Name: Keith Pham

SID: 32507133

# Homework 2

## Best Practice

1. Modularity (Remove DRY code, replaced with OpenClose methods)
   - Replace hardcoded win conditions with Computing Algorithm
   - A function' lengths are shortened to at most one screen
2. Open Close Principle (Game can be expanded to nxn size)
   - Game can be expanded to NxN size upon initialization without changing any internal variable
   - Minimize the need to modify old functions when the app is developed further
3. Magic strings/numbers are replaced with Static String
4. Encapsulation: All fields are set to Private; fields now can be accessed via Setter and Getter

## Composite Design Pattern

First we need to split RowGameUI constructor into smaller functions:

```
public void initCompA(RowGameController controller) {
    JPanel gamePanel = new JPanel(new FlowLayout());
    JPanel game = new JPanel(new GridLayout(this.width, this.width));
    gamePanel.add(game, BorderLayout.CENTER);

    gui.add(gamePanel, BorderLayout.NORTH);

    // Initialize a JButton for each cell of the NxN game board.
    for(int row = 0; row < this.width; row++) {
        for(int column = 0; column < this.width; column++) {
            blocks[row][column] = new JButton();
            blocks[row][column].setPreferredSize(new Dimension(75,75));
            game.add(blocks[row][column]);
            blocks[row][column].addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    controller.move((JButton)e.getSource());
                }
            });
        }
```

```
public void initCompC(RowGameController controller) {
    JPanel messages = new JPanel(new FlowLayout());
    messages.setBackground(Color.white);

    gui.add(messages, BorderLayout.SOUTH);

    messages.add(playerturn);
    playerturn.setText(RowGameModel.START_TURN);
}
```

These two new functions will be constructor of two new classes RowGameBoardView and RowGameStatusView. Respectively, the fields for RowGameBoardView are

```
private RowGameModel gameModel = new RowGameModel();
private JButton[][] blocks;
```

update method will update the blocks from gameModel and update blocks from View

and RowGameStatusView

```
private RowGameModel gameModel = new RowGameModel();
private JTextArea playerturn = new JTextArea();
```

update method will update the Component C text area from gameModel and text area from View

## Observer Design Pattern

Field that corresponds to an Observable: blocksData

Java Swing that corresponds to its Observer: JButton blocks

Implementation of update method: addActionListener