Concordia University
Engineering and Computer Science

SOEN 6441: Advanced Programming Practices

Winter 2019

Project – Risk Game

(Build 3)

Refactoring Document

Submitted By:
Team 30

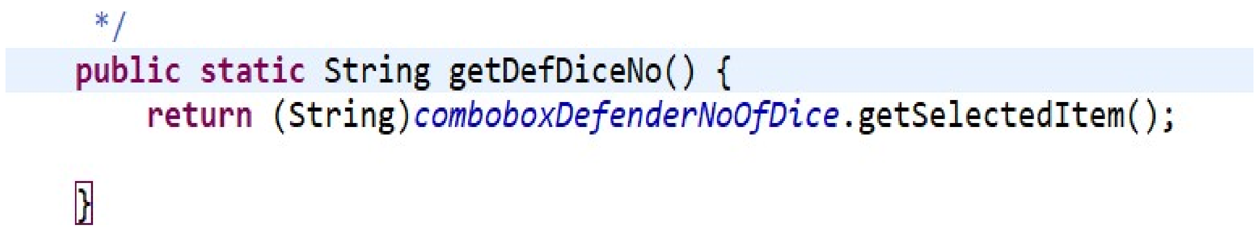
Submitted To:
Amin Ranj Bar

NAME	ID
1. Gargi Sharma	40042837
2. Jaiganesh Varadharaju	40081862
3. Md Hasibul Huq	40087646
4.Narendran Krishnakumar	40089619
5. Zakiya Jafrin	40021416

Refactoring

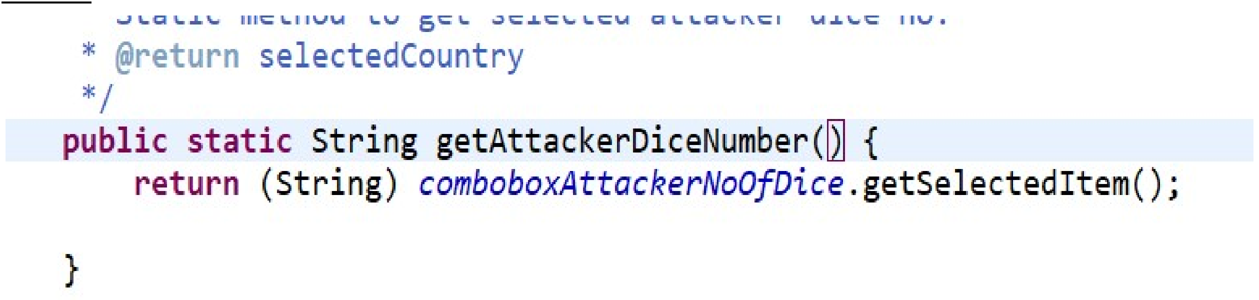
Refactoring is the practice for restructuring the existing code or altering the internal structure without changing the external behaviour which improves the code quality.

1. **Refactoring Type**: **Rename method Name**
   Refactoring Target: Understandability
   Refactoring Class: BoardView.java
   Description: Change the function name from getDefDiceno() to getDefDicenumber() to get the dice numbers of defender country.
   <u>**Build2**</u>



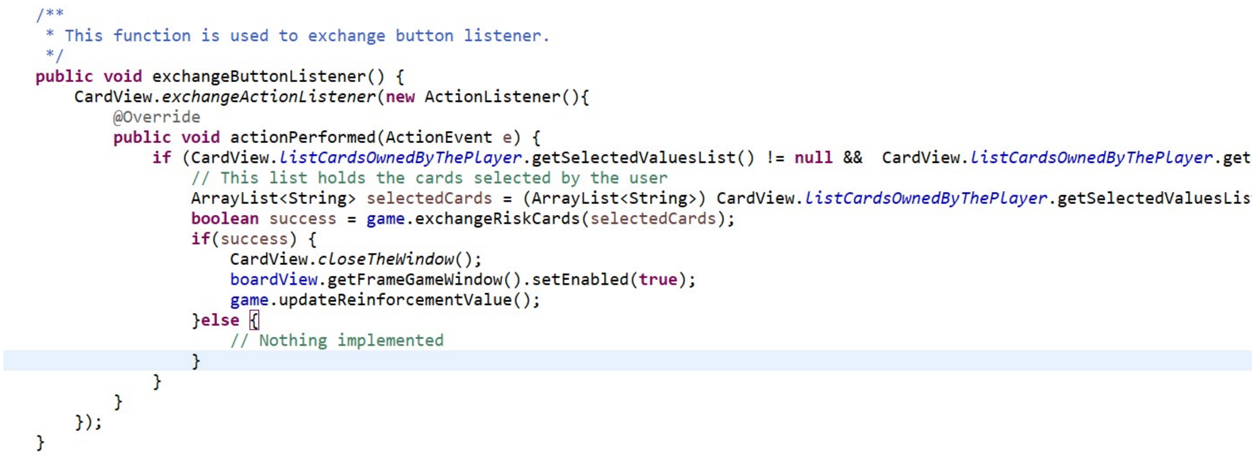
```
  */
 public static String getDefDiceNo() {
     return (String)comboboxDefenderNoOfDice.getSelectedItem();

 }
```

   <u>**Build3**</u>



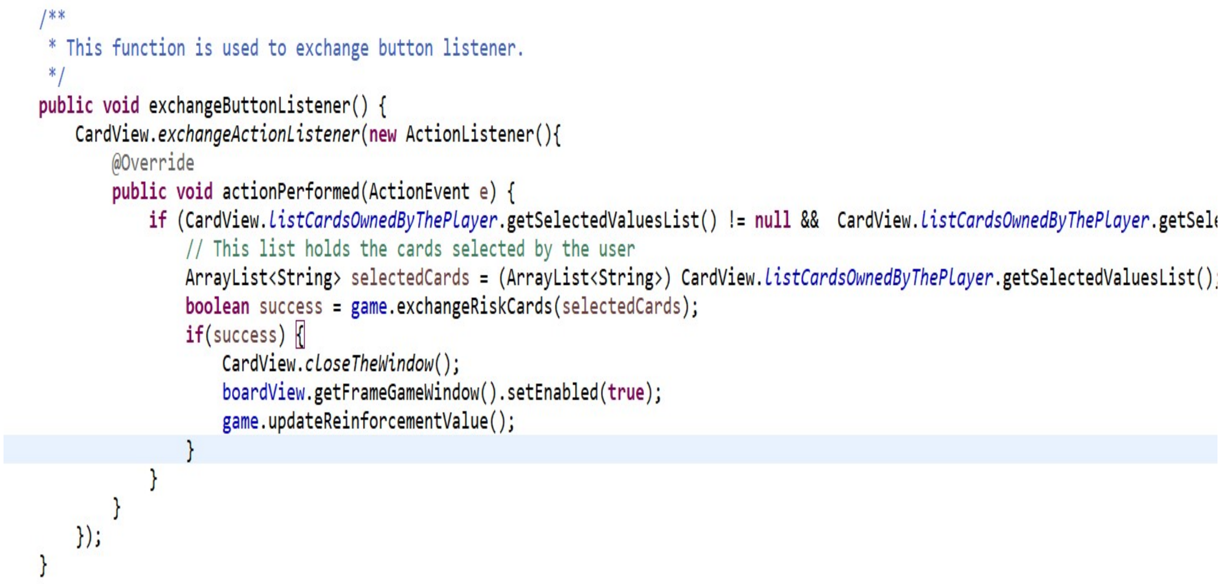
```
   Static method to get selected attacker dice no.
 * @return selectedCountry
 */
public static String getAttackerDiceNumber() {
    return (String) comboboxAttackerNoOfDice.getSelectedItem();

}
```

2. **Refactoring Type**: **Restructure Conditional Statement**
Refactoring Target: Readability
Refactoring Class: GameController..java
Description: Change the Conditional statement and remove if condition.
<u>Build2</u>



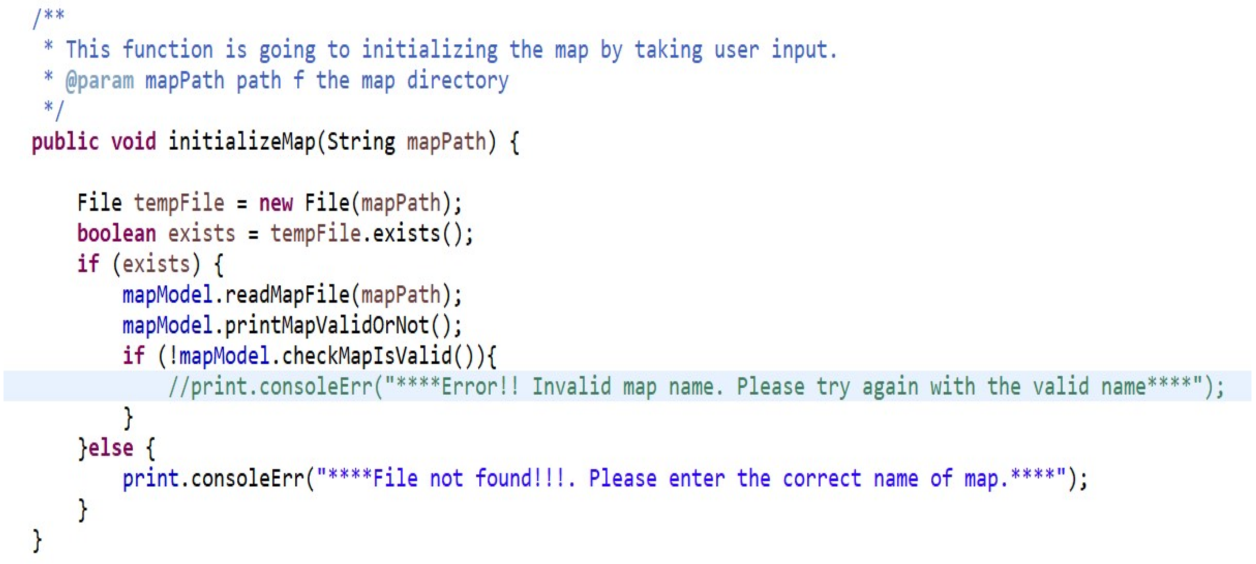
```
/**
 * This function is used to exchange button listener.
 */
public void exchangeButtonListener() {
    CardView.exchangeActionListener(new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            if (CardView.listCardsOwnedByThePlayer.getSelectedValuesList() != null &&  CardView.listCardsOwnedByThePlayer.get
                // This list holds the cards selected by the user
                ArrayList<String> selectedCards = (ArrayList<String>) CardView.listCardsOwnedByThePlayer.getSelectedValuesLis
                boolean success = game.exchangeRiskCards(selectedCards);
                if(success) {
                    CardView.closeTheWindow();
                    boardView.getFrameGameWindow().setEnabled(true);
                    game.updateReinforcementValue();
                }else {
                    // Nothing implemented
                }
            }
        }
    });
}
```

<u>Build3</u>



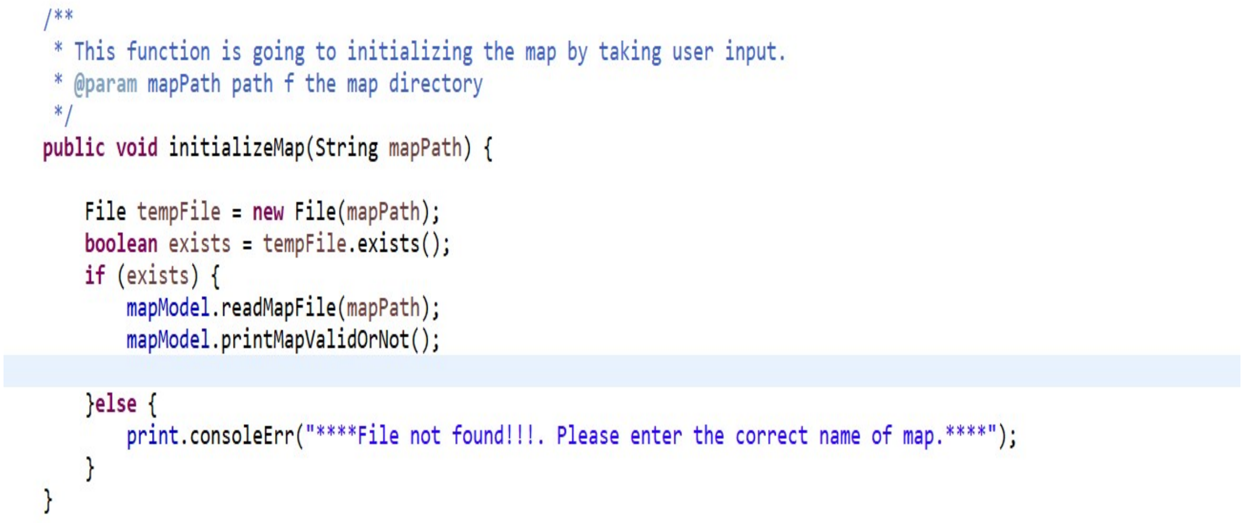
```
/**
 * This function is used to exchange button listener.
 */
public void exchangeButtonListener() {
    CardView.exchangeActionListener(new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            if (CardView.listCardsOwnedByThePlayer.getSelectedValuesList() != null &&  CardView.listCardsOwnedByThePlayer.getSel
                // This list holds the cards selected by the user
                ArrayList<String> selectedCards = (ArrayList<String>) CardView.listCardsOwnedByThePlayer.getSelectedValuesList();
                boolean success = game.exchangeRiskCards(selectedCards);
                if(success) {
                    CardView.closeTheWindow();
                    boardView.getFrameGameWindow().setEnabled(true);
                    game.updateReinforcementValue();
                }
            }
        }
    });
}
```

3. **Refactoring Type**: **Restructure Conditional Statement**
Refactoring Target: Readability
Refactoring Class: GameController..java
Description: Remove unimplemented if condition to improve readibility..
Build2



```
/**
 * This function is going to initializing the map by taking user input.
 * @param mapPath path f the map directory
 */
public void initializeMap(String mapPath) {

    File tempFile = new File(mapPath);
    boolean exists = tempFile.exists();
    if (exists) {
        mapModel.readMapFile(mapPath);
        mapModel.printMapValidOrNot();
        if (!mapModel.checkMapIsValid()){
            //print.consoleErr("****Error!! Invalid map name. Please try again with the valid name****");
        }
    }else {
        print.consoleErr("****File not found!!!. Please enter the correct name of map.****");
    }
}
```

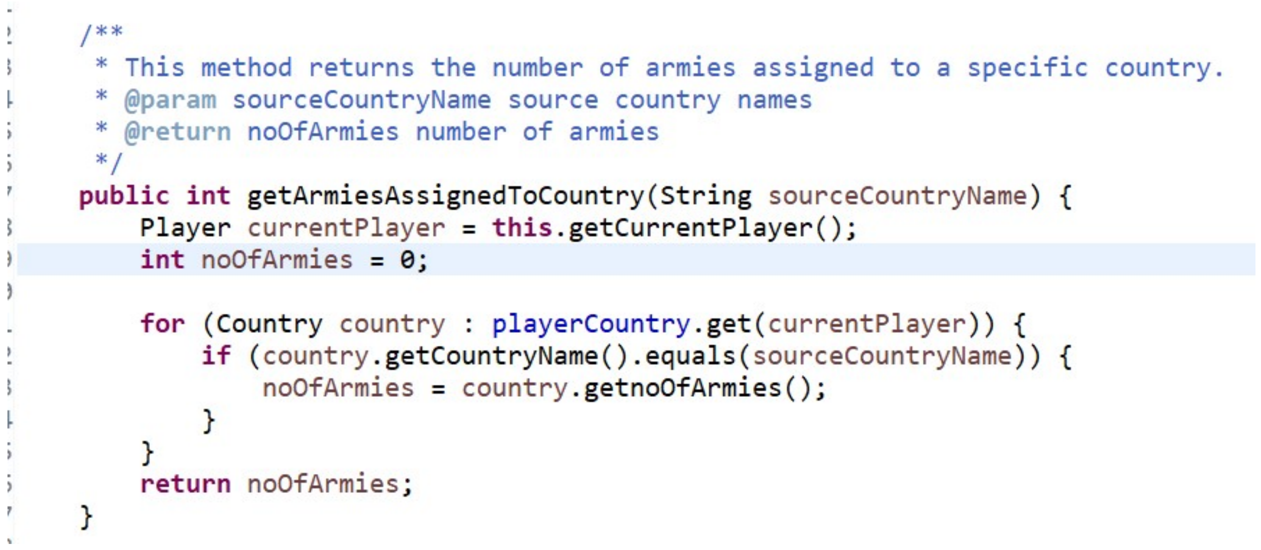
Build3



```
/**
 * This function is going to initializing the map by taking user input.
 * @param mapPath path f the map directory
 */
public void initializeMap(String mapPath) {

    File tempFile = new File(mapPath);
    boolean exists = tempFile.exists();
    if (exists) {
        mapModel.readMapFile(mapPath);
        mapModel.printMapValidOrNot();

    }else {
        print.consoleErr("****File not found!!!. Please enter the correct name of map.****");
    }
}
```

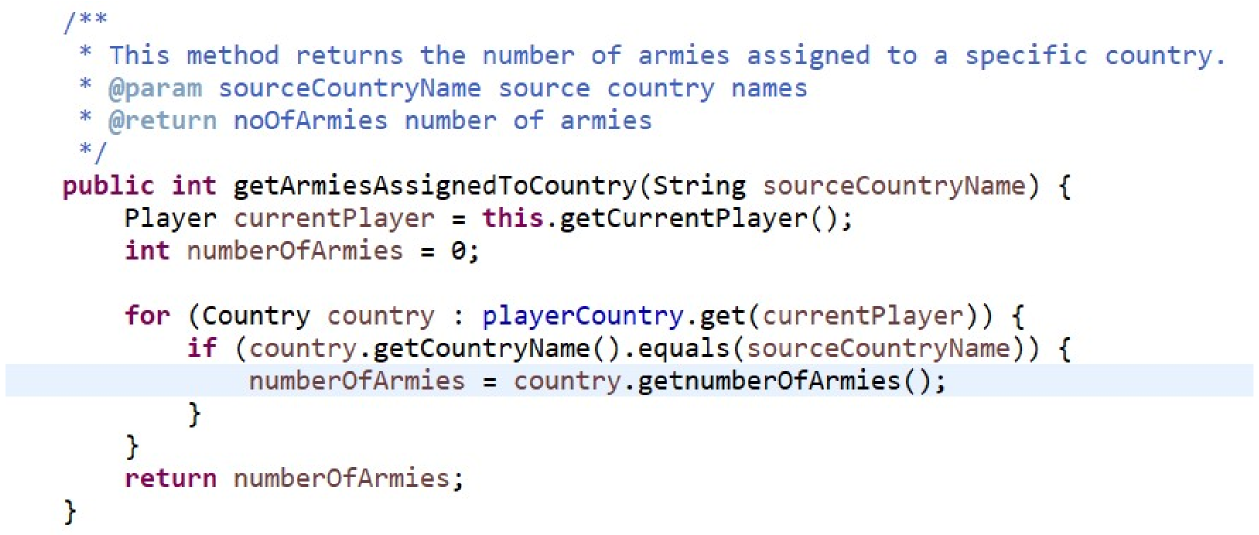
4. **Refactoring Type**: **Rename variable name**
Refactoring Target: Understandability
Refactoring Class: Game.java
Description: Rename the variable name from noOfArmies() to numberfArmies() to count the number for armies needed in the Build3 requirements.

Build2



```
    /**
     * This method returns the number of armies assigned to a specific country.
     * @param sourceCountryName source country names
     * @return noOfArmies number of armies
     */
    public int getArmiesAssignedToCountry(String sourceCountryName) {
        Player currentPlayer = this.getCurrentPlayer();
        int noOfArmies = 0;

        for (Country country : playerCountry.get(currentPlayer)) {
            if (country.getCountryName().equals(sourceCountryName)) {
                noOfArmies = country.getnoOfArmies();
            }
        }
        return noOfArmies;
    }
```

Build3



```
    /**
     * This method returns the number of armies assigned to a specific country.
     * @param sourceCountryName source country names
     * @return noOfArmies number of armies
     */
    public int getArmiesAssignedToCountry(String sourceCountryName) {
        Player currentPlayer = this.getCurrentPlayer();
        int numberOfArmies = 0;

        for (Country country : playerCountry.get(currentPlayer)) {
            if (country.getCountryName().equals(sourceCountryName)) {
                numberOfArmies = country.getnumberOfArmies();
            }
        }
        return numberOfArmies;
    }
```