*Project 2: The Storm Chaser*

**Goals**

This is the instructions and helper file for Project #2 (The Storm Chaser). This project incorporates concepts that we already understand (i.e. methods, loops, input, output) and introduces some new ones (i.e. a deeper look at classes).

**Step-by-Step Development**

As you know, programs are written in stages, not all at once. Start with a small piece of the program, and work up to the final version in small steps. Here is a *suggested* strategy for this project. At each step, make sure you have a working program before you go to the next step!

1. **Get Familiar with the Problem**

Carefully read the program description and look at the data file to gain an understanding of what is to be done. Make sure you are clear on what is to be calculated and how. That is, study the file and program description and ponder!

Think!

1. **The Storm Class Type**

Now concentrate on the Storm class that we define to hold the summary information for one storm.

* First, look at the definition of Storm in the Storm.java file.
* You must complete the methods in the Storm class and use them effectively throughout your program.
* Note: The SaffirSimpson() method uses ***wind*** and ***pressure*** to calculate the ***category*** of a Storm.
* Whenever you update ***wind***, you must convert it from knots (how it is stored in the data file) to miles per hour.
* The SaffirSimpson() and toString() methods have been written for you and do not need any further coding. The toString() method might need some tweaking for alignment purposes.

1. **The GetStorm() Function**

This is the most challenging method in the program! Here are some points to consider as you try to design this function.

* Take another look at the data file - and notice that typically there are several records associated with one storm. It will be the job of GetStorm() to process these records, and build a Storm object, containing the relevant summary information.
* Work out the *while* loop needed to process the correct records. For example, process records until the sequence number changes, or ....
* Remember that this method returns a Storm object. It will have to use the ***new*** operator to allocate memory for this object.

1. **Add the sort function**

Use the selection sort algorithm to sort the Storms. Remember, we are sorting the array of Storms but the key is the Storm category.

1. **Final cleanup and testing**

Clean up any small bugs and add comments for documentation.

**What to Hand In**

Submit your code and screen output by the due date.

**Project Files**

The relevant files for this project are:

* StormChaser.java (program starter file)
* Storm.java (Storm class definition)
* hurricanedata1950to2015.txt (data file)

**Work Step-by-Step**

**Get a Working Program at Each Step**

**Have Fun!**