Driver Class

```
/**************************************
Author: Sergiy Kolodyazhnyy
Course: CS 2050
Date: Sept 16 2015
Instructor: Prof Gurka
Java version: OpenJDK, 1.7.0
IDE: nano text-editor and java compiler
**************************************/
import javax.swing.JOptionPane;
import java.io.PrintWriter;
import java.io.FileWriter;
import java.io.IOException;

public class cs2hw3Driver
{
    public static void main (String[] args) throws IOException
    {

    String myName = "Sergiy Kolodyazhnyy";
        String outputFilePath = getFileName();
    FileWriter outputFile = new FileWriter(outputFilePath);
    PrintWriter writeToFile = new PrintWriter(outputFile);
        int numsArr [] = getNums( );
    cs2hw3Lotto Lotto  = new cs2hw3Lotto (numsArr);

    int [] statistics = new int[5];
    int[] sums = new int[5];
    printHeader(writeToFile,numsArr,myName);
    for (int x = 1; x<=10; x++)
    {     statistics = Lotto.playTillJackpot();
          printToFile(writeToFile,statistics,x);
          for (int y = 0; y<5; y++)
          {
                sums[y] += statistics[y];
          }
    }
    printAverages(writeToFile,sums);
    writeToFile.close();

    }
//---------------------------------------------------------------------
    public static int[]  getNums(  )
    {
    // had to declare instance of the Lotto object here to use
    // hasDuplicates method
    cs2hw3Lotto Lottery = new cs2hw3Lotto();

        String numString = "";
    int arr[];
    // loop until the user gives right input
    while ( true )
{   numString = JOptionPane.showInputDialog("Please input 6 unique digits , from 1
to 41, comma-separated");
    // check if we have illegal characters with regex patter matching
        if(numString.matches(".*[^0-9,].*"))
         {
                System.out.println("REgex works");
                continue;
         }

    // protective feature - removing stray blanks from user's input
        numString = numString.replaceAll("\\s+","");
```

```
    62:
    63:
    64:     // split the string into array of strings using comma as delimiter
    65:         String [] numArrStr = numString.split(",");
    66:     //declare int array of same size as numeric string array
    67:         arr = new int[numArrStr.length];
    68:     // check if the length is wrong (meaning user missed comma or put too many
nums)
    69:     if ( numArrStr.length < 6 || numArrStr.length > 6)
    70:           continue;
    71:     // If above conditions are OK, fill the array of ints by parsing array
    72:     // of strings to int
    73:         for (int i = 0 ; i < numArrStr.length; i++)
    74:             {
    75:                 arr[i] = Integer.parseInt(numArrStr[i]);
    76:             }
    77:
    78:     // Check if we have duplicate numbers
    79:     if ( Lottery.hasDuplicates(arr))
    80:     {     continue;
    81:     }
    82:     else
    83:     {     break;
    84:     }
    85:
    86: }
    87:         return arr;
    88:     }
    89: //=================================================================
    90:     // as the name suggests, here we are getting the filename
    91:     // where the output will be stored
    92:     public static String getFileName ()
    93:     {
    94:         return JOptionPane.showInputDialog("Please input filename where data will
be stored");
    95:     }
    96:
    97: //=================================================================
    98: // the three methods bellow output to the file.  printToFile is called repeatedly
    99: // while printHeader and printAverages serve purpose at the beginnning and ending
   100: // of the 10 games
   101:     public static void printToFile( PrintWriter statsFile,int[] stats, int
game   )
   102:     {
   103:           statsFile.printf("%-2d: %-10d %-10d %-10d %-10d %-
10d\n",game,stats[0],stats[1],stats[2],stats[3],stats[4]);
   104:     }
   105:
   106:     public static void printHeader( PrintWriter statsFile,int[] array,String
author  )
   107:     {
   108:           statsFile.println(author);
   109:           statsFile.print("User Input: ");
   110:           for (int i = 0; i< array.length; i++)
   111:                 statsFile.print(array[i] + " ");
   112:           statsFile.printf("\n%s %-10s %-10s %-10s %-10s %-
10s\n","Game#","Rolls","Match 3","Match 4","Match 5","Payout");
   113:
   114:     }
   115:     public static void printAverages( PrintWriter statsFile, int sums [] )
   116:     {
   117:       statsFile.println("Average values:");
   118:      for (int i=0; i<5;i++)
```

```
119:       statsFile.printf("\t%-10d",sums[i]/10);
120:     }
121: }
```

Lotto Class

```
 1: import java.lang.Math;
 2: import java.util.Random;
 3: import java.util.Arrays;
 4:
 5: public class cs2hw3Lotto
 6: {
 7:     // MAX and MIN are made private and final, because
 8:     // we want these to remain constant and unalterable
 9:     private final int MAX =  41;
10:     private final int MIN = 1;
11:
12:     int [] userNums;
13: //  int stats [] = new int[5];
14:
15:     Random rand = new Random();
16:
17:     public cs2hw3Lotto()
18:     {
19:
20:     }
21:
22:     // Constructor for our Lotto object
23:     public cs2hw3Lotto (int [] a)
24:     {
25:       userNums = a;
26:     }
27:
28:     public void checkUserInput()
29:     {
30:           for (int i = 0 ; i < userNums.length; i++)
31:                 System.out.print(userNums[i]);
32:     }
33: //=========================================================================
34:     // random number generator method
35:     public int getRandomInt ()
36:     {
37:           int out;
38:           out = rand.nextInt((MAX - MIN) + 1) + MIN;
39:           return out;
40:           }
41:
42: //=========================================================================
43:     // a helper method that tests whether an array has duplicates
44:     // will be used to test user's input as well as
45:     // the generated numbers
46:     public boolean hasDuplicates (int[] a)
47:     {
48:           boolean result = false;
49:           Arrays.sort(a);
50:
51:           for(int i = 1; i < a.length; i++)
52:             {
53:                 if(a[i] == a[i - 1])
54:                     {
55:                         result = true;
56:                     }
```

```
 57:             }
 58:           return result;
 59:
 60:           }
 61: //=============================================================================
 62:
 63:     public int [] playTillJackpot()
 64:     {
 65:           // stats array hold total number of plays,
 66:           // how many times we matched 3 numbers,
 67:           // 4 numbers, 5 numbers, and total payouts in
 68:           // that order respectivelly
 69:           int stats [] = new int[5];
 70:           int countMatched = 0;
 71:           int lottoNums [] = new int[6];
 72:           while (true)
 73:                     {
 74:                 // generate numbers
 75:                         for (int i = 0; i < 6; i++)
 76:                             {
 77:                                 lottoNums[i] = getRandomInt();
 78:                             }
 79:                 // re-run number generator if we have duplicate numbers
 80:                         if (hasDuplicates(lottoNums))
 81:                             {
 82:                                 continue;
 83:                             }
 84:                 stats[0] ++ ;
 85:                         //compare userNums to lottoNums
 86:                         // count matched numbers
 87:                         for (int j = 0; j < 6; j++)
 88:                             {
 89:                                 for (int k = 0 ; k < 6; k++)
 90:                                     {
 91:                                         if (lottoNums[k] == userNums[j])
 92:                                             countMatched++;
 93:                                     }
 94:                             }
 95:                 switch (countMatched)
 96:                 {
 97:                       case 3: stats[1]++;break;
 98:                       case 4: stats[2]++;break;
 99:                       case 5: stats[3]++;break;
100:                       default: break;
101:                 }
102:
103:                           if (countMatched == 6)
104:                               {
105:                                  // System.out.println("Jackpot");
106:                       //System.out.println("Played " + stats[0] + " times");
107:                       stats[4] = stats[1]*10 + stats[2]*50 + stats[3]*1000;
108:                       //System.out.println("Payout " + stats[4]);
109:                                 break;
110:                             }
111:                           countMatched = 0;
112:
113:     }
114:   return stats;
115:   }
116: }
```

4 sample outputs of the programs

```
Sergiy Kolodyazhnyy
User Input: 2 3 5 7 9 11
Game# Rolls      Match 3    Match 4    Match 5    Payout
1 : 3843544    111972     7526       166        1662020
2 : 842130     24361      1731       39         369160
3 : 1106191    31819      2218       62         491090
4 : 5677071    165613     11194      261        2476830
5 : 13992974   408122     27786      683        6153520
6 : 6427067    187116     12544      269        2767360
7 : 19941591   580036     39606      887        8667660
8 : 7894263    229342     15747      349        3429770
9 : 532722     15676      1016       17         224560
10: 8651854    252251     17133      353        3732160
Average values:
      6890940     200630      13650       308         2997413

Sergiy Kolodyazhnyy
User Input: 11 13 20 25 34 41
Game# Rolls      Match 3    Match 4    Match 5    Payout
1 : 11366903   331567     22631      529        4976220
2 : 3103730    90468      6279       153        1371630
3 : 6909764    201225     13501      291        2978300
4 : 6652683    193933     13016      307        2897130
5 : 1202868    34821      2537       44         519060
6 : 9821277    285482     19465      454        4282070
7 : 5475779    159680     10849      237        2376250
8 : 12807005   373855     25422      613        5622650
9 : 6809572    198739     13451      337        2996940
10: 1817271    52954      3606       84         793840
Average values:
      6596685     192272      13075       304         2881409

Sergiy Kolodyazhnyy
User Input: 2 5 9 13 40 41
Game# Rolls      Match 3    Match 4    Match 5    Payout
1 : 820123     23914      1580       39         357140
2 : 473346     13639      910        23         204890
3 : 2405562    69777      4794       104        1041470
4 : 1146747    33129      2223       57         499440
5 : 11286532   329050     22272      558        4962100
6 : 1151675    33840      2283       55         507550
7 : 538543     15344      1069       24         230890
8 : 777072     22735      1561       37         342400
9 : 761373     22095      1478       32         326850
10: 734498     21391      1477       41         328760
Average values:
      2009547     58491       3964        97          880149

Sergiy Kolodyazhnyy
User Input: 1 3 13 21 39 40
Game# Rolls      Match 3    Match 4    Match 5    Payout
1 : 936098     27336      1809       51         414810
2 : 1843732    53722      3778       94         820120
3 : 6852636    199452     13580      303        2976520
4 : 562567     16726      1108       31         253660
5 : 99587      2894       212        2          41540
6 : 5523202    160393     11089      241        2399380
7 : 5260288    152814     10366      223        2269440
8 : 3087521    89581      6019       136        1332760
```

```
9 : 2678326    77910      5271       134        1176650
10: 3489292    101339     6906       166        1524690
Average values:
      3033324     88216       6013        138         1320957
```