Clock Animation Part 2 (due **Monday, November 2, 2020** at 11:59pm)

In Part 2 of the project, we add the capability of setting or re-setting the time.

Recall that in Part 1, the ClockShape object called “shape” was constructed so that the initial value of the variable “count” was 0. Together with the initializations of theta, alpha, and gamma as -pi/2, these initial values caused the clock to start at 12 : 00 : 00. Then, each time the timer ticked off an additional second, the timer event handler called the “translate” method to increment “count” by 1 second and rotate the seconds hand through 1 second. Further, if “count” was a multiple of 60, we rotated the minutes hand through 1 minute, and if “count” was a multiple of 720 (12 minutes worth of seconds), we rotated the hours hand through 1/5 of an hour.

To permit the user to set the initial time or to later re-set the time, in the main program we clearly need to add 2 textfields and a button to a JPanel (which must then be added to the JFrame). One textfield can be initialized with “enter starting hour”, the other with “enter starting minute”. The button can be labelled “set time” or “set/re-set”, or something similar. When the window first becomes visible, the clock shows 12: 00 : 00 and the clock is not moving. The user should enter the starting hours and minutes and then click the button. How should the button click event handler respond?

To set or re-set the time, the button click handler can begin by temporarily stopping the timer. It can then convert the hours that the user entered from String to int (we have done this kind of thing before). Similarly for the minutes. Recall that the ClockShape object “shape” has methods setH, setM, setC. We can use these to set the value of h in “shape” to the hours, the value of m in “shape” to the minutes, and we can set the value of “count” to 0. We should also restore the original text to the textfields, so that later, someone can re-set the time. Finally, re-start the timer.

The timer handler is the same as before: it calls the “translate” method in ClockShape to re-position the hands and then it calls label.repaint() to repaint the image. So it is the “translate” method that will re-set the time. Most of the code for “translate” can be the same as before: increment count and theta, then check to see if the munites or hours hand have to be updated. But we can begin “translate” with new code which asks whether “count” = 0, which will be true as a result of a button click event. We can use this test to set or re-set the time. We will always start the seconds as 0, so theta can always be initialized with -pi/2. (If you want to, you can include a third textfield for initializing the seconds, in which case theta would be initialized with the input value s.) Then initialize alpha (for minutes) as -pi/2 + m \* pi/30. Initialize gamma for hours by using -pi/2 + h\* pi/6, but further rotate by the number of fifths of an hour (12-minute periods) corresponding to m. Finally, re-set “count” to equal the total number of seconds corresponding to this new time (include s if you are re-setting seconds). All of this happens only when “count” = 0, as a result of a button click even followed by the next timer tick.

You can also experiment with other design features. Perhaps you want the seconds or hours marks around the outside to have flickering random color, or some other color scheme, etc. You can add in other features as you wish.

\*\*The user should be able to set the clock to a specified hours and minutes via textfields and a button. Be sure to properly position the seconds/minutes marks (using the upper left corner of the surrounding square as the anchor point in the Ellipse statement.)