Consider a program that measures or deals with elapsed intervals of time, such as a stopwatch, a scheduler, a TV recorder, etc. A useful abstraction is such a program would be an object representing an elapsed span of time. (Relax! You’re not writing that application!)

You are responsible for developing a new class called TimeSpan, where each TimeSpan object represents an interval of elapsed hours and minutes. For example, we could construct a TimeSpan representing an interval of 6 hours and 15 minutes. We’ll only represent hours and minutes with this class, ignoring larger or smaller units such as days or seconds.

The straightforward (aside: though not necessarily the most efficient or “best”) way to represent the interval is simply with two integer instance variables called **hours** and **minutes**. **Note:** the class invariant which must be maintained is that **minutes** must be less than 60...this does NOT mean, however, that the user or application cannot enter a number >= 60 for **minutes**, but that the class methods must convert any minutes 60 or over into an appropriate number of hours.

The class has two constructors, one with no parameters that creates a TimeSpan with a default setting of 0 hours, 0 minutes, the second one taking parameters for hours and minutes and using those values to initialize the time.

Additional operations for the class include:

- **add( )**, which accepts parameters for hours and minutes and adds them to the current setting of the TimeSpan.

- **set( )**, which accepts parameters for hours and minutes and uses them to set the current setting of the TimeSpan.

- **reset( )**, which takes no parameters and resets the current setting of the TimeSpan to 0 hours, 0 minutes.

- **equals( )**, which compares two TimeSpans for equality.

- **toString( )**, which returns a string representation of the current setting of the TimeSpan.
How to Proceed

An outline of the TimeSpan class can be downloaded from our grader account /home/grader/lifick162/assignfiles/timespan. You will also find a driver program called TimeTest.java that will help you test the code you add to the TimeSpan class. You do NOT need to change any code in the TimeTest.java program or write any additional application for this assignment.

Your task by the end of this period is to complete the code required for the TimeSpan class.

Suggestions

You obviously need to develop the constructors first. Then develop the toString( ) method, so you can print out a TimeSpan object. Test just this much code with the driver program.

Follow this with the set( ) and reset( ) methods. Test this much code.

Implement the add( ) method. Test this much code.

Implement the equals( ) method…although there is not a test in the test driver for this method, it should be very simple to implement equals( ). You can write your own code to test this method.

Submit

Submit this assignment as the timespan lab.

Sample Output of Test Driver (underlined values are inputs)

Menu choices are:
q: quit the testing program
?: print this menu
+: add time to the current time span
p: print the current time span
r: reset the time span
s: set the time span

Enter command: ?

Menu choices are:
q: quit the testing program
?: print this menu
+: add time to the current time span
p: print the current time span
r: reset the time span
s: set the time span

Enter command: +
Add hours: 3
Add minutes: 65
Enter command: p
Current Time Span: 4h5m

Enter command: r
Enter command: p
Current Time Span: 0h0m

Enter command: s
New hours: 5
New minutes: 75
Enter command: p
Current Time Span: 6h15m

Enter command: q
Thank you for using TimeSpan.