Homework 4 Practice Using JUnit
CSCI 475 Senior Design
Due Wednesday September 28, 2004

Learning JUnit

Set up your account

Under “Resources on JUnit - unit testing in Java” link on CSCI 475 web page

`http://www.eg.bucknell.edu/~cs475/F04-S05/JUnit.html`

read “Sample program to demo JUnit”. Set up your account by adding the three lines mentioned to your .cshrc file.

Learn about JUnit

Read and study **carefully** the article at “Test Infected: Programmers Love Writing Tests” link. Also, read pages 93-106 in our XP text “Extreme Programming Installed.”

Exercise to practice JUnit

Select a partner to do pair programming for this exercise. Design a Java application that implements a class for latitudes on Earth. You must use XP Unit Testing process with JUnit.

A latitude is an angular distance north or south from the Earth’s equator measured from 0 degrees to 90 degrees. Each degree is divided into 60 minutes and each minute is divided into 60 seconds. For example, the latitude for Lewisburg, Pennsylvania is $40°57'57''N$ which means Lewisburg is 40 degrees, 57 minutes and 57 seconds of arc north of the equator.

Your Latitude class must implement the following interface.

```
public class Latitude{
  public Latitude() // sets deg, min and sec to 0.0 N
  public Latitude(double degrees, double minutes, double seconds,
                  String hemi) // where hemi is either "N" or "S".
  public boolean equals(Latitude lat)  // a.equals(b) is true if a = b
  public boolean greater(Latitude lat) // a.greater(b) is true if a > b
  public double getDeg()
  public double getMin()
  public double getSec()
  public String getHemi()
  public Latitude add(Latitude lat)
  public Latitude sub(Latitude lat) // a.sub(b) is a - b
  public String toString()
  public void input() // a.input() sets deg, min, sec and hemi of a
}
```

That is, your class should be able to store, retrieve, input, output, compare, add and subtract latitudes. Notice that $0°S$ equals $0°N$. Allow user to use negative values on input or in a constructor, e.g., input of 3D-20'N means $2°40'N$. If at any time a latitude value exceeds $90°N$ or $90°S$ your program should display an appropriate error message and set the object to $90°N$ or $90°S$ respectively. Keep the input method simple.

Remember in Unit Testing, you design the test cases **BEFORE** you implement the method. Both partners should be involved in designing the test cases. As you implement the method for the class, you add the code for the tests. Then you run JUnit and change code until no test fails. Then on to the next method. Each time you run the new tests and **all** the previous tests.

Please email me your working code for the Latitude class and the code for your unit tests. Provide me with proof that all your unit tests pass. As part of your grade, I will run your Latitude class on *my* unit tests.