```
1 import java.util.Scanner;
3 /**
  * This class holds the user interaction functionality of the card game and
   * contains the main method to start the game. It extends Board class
5
   * in order to access BOARD listArray and protected methods.
8
 9 public class ElevensGame extends Board
10 {
11
       private int selectCard() // Allows user to select card slot with appropriate validation in place
12
13
           Scanner scan = new Scanner(System.in);
14
           AListArray<Integer> SELECTION = checkPossibleMoves(BOARD);
15
           int selection = 0;
16
17
           System.out.print("Please Enter a Slot Number for any Card (or any letter key for a hint): ");
18
           boolean isNum=false;
19
           while(!isNum) { // while loop validation only accepting when isNum is set to true
20
               if(scan.hasNextInt()) {
21
                       selection = scan.nextInt();
22
                   if(selection >= 1 && selection <= getBoardLength())</pre>
23
                       isNum=true:
24
                   else {
25
                       System.out.println("Invalid Entry. Not within range of slots in Board.");
26
                       System.out.print("\nPlease Enter a slot number for any card (or any letter key for
    a hint): ");
27
                   }
28
               }
29
               else {
                   if(SELECTION.getLength() < 3) {</pre>
30
31
                       System.out.println("Hint: " + getBoardEntry(SELECTION.getEntry(1)) + " & " +
32
                               getBoardEntry(SELECTION.getEntry(2)));
33
                   }
34
                   else {
35
                       System.out.println("Hint: " + getBoardEntry(SELECTION.getEntry(1)) + " & " +
                                getBoardEntry(SELECTION.getEntry(2)) + " & " +
36
37
                                getBoardEntry(SELECTION.getEntry(3)));
38
39
                   System.out.print("\nPlease Enter a slot number for any card: ");
40
                   scan.next();
41
               }
42
               isNum = false; //resetting isNum to false for next use
43
44
           return switch (selection) {
45
               case 2 -> 2;
               case 3 -> 3:
46
47
               case 4 -> 4;
48
               case 5 -> 5;
49
               case 6 -> 6;
               case 7 -> 7;
50
51
               case 8 -> 8;
               case 9 -> 9;
52
53
               default -> 1:
54
           };
55
56
57
       private void demonstrationMode() // allows user to increment through a simulated game of Elevens
   step by step
58
59
           Scanner scan = new Scanner(System.in);
60
           while(getBoardLength() > 0) {
               String enterkey = "\nPress 'Enter' key to increment through Demonstration Mode.";
61
62
               System.out.print(enterkey);
63
               enterkey = scan.nextLine();
               System.out.print(enterkey);
64
65
               if(enterkey.equals("")) {
                   displayBoard(BOARD);
66
67
                   AListArray<Integer> SELECTION = checkPossibleMoves(BOARD);
68
                   if (checkPossibleMoves(BOARD).isEmpty()) {
                       System.out.println("----STALEMATE----\n");
69
70
                       menu();
71
                   } else if (checkPossibleMoves(BOARD).getLength() < 3) {</pre>
                       if (getBoardEntry(SELECTION.getEntry(1)).equalEleven(getBoardEntry(SELECTION.
72
   getEntry(2))) == 1) {
73
                            System.out.println("----Removed: "+getBoardEntry(SELECTION.getEntry(1))+" & "
```

```
+getBoardEntry(SELECTION.getEntry(2))+"----");
  74
                                                      replaceCards(SELECTION.getEntry(1), SELECTION.getEntry(2));
  75
  76
                                              }
  77
                                      } else {
                                              if (getBoardEntry(SELECTION.getEntry(1)).equalJQK(getBoardEntry(SELECTION.
  78
       getEntry(2)),
  79
                                                              getBoardEntry(SELECTION.getEntry(3))) == 1) {
                                                      System.out.println("----Removed: " + getBoardEntry(SELECTION.getEntry(1)) +
  80
        " & "
                                                                      + getBoardEntry(SELECTION.getEntry(2))
  81
  82
                                                                      + " & " + getBoardEntry(SELECTION.getEntry(3)) + "----");
                                                      replaceCards(SELECTION.getEntry(1), SELECTION.getEntry(2), SELECTION.getEntry
  83
        (3));
                                              }
  84
                                     }
  85
                              }
  86
                       }
  87
  88
               }
  89
  90
               private void menu() // Elevens menu functionality
  91
  92
                       Scanner scan = new Scanner(System.in);
                       System.out.print("Enter 'r' to replay your moves, 'p' to play again or any other key to exit
  93
        : ");
  94
                        String selection = scan.next();
  95
                       if(selection.equals("r") || selection.equals("R")) {
  96
                               replaySteps();
  97
                               menu();
  98
  99
                       else if(selection.equals("p") || selection.equals("P")) {
100
                               newBoard();
101
                               game();
102
                       }
103
                       else {
104
                               System.exit(0);
105
               }
106
107
108
               public void game() // Elevens main game functionality
109
110
                       Scanner scan = new Scanner(System.in);
111
                                                                                                 ------Welcome to Elevens
                       System.out.println("\n-----
                               ----");
                       System.out.println("Rules");
112
113
                       System.out.println("Select two cards that add to eleven or a Jack, Queen, King combination."
       );
114
                       System.out.println("If you can remove all the cards from the board and the deck you Win!\n");
115
                       System.out.print("Enter 'p' to play or 'd' for demonstration mode: ");
116
117
                       String selection = scan.next();
                       if(selection.equals("d") || selection.equals("D")) {
118
119
                               demonstrationMode();
120
121
                       int first, second, third;
122
                       while(getBoardLength() > 0) {
123
                               displayBoard(BOARD);
                               if(checkPossibleMoves(BOARD).isEmpty()) {
124
125
                                       System.out.println("---STALEMATE----\n");
126
                                      menu();
127
                               if (getBoardEntry(first = selectCard()).getRankValue() <= 10) {</pre>
128
                                       System.out.println("\nPlease select a second card to add to Eleven");
129
                                       if (getBoardEntry(first).equalEleven(getBoardEntry(second = selectCard())) == 1) {
130
                                              System.out.println("\n----Removed: "+getBoardEntry(first) + " & " + getBoardEntry
131
        (second) +"----");
132
                                              replaceCards(first, second);
133
                                      } else {
                                              System.out.println("\n" + getBoardEntry(first) + " & " + getBoardEntry(second) +
134
        " selected.");
135
                                              System.out.println("----They don't add to Eleven. Please try again----");
136
                                      }
137
                               } else if (getBoardEntry(first).getRankValue() >= 11) {
                                       System.out.println("\nPlease select two more different picture cards.");
138
                                        \textbf{if} \ (\texttt{getBoardEntry(first).equalJQK(getBoardEntry(second = \texttt{selectCard())}, \ \texttt{getBoardEntry}} \\ ), \ \ \textbf{getBoardEntry(second = \texttt{selectCard())}}, \ \ \textbf{getBoardEn
139
        (third = selectCard())) == 1) {
140
                                              System.out.println("\n----Removed: "+getBoardEntry(first) + ", " + getBoardEntry(
```

$File-C: \label{local-composition} File-C: \label{local-composition} I like - C: \label{local-composition}$

```
140 second) +
                                ", " + getBoardEntry(third) +"----");
141
142
                        replaceCards(first, second, third);
143
                    } else {
144
                       System.out.println("\n" +getBoardEntry(first) + ", " + getBoardEntry(second) +
145
                                getBoardEntry(third) +" selected.");
                        System.out.println("----Not a Jack, Queen and King. Please try again----");
146
147
               }
148
149
            }
150
            System.out.println("\n----Congratulations! You've Won!----\n");
151
            menu();
        }
152
153
154
        public static void main(String[] args) // main to instantiate a new game of Elevens
155
156
            ElevensGame elevens = new ElevensGame();
157
            elevens.game();
        }
158
159 }
160
```