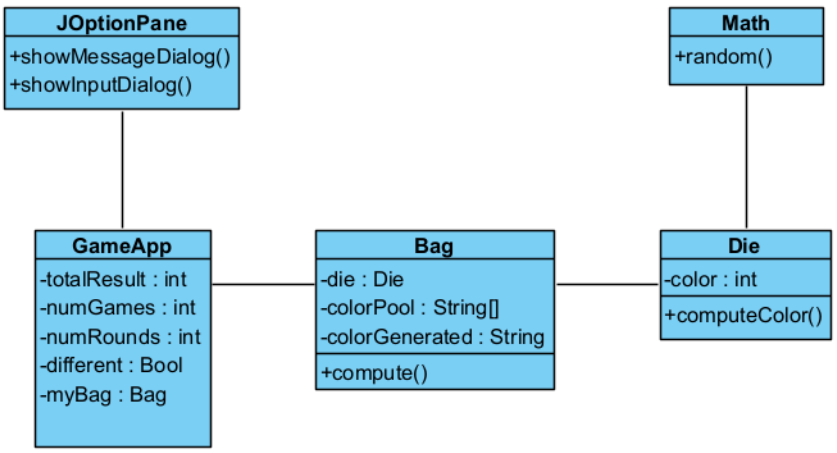
Erick Lopes
2219550

5 - Cube Game
0 - 4 Rounds
5 - Ask player for games

IPO

Input	Process	Output
Enter num of games	Generate random number Each number correlates to a color This process repeat itself for 4 rounds If first and last color are the same add 50 to total If all the colors are different subtract 26 from total If all the colors are the same add 110 total Once a game finishes display sub total of points Repeat the process for as many games as inputed	Display total points of all games How many games were played

Class Diagram



Decision making

It was decided to use 3 different classes to represent the game in order to create an application that can be easily modified and re-strategize if needed at any point.

This separation keeps the cohesion and responsibility of each class within each class.
For example, it's not a responsibility of the Bag to generate the Die with a color, but it is the responsibility of the class to keep the information of the color that a Die has.

The class GameApp is a class that controls the behavior and rules of the game, controlling the score and has the responsibility to exhibit the result to the player.

This way each class is disattached enough from the other where a change in one class will not impact the other class behavior.

Source Code

```
         ////////////////////
         //  GameApp.java //
         //  Erick Lopes  //
         //    2219550    //
         //   08/03/2023  //
         ////////////////////
import javax.swing.JOptionPane;
import java.util.Arrays;

         public class GameApp{
  public static void main (String args[]){

    // Declare and create objects
    int totalResult = 0, numGames, numRounds = 0;
    Boolean different = true;
    Bag myBag = new Bag();

    // Explanation of the rules
   JOptionPane.showMessageDialog(null, "The points are assigned as follows:" +
                              "\n   50 points are awarded when a cube with the same colour is drawn
in the first and last rounds" +
                              "\n   -26 (minus 26) points are awarded when a cube with a different
colour is drawn in every round" +
                              "\n   In addition, 110 bonus points are awarded when a cube with the
same colour is drawn in all the rounds." +
                              "\nIf the above rules are not met, zero points are awarded for that
game." );

    // Input
    numGames = Integer.parseInt(JOptionPane.showInputDialog(null, "Select the quantity of
games you wish to play"));

    do{

      //Declare //The following 2 lines are declared here in order to refresh each new round
      int  roundResult = 0;
      String [] color = new String[4];
```

```
    //In order to generate an optimization its used an Array to hold all the Dice generated in all rounds
    for (int i = 0; i < 4; i++) {
      // Process to get the color of a rolled die
      myBag.compute();
      color[i] = myBag.getColorGenerated();

      //Input used to test of each rule
      //color[0] = "Green"; color[1] = "Green"; color[2] = "Green"; color[3] = "Green";
    }

    //Application of the first rule that says if First and Last Die are equal in color add 50 points
    if(color[0] == color[color.length-1]){
      roundResult += 50;
    }

    //Application of the third rule that says if all Dice are the same in color add 110 points
    if(color[0] == color[1] && color[1] == color[2] && color[2] == color[3]){
      roundResult += 110;
    }

    //Validation of the second rule that says if all Dice are different in color subtract 26 points
    different = true;
    for (int i = 0; i < color.length - 1; i++) {
      for (int j = i+1; j < color.length; j++){
        if(color[i] == color[j]){
          different = false;
          break;
        }
      }
    }

    //Application of the second rule
    if(different == true){
      roundResult -= 26;
    }

    //Score total of all games played being registered and updating number of rounds played
```

```
        totalResult += roundResult;
        numRounds++;

        //Message with the result of the game played
        JOptionPane.showMessageDialog(null, "The dice drawn in this Game were\n
"+color[0]+"\n  "+color[1]+"\n  "+color[2]+"\n  "+color[3]+"\n\n"+"Score of the "+
numRounds +" game is: "+ roundResult);

    }while( numGames > numRounds );

    //Message with the result of all the games played
    JOptionPane.showMessageDialog(null, "Total score of all games is: "+ totalResult +
                                    "\nThe Total of games chosen to be played was: "+ numGames);
  } // main
} // class
```

```
                    ////////////////////
                    //        Bag.java   //
                    //  Erick Lopes  //
                    //        2219550         //
                    //   08/03/2023  //
                    ////////////////////
```

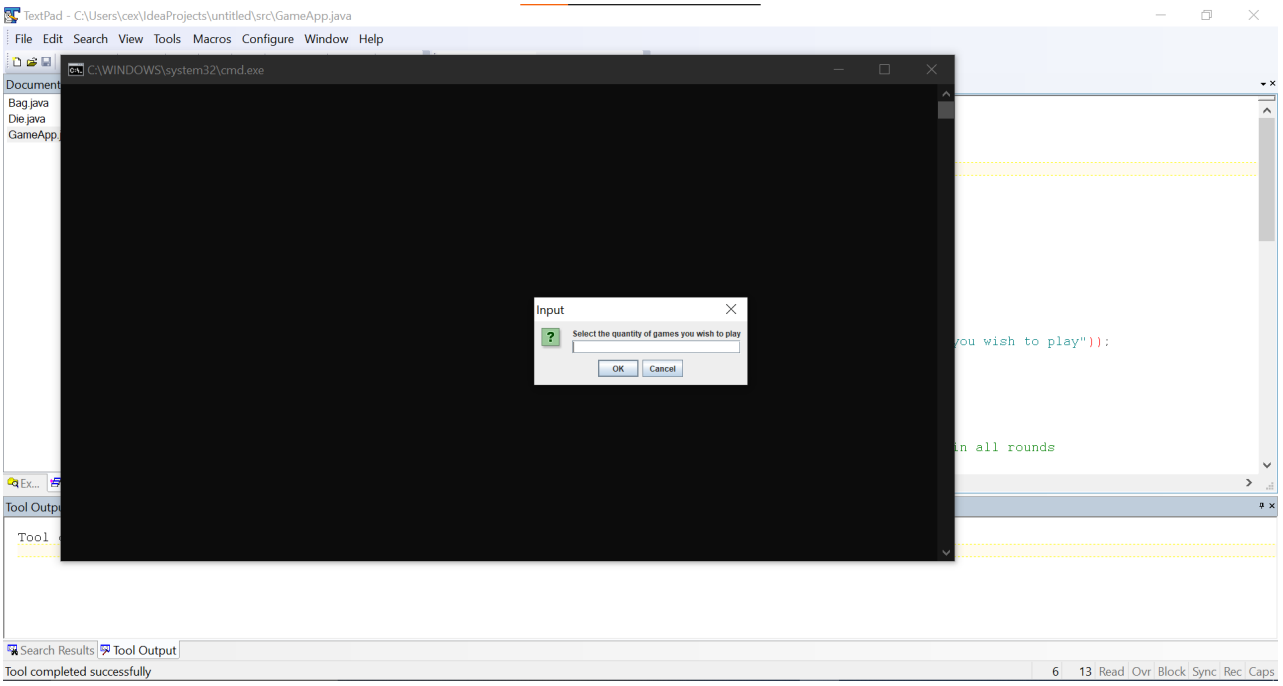
```
public class Bag {
   // Declare variables and pool of colors to be chosen
   private String colorGenerated;
   private String [] colorPool =
{"Green","Green","Green","Green","Yellow","Yellow","Orange","white"};

   //Whenever we create a bag we create a die that will be selected
   Die die = new Die();

   // Compute the color of the die decided
   public void compute(){
         die.computeColor();
         colorGenerated = colorPool[die.getColor()];
   }

   // Getters and Setters
   public String getColorGenerated(){
         return colorGenerated;
   }

}
```

```
//////////////////
//   Die.java     //
//  Erick Lopes  //
//        2219550         //
//   08/03/2023  //
//////////////////
```

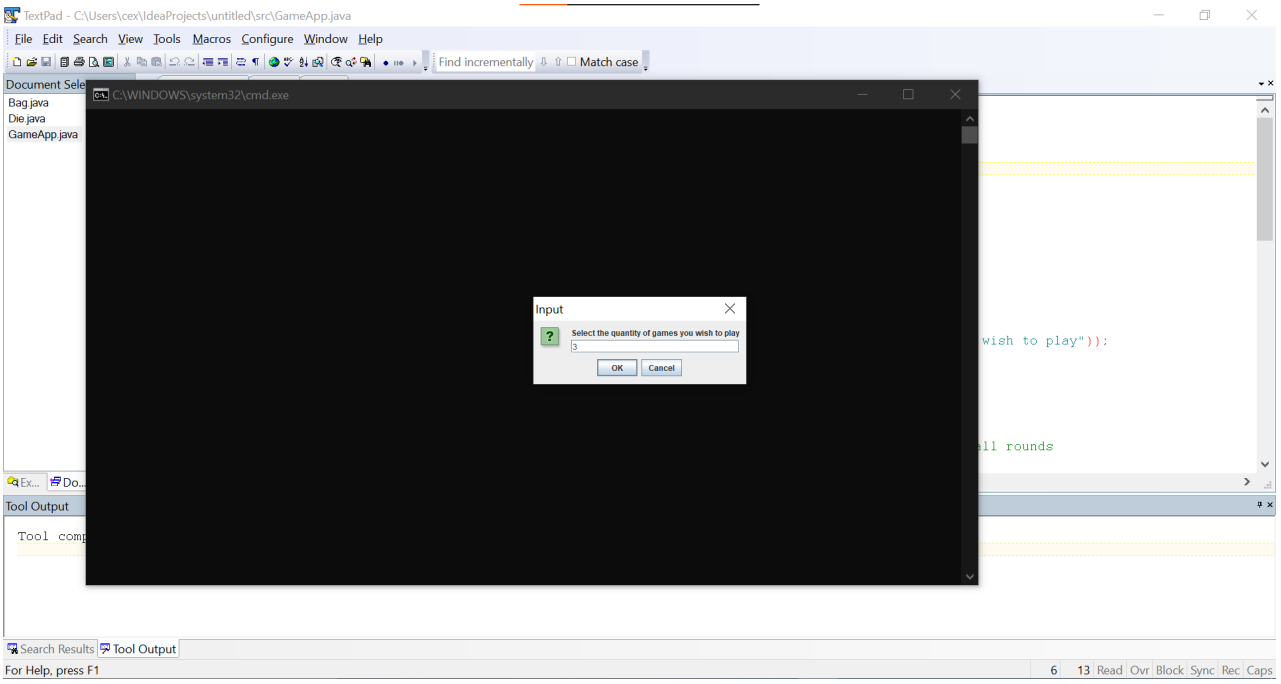
```
// To keep code organized and coherent, it was decided to create an object that would represent the die;
// This way if any new implementation or change is needed it can be done without interference with the rest of the code;
import java.lang.Math;
public class Die {
	// Declare variables
	private int color; // this variable is used to store the number that represents the color of the die

	// Generate a random number from 0 to 7
	public void computeColor() {
	//Select a random number to represent the color of the die
	color = (int)(Math.random() * 8);
	}

	// Getters and Setters

	public int getColor() {
	return color;
	}
}
```

Screenshots of the application

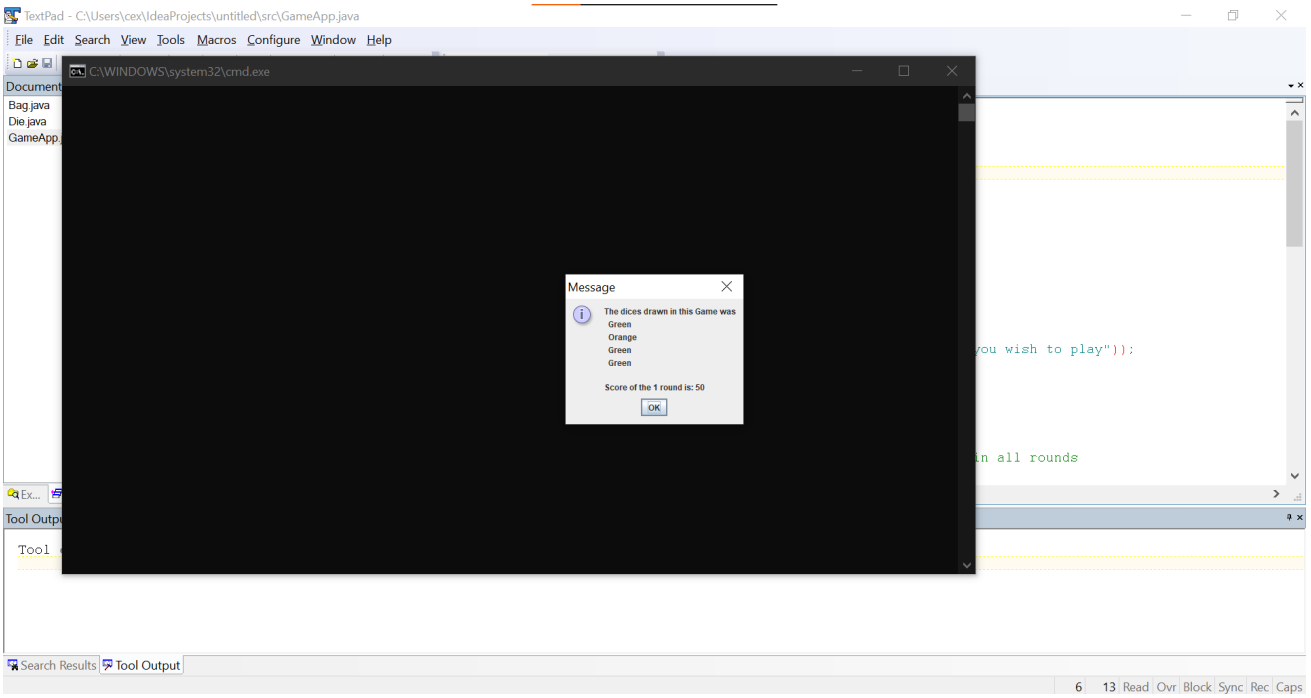
1- First screen of the application asking for the input of how many games should be played



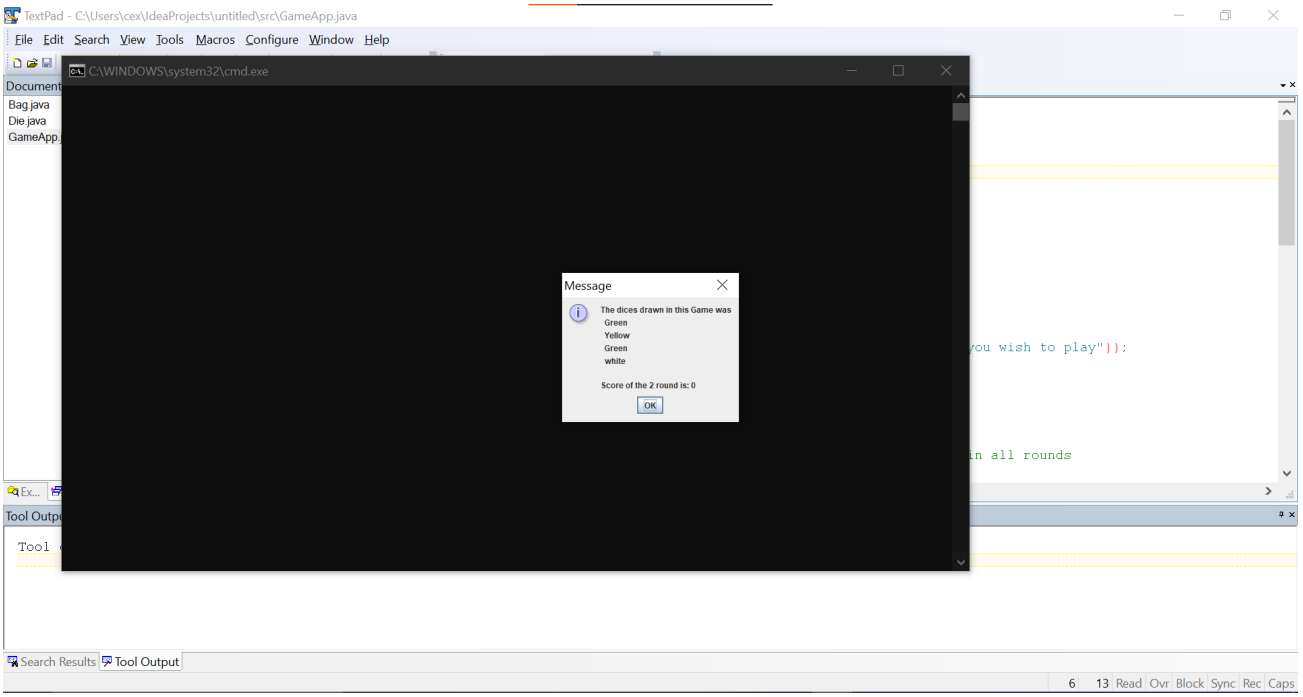
2- Screen of the application with number of games that should be played



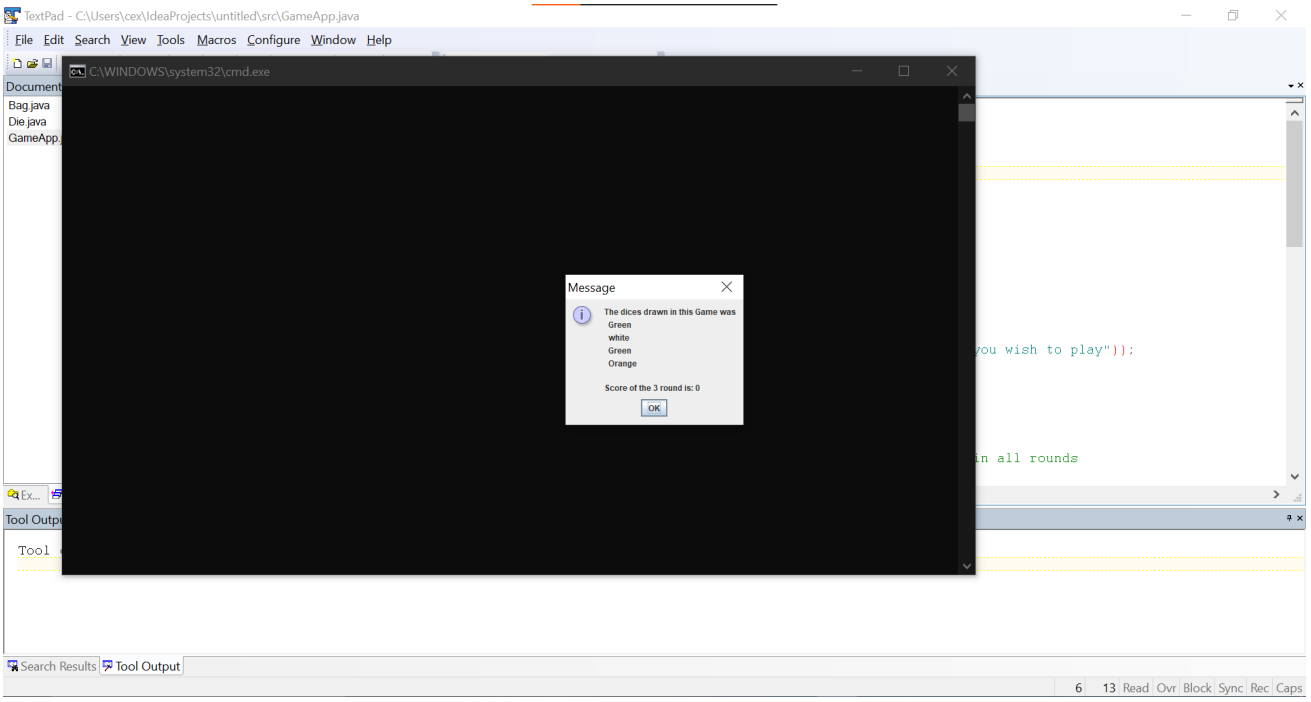
3- Result of the first game exhibiting the die drawn in each of the rounds and the result of that game



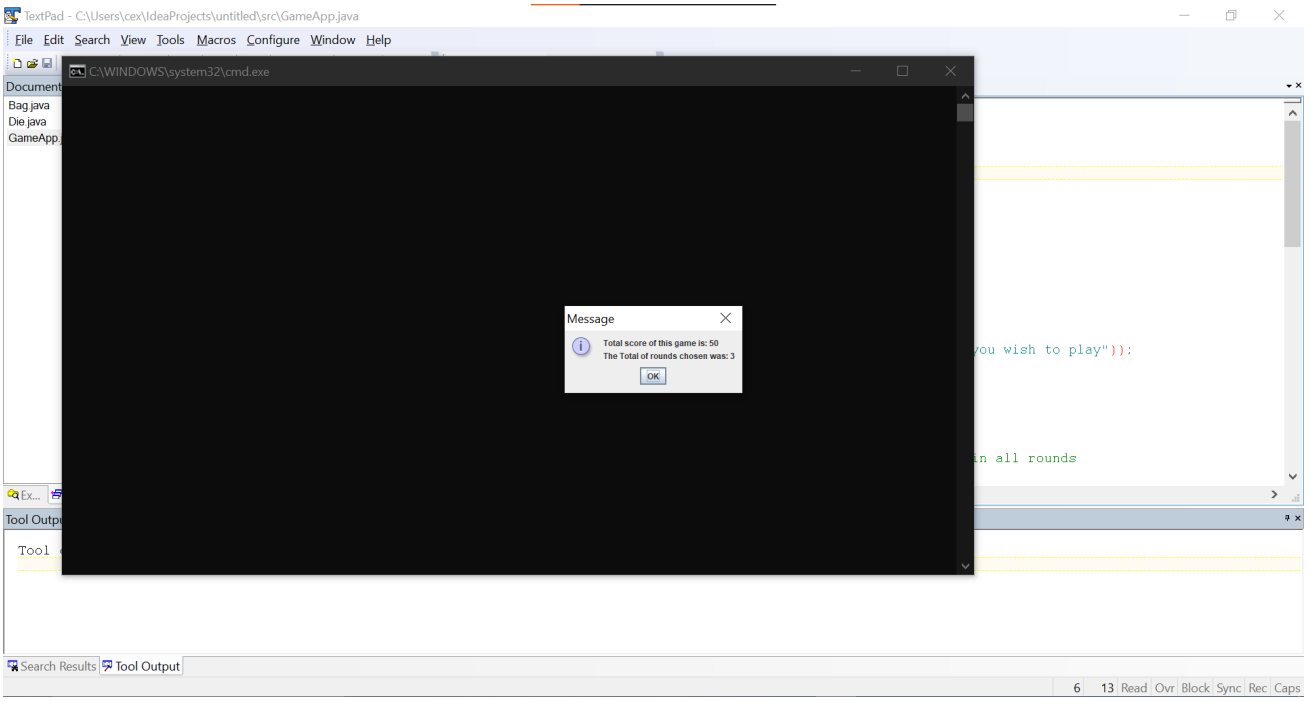
4- Result of the second game



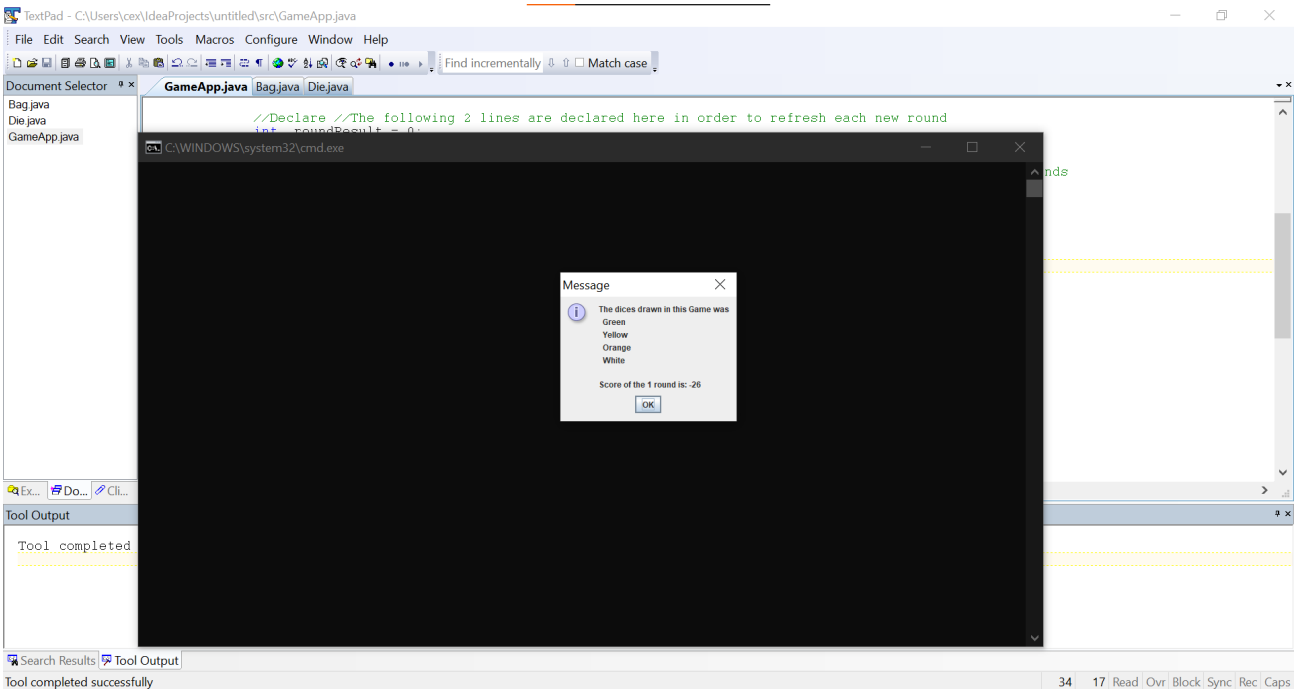
5- Result of the third game



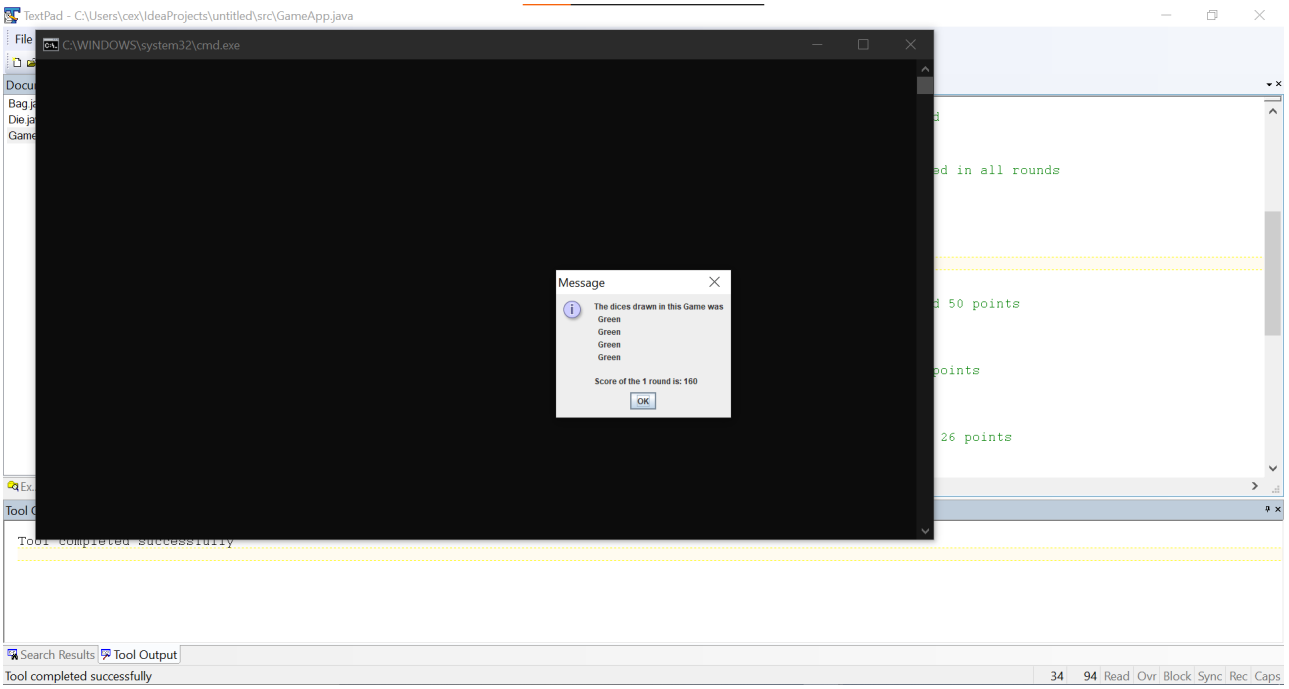
6- Screen showing the result of the 3 games total score and how many games were played



7- Screen showing the rule case each die is different from each other



8- Screen showing result if each die is the same as each other



9- Specification and implementation of each rule

```
//Application of the first rule that says if Fisrt and Last Die are equal in color add 50 points
if(color[0] == color[color.length-1]){
    roundResult += 50;
}

//Application of the third rule that says if all Dice are different in color add 110 points
if(color[0] == color[1] && color[1] == color[2] && color[2] == color[3]){
    roundResult += 110;
}

//Validation of the second rule that says if all Dice are different in color subtract 26 points
different = true;
for (int i = 0; i < color.length - 1; i++) {
    for (int j = i+1; j < color.length; j++){
        if(color[i] == color[j]){
            different = false;
            break;
        }
    }
}

//Application of the second rule
if(different == true){
    roundResult -= 26;
}
```