Drew Sweeney

Homework 6

Dr. Hu

March 3, 2022

Summary:

Logically going through this homework was very fun. I like seeing my code animated it is very enjoyable. A problem I had was with the HashMap. As a default its value was null, luckily, I was able to find a putifabsent method that fixed the issue for me. Otherwise making the program run for me was not an issue I had to play around with my lines and understand the coordinate plane a little but after I got the kinks out, I did not have much issue with the rest of the program.

Ball Animation:

```
package RacingGame;


import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.util.*;
import java.util.List;
import javax.swing.Timer;
import javax.swing.*;
import java.awt.geom.Ellipse2D;
```

```
public class BallAnimation
{
           RacingRules racingRules = new RacingRules();
           
    public static void main(String[] args)
    {
        new BallAnimation();
    }

    public BallAnimation()
    {
        EventQueue.invokeLater(new Runnable()
        {
            @Override
            public void run()
            {
                try
                {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
                } catch (ClassNotFoundException | InstantiationException |
IllegalAccessException | UnsupportedLookAndFeelException ex)
                {
                    ex.printStackTrace();
                }
            JFrame frame = new JFrame("Racing");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setSize(640,480);
            frame.add(new RacingGame());
            frame.setLocationRelativeTo(null);
            frame.setVisible(true);

         }
      });
   }

   public class RacingGame extends JPanel
   {
        private int x1 = racingRules.getRacer1Value();
        private int x2 = racingRules.getRacer2Value();

        public RacingGame()
        {
                Timer timer = new Timer(40, new ActionListener()
                {
                        @Override
                        public void actionPerformed(ActionEvent e)
                        {
```

```
                        if (x1 < 495 && x2 < 495)
                        {
                            moveBall1();
                            moveBall2();
                            repaint();
                            RacingRules racingRules = new RacingRules();
                        }
                }
        });
        timer.start();
    }
    
    protected void moveBall1()
    {
        //if getValue() != 495
        if (x1 != 495)
        {
            x1 = racingRules.getRacer1Value();
            
            //make two move balls
            //only commands should be getValue();
        }
    }
    
    protected void moveBall2()
    {
        if (x2 != 495)
        {
            x2 = racingRules.getRacer2Value();
        }
    }

    @Override
   protected void paintComponent(Graphics g)
    {
    super.paintComponent(g);

                //racer1
                Graphics2D g2d = (Graphics2D) g.create();
                g2d.setColor(Color.RED);
                g2d.fillOval(x1+30, 80, 80, 80);
                g2d.dispose();

                //racer2
                Graphics2D g3d = (Graphics2D) g.create();
                g3d.setColor(Color.BLUE);
                g3d.fillOval(x2+30, 280, 80, 80);
                g3d.dispose();
```

```
                    //finish line
                    Graphics2D g4d = (Graphics2D) g.create();
                    g4d.setColor(Color.BLACK);
                    g4d.setStroke(new BasicStroke(5));
                    g4d.drawLine(600, 0, 600, 480);
                    g4d.dispose();

                    //start line
                    Graphics2D g5d = (Graphics2D) g.create();
                    g5d.setColor(Color.BLACK);
                    g5d.setStroke(new BasicStroke(5));
                    g5d.drawLine(30, 480, 30, 0);
                    g5d.dispose();
    }
  }
}
```

RacingGame:

```
package RacingGame;

import java.util.*;

public class RacingRules
{
	String racer1 = "Racer1";
	String racer2 = "Racer2";
	int maxNumber = 250;

	Map<String, Integer> Racer1 = new HashMap<String, Integer>();
	Map<String, Integer> Racer2 = new HashMap<String, Integer>();
```

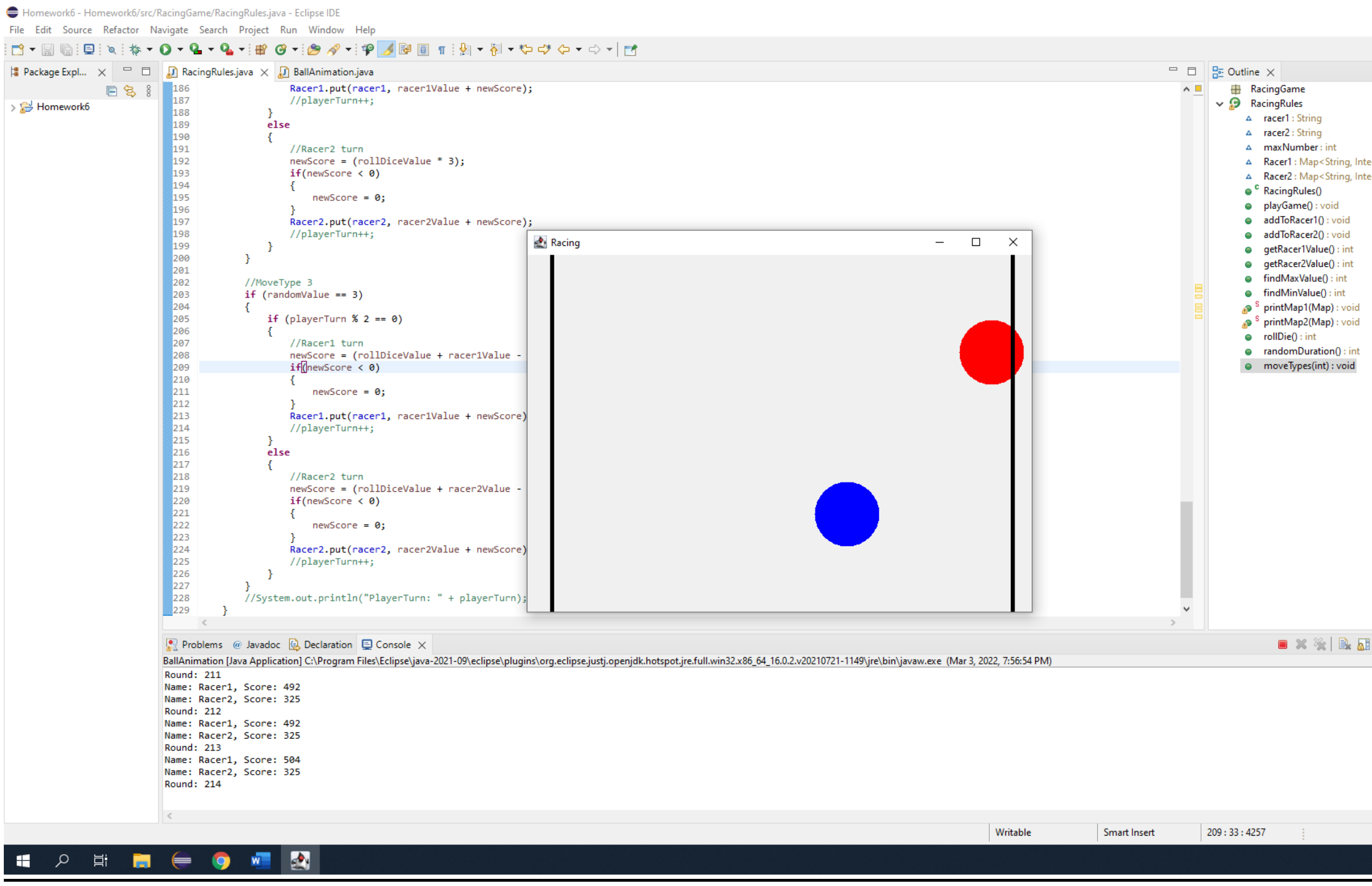
```
public RacingRules()
{
        addToRacer1();
        addToRacer2();
        playGame();

}
public void playGame()
{
        for (int i = 1; i < maxNumber; i++)
        {
                try
                {
                        if (getRacer1Value() < 495 && getRacer2Value() < 495)
                        {
                                moveTypes(i);
                                Thread.sleep(100);
                                printMap1(Racer1);
                                printMap2(Racer2);
                                System.out.println("Round: " + i);
                        }
                } catch (InterruptedException e)
                {
                        e.printStackTrace();
                }
        }
}

public void addToRacer1()
{
        Racer1.putIfAbsent(racer1, 0);
}
public void addToRacer2()
{
        Racer2.putIfAbsent(racer2, 0);
}
public int getRacer1Value()
{
        int score = Racer1.get(racer1);
        return score;
}
public int getRacer2Value()
{
        int score = Racer2.get(racer2);
        return score;
}
public int findMaxValue()
{
        int maxValue1 = (int) (Collections.max(Racer1.values()));
        int maxValue2 = (int) (Collections.max(Racer2.values()));
        if (maxValue1 >= maxValue2)
        {
                return maxValue1;
        }
```

```
		else
		{
			return maxValue2;
		}
	}
	public int findMinValue()
	{
		int minValue1 = (int) (Collections.min(Racer1.values()));
		int minValue2 = (int) (Collections.min(Racer2.values()));
		if (minValue1 > minValue2)
		{
			return minValue1;
		}
		else
		{
			return minValue2;
		}
	}

	public static void printMap1(Map Racer1)
	{
		Iterator iterator = Racer1.entrySet().iterator();
		while (iterator.hasNext())
		{
			Map.Entry mEntry = (Map.Entry)iterator.next();
			System.out.println("Name: " + mEntry.getKey() + ", Score: " +
mEntry.getValue());
		}
	}
	public static void printMap2(Map Racer2)
	{
		Iterator iterator = Racer2.entrySet().iterator();
		while (iterator.hasNext())
		{
			Map.Entry mEntry = (Map.Entry)iterator.next();
			System.out.println("Name: " + mEntry.getKey() + ", Score: " +
mEntry.getValue());
		}
	}

	public int rollDie()
	{
		int dieRoll = 0;
		String temp;

		String numbers []={"1","2","3","4","5","6"};

		Random generator = new Random();
		int random = generator.nextInt(numbers.length);

		temp = numbers[random];
		dieRoll = Integer.parseInt(temp);

		return  dieRoll;
	}
```

```
public int randomDuration()
{
	int number = 0;
	String temp;

	String numbers [] = {"1","2","3"};
	Random generator = new Random();
	int random = generator.nextInt(numbers.length);
	temp = numbers[random];
	number = Integer.parseInt(temp);

	return number;
}

public void moveTypes(int playerTurn)
{
	int rollDiceValue = rollDie();
	int randomValue = randomDuration();
	int newScore;
	int firstPlace = findMaxValue();
	int lastPlace = findMinValue();
	int racer1Value = getRacer1Value();
	int racer2Value = getRacer2Value();

	//MoveType 1
	if (randomValue == 1)
	{
		if (playerTurn % 2 == 0)
		{
			//Racer1 turn
			newScore = (rollDiceValue + racer1Value - firstPlace)/2;
			if(newScore < 0)
			{
				newScore = 0;
			}
			Racer1.put(racer1,racer1Value + newScore);
			//playerTurn++;
		}
		else
		{
			//Racer2 turn
			newScore = (rollDiceValue + racer2Value - firstPlace)/2;
			if(newScore < 0)
			{
				newScore = 0;
			}
			Racer2.put(racer2,racer2Value + newScore);
			//playerTurn++;
		}

	}

	//MoveType 2
	if (randomValue == 2)
```

```java
        {
            if (playerTurn % 2 == 0)
            {
                //Racer1 turn

                newScore = (rollDiceValue * 3);
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer1.put(racer1, racer1Value + newScore);
                //playerTurn++;
            }
            else
            {
                //Racer2 turn
                newScore = (rollDiceValue * 3);
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer2.put(racer2, racer2Value + newScore);
                //playerTurn++;
            }
        }

        //MoveType 3
        if (randomValue == 3)
        {
            if (playerTurn % 2 == 0)
            {
                //Racer1 turn
                newScore = (rollDiceValue + racer1Value - lastPlace)/2;
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer1.put(racer1, racer1Value + newScore);
                //playerTurn++;
            }
            else
            {
                //Racer2 turn
                newScore = (rollDiceValue + racer2Value - lastPlace)/2;
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer2.put(racer2, racer2Value + newScore);
                //playerTurn++;
            }
        }
        //System.out.println("PlayerTurn: " + playerTurn);
    }
}
```

<u>OutPut:</u>



```java
                Racer1.put(racer1, racer1Value + newScore);
                //playerTurn++;
            }
            else
            {
                //Racer2 turn
                newScore = (rollDiceValue * 3);
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer2.put(racer2, racer2Value + newScore);
                //playerTurn++;
            }
        }

        //MoveType 3
        if (randomValue == 3)
        {
            if (playerTurn % 2 == 0)
            {
                //Racer1 turn
                newScore = (rollDiceValue + racer1Value -
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer1.put(racer1, racer1Value + newScore)
                //playerTurn++;
            }
            else
            {
                //Racer2 turn
                newScore = (rollDiceValue + racer2Value -
                if(newScore < 0)
                {
                    newScore = 0;
                }
                Racer2.put(racer2, racer2Value + newScore)
                //playerTurn++;
            }
        }
        //System.out.println("PlayerTurn: " + playerTurn);
    }
```

```
BallAnimation [Java Application] C:\Program Files\Eclipse\java-2021-09\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe  (Mar 3, 2022, 7:56:54 PM)
Round: 211
Name: Racer1, Score: 492
Name: Racer2, Score: 325
Round: 212
Name: Racer1, Score: 492
Name: Racer2, Score: 325
Round: 213
Name: Racer1, Score: 504
Name: Racer2, Score: 325
Round: 214
```