                                win = false;
374:                        } else if ((!(win)) && (turn > 9)) {
375:                                message = "Both players have tied.\nBetter luck next time!";
376:                        }
377:                        JOptionPane.showMessageDialog(null, message);
378:                        for (int i = 1; i <= 9; ++i)
379:                                btnEmpty[i].setEnabled(false);
380:                }
381:        }
382:
383:        public void clearPanelSouth() {
384:                pnlSouth.remove(lblTitle);
385:                pnlSouth.remove(pnlTop);
386:                pnlSouth.remove(pnlBottom);
387:                pnlSouth.remove(pnlPlayingField);
388:                pnlTop.remove(pnlNewGame);
389:                pnlTop.remove(txtMessage);
390:                pnlBottom.remove(btnBack);
391:        }
392:
393:        public void clearGame(){
394:                btnEmpty[wonNumber1].setBackground(new Color(220,
395:                                100, 90));
396:                btnEmpty[wonNumber2].setBackground(new Color(220,
397:                                100, 90));
398:                btnEmpty[wonNumber3].setBackground(new Color(220,
399:                                100, 90));
400:                turn = 1;
```

```
401:                for (int i = 1; i < 10; ++i) {
402:                        btnEmpty[i].setText("");
403:                        btnEmpty[i].setEnabled(true);
404:                }
405:                win = false;
406:                didAnyoneWin = false;
407:        }
408:
409:        public void commonClickActions(JButton button){
410:                btnEmptyClicked = true;
411:                button.setEnabled(false);
412:                pnlPlayingField.requestFocus();
413:        }
414:
415:        public static void main(String[] args) {
416:                new TicTacToeEng();
417:        }
418:  }
```

3. For the code from question 2 above, refactor it to fix what is wrong with it. (at least from a clean code standpoint). If turning this in electronically, include your code as an attachment. If turning it in in hard copy, staple you code to the back. The space below left for my grading comments.

4. In class we discussed team building for software projects. Matt also introduced us to several collaboration tools. In the space below, discuss the following two things:
   (a) which of these tools (if any) could help in building a team and why?
   (b) If a team was already formed/gelled and then some members had to relocate to a remote site for a few months, which of these tools would help the team stay coherent and why?

5. Sean gave a presentation on disruptive innovation in technology. Evaluate the following technologies and argue if they are incremental or disruptive technological innovations. If they are disruptive, be sure to mention which industry they eliminated (or are eliminating).
   (a) CDs

   (b) Tivo

   (c) World wide web

   (d) hybrid cars like the Toyota Prius

   (e) Driverless cars like the ones google is producing