Rahil Thanawala, Aaron Lack

Version1

Public class ClockSystem{

Arraylist variable to store alarms

Minutes, seconds, hours variables

Method addAlarm – takes arguments hours, mins, secs

Sets the alarm by adding it to the arraylist

Returns your arraylist and verifies addition

Method deleteAlarm – takes same args as addAlarm

Removes the alarm from arraylist

Returns your arraylist and verifies subtraction

Method activateAlarm

If time is equal to any element of the alarms arraylist

Returns a noise

Method startStopwatch – void

Begins a time counter

Returns nothing

Method getstopwatch – string return

Returns the time elapsed

Method stopStopwatch – no args

Ends stopwatch and returns time elapsed

Method clock – void

Returns time

}

Main calls

Create clock system object

Print the following

addAlarm 6:41:20 – returns [6:41:20]

deleteAlarm 6:41:20 – returns []

call activate alarm method if needed

startStopwatch – returns nothing

getstopwatch – returns the time elapsed

stopstopwatch – returns time elapsed

clock() – returns time

Version 2

public class alarmClock{

vars: arraylist of alarm clocks, hours, mins, secs

public alarmClock – takes arguments hours, mins, secs

Sets the alarm by adding it to the arraylist

Creates alarmclock object

Tostring method will return the list of alarm clocks arraylist as a string

Method deleteAlarm – takes same args as addAlarm

Removes the alarm from arraylist

Returns your arraylist and verifies subtraction

Method activateAlarm

If time is equal to any element of the alarms arraylist

Returns a noise

}

public class stopwatch{

vars: Boolean active, Time object(for comparing time elapsed)

Method startStopwatch – void

Begins a time counter

Returns nothing

Method getstopwatch – string return

Returns the time elapsed

Method stopStopwatch – no args

Ends stopwatch and returns time elapsed

}

Public class timer{

Vars: startTime to count down from , time object

Public timer input time

Initialize timer object

Startcountdown method

Begin decrementing time

Pause and resume countdown method

Pause or resume iteration time

Countdown method

Decrement time

Make a noise if time at 0

}

Public class time{

Vars: hours mins secs

Time constructor public time(input looks like normal time 12:15:00)

Assign time to variables

Use substring to convert to hours mins secs

Public int gethours

Return hours as int

Public int getmins

Return mins as int

Public int getSeconds

Return seconds as int

}

Both versions will do the job of giving the user full accessibility to a comprehensive clock system. This includes a timer, stopwatch, alarm, and clock. The first version would put everything in one class. This has the benefit of being accessible in one file, but it results in difficulty with debugging. If there is an error, it becomes difficult to isolate it because everything is in one spot. However, if we use the second version, this problem goes away. You no longer have to worry about the code being jumbled and overlapping with each other. This way we can minimize dependencies and side effects. I also find it easier to plan project design when there are multiple classes. It helps you split up your goals and assign what you need. This can also benefit group workflow.