Assignment 9 – Mercer Road Race
CSC 204 Spring 2017
Due Friday April 28, 2017

For this assignment you are going to create the “Mercer Road Race” with several instances of your MercMen class. You will animate a group of MercMen ‘running’ across the screen. You will accomplish this through a collection of three Java files: MercerRoadRace (the main driver program), RacerComponent (the JComponent responsible for calling all of the MercMen’s draw() methods), and of course your MercMan with a few modifications. Each of these file is described below. Do your work in an Eclipse project named: MercerRoadRace.

Enhancements to the MercMan class:

First off, in your MercMan (MM) class definition, you need to modify your MM’s mouth. We want to have two choices for his/her mouth: smile and frown. There needs to be a new boolean instance variable named “smile” that indicates whether MM is smiling or not (true = smile, false = frown). In the constructor always set “smile” to true (we all start out smiling). You will have to write a little more code in your draw() method that uses “smile” to decide how to draw the mouth (smiling or frowning). You should also create a method that sets the “smile” variable so you can change a MM’s mouth during the animation.

> **setSmile(boolean s);**
> You need to create a public method named setSmile() that is passed a boolean and sets the instance variable smile to this new value. Nothing is returned.

You will also need to add four more instance methods that allow us to find out where MM is and to set his/her location elsewhere:

> **getX();**
> You need to create a public method named getX() that is passed no arguments and returns an integer equal to the current x value of the MercMan. This type of method is called a “getter”.
>
> **getY();**
> You need to create a public method named getY() that is passed no arguments and returns an integer equal to the current y value of the MercMan.
>
> **setX();**
> You need to create a public method named setX() that is passes an integer and sets the instance variable x to this new value. Nothing is returned.
>
> **setY();**
> You need to create a public method named setY() that is passed an integer and sets the instance variable y to this new value. Nothing is returned.

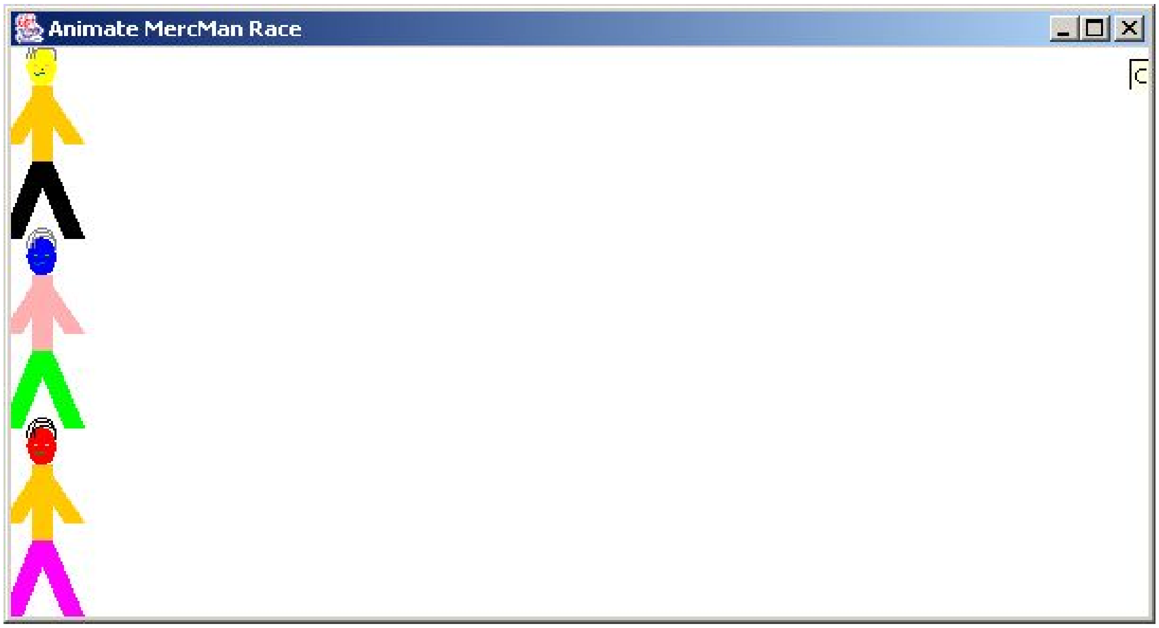
Finally, make sure you have a MercMan constructor that is passed (x,y,width,height) and will generate random colors for the various MercMan parts.

MercerRoadRace:
You are to write a "driver" program to test the enhancements of your MercMan class. When this program begins to run it should print a welcome message to your users (in the console, not on a graphics window). It should then prompt the user to enter the number of racers for this Mercer Road Race. Only allow the number of racers to be in the range:

$$1 \leq \text{number\_of\_racers} \leq 20.$$

Continue to prompt the user until you get a valid number of racers. Now you should proceed to create a graphics window (JFrame) that is 800 pixels wide, 600 pixels tall, and has a jazzy little title that includes your name. You should view this window as a race track that runs from left to right. You should also view this race track as having *number_of_racers* lanes for running. Each "lane" is as tall as the height of the screen divided by the number of racers and runs from left to right, one on top of the next.

Next your program should create *number_of_racers* randomly colored MercMan objects and place them into an ArrayList. Calculate reasonable proportions for the MercMan's height and width to fit into a lane on the race track. As you create the MercMen, you should calculate appropriate offset values (x,y) as to place them at the left edge of the screen (one on top of the next). For example, if *number_of_racers* were 3, your program should produce 3 MercMen that would be placed on an initial screen like:

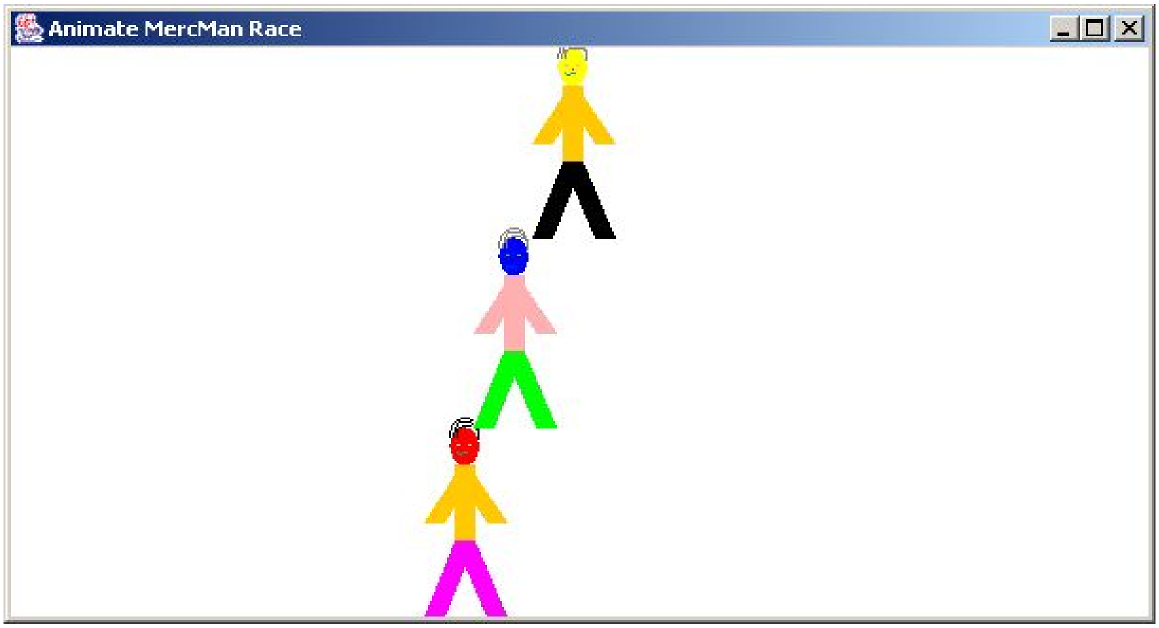


This ArrayList of MercMen should be passed into the RacerComponent when you create it (details about RacerComponent below) and you should add the RacerComponent to the JFrame.

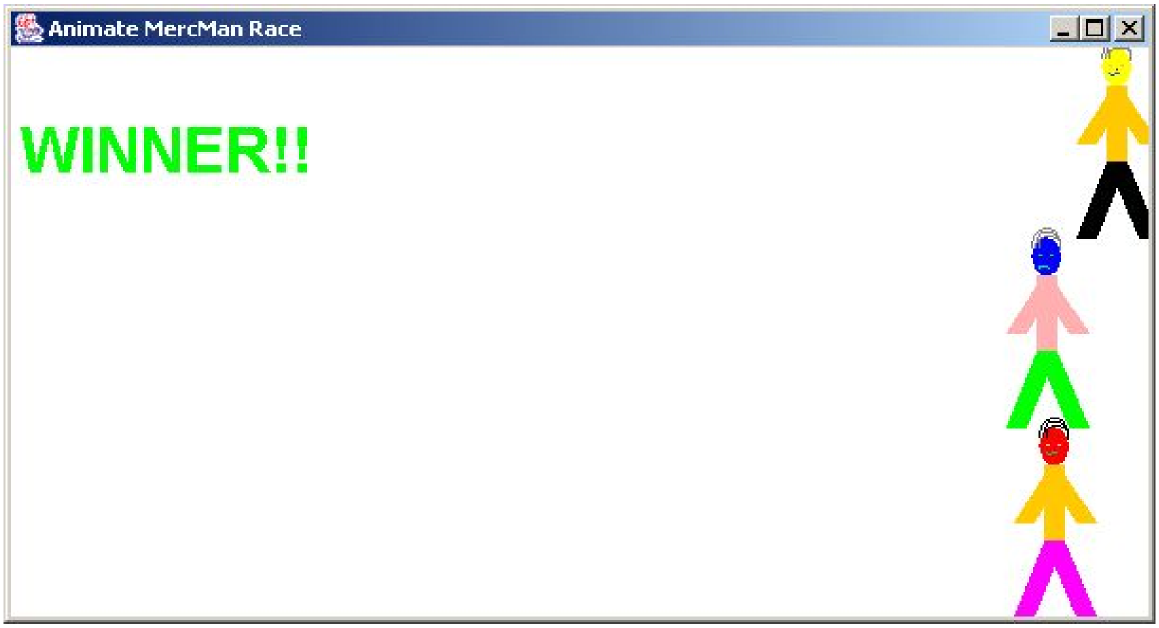
Now it is time for the races to begin! Your program should flow into a loop that randomly selects one MercMan at a time from the ArrayList to advance one pixel closer towards the finish line (right edge of the screen). This loop should continue until a MercMan touches the right edge of the screen.

To do this animation, after you randomly select which MercMan to advance, your program should then use the getters and setters to find out where the MercMan is currently on the screen and 'move' him/her one pixel to the right. Have your JFrame call its "repaint()" method. This should give the illusion of the

MercMen moving across the screen towards the finish line. During the "race" your screen might look like:



This process should be repeated until one MercMan is all the way to the right side of the screen. Whenever a MercMan reaches the "finish line" (far right side of the screen) you should break out of your loop and insert the word "WINNER!!" at the starting position of the winner's lane. Change the losing MercMen to have a frown. At this time your program should stop. Your final screen might look something like:



<u>RunnerComponent</u>

This should be a fairly short and sweet class to write. Recall that most of our other 'component' classes extend JComponent and have not had a constructor (we were relying on JComponent to construct). This class will need a constructor because 'main' is going to be passing us the ArrayList of MercMen. So, write a constructor that calls JComponent's constructor (i.e. "super();") and then copy the passed in ArrayList into a private instance variable. Finally, this class should have a method named paintComponent(Graphics g) that merely loops through all of the MercMen in the ArrayList and calls their draw() methods. Look at some previous assignments to assist you with this class.

<u>Bonus 10 points:</u>

When a winner is determined, make him do jumping jacks or some other celebration (forever). That is, you would have to create a new method in MercMan that displays him in different positions. The main routine would then need to alternate displaying MercMan in the two or more different positions. These changes will happen very fast unless you place some Java code in there to slow him down. The following code makes Java pause, or sleep for one half a second. You can adjust the integer to make Java pause for longer or shorter times:

```
try {
  Thread.sleep(500);
  }
catch (InterruptedException e){}
```

<u>Bonus 10 points:</u>

Make all the losers do push-ups once a winner is established.

<u>Deliverables:</u>

When finished, you should copy your Eclipse src folder to your shared Google folder, and rename it MercerRoadRace. Have fun, be creative, and feel free to add enhancements. Let the races begin!!! ☺