CS 6340 – Spring 2013 – Assignment 8 **Name**_____
Assigned: March 4, 2013
Due: March 13, 2013 **Name**_____

At the beginning of class on the due date, submit your neatly presented solution with this page stapled to the front (100 points).

Part 1

Your new position as Test Manager requires that you establish a set of requirements that developers will use for unit testing of the software that they write. Before you establish these requirements, you want to assess the fault-detection ability, expense, tool availability, etc. of various techniques that have been proposed in the literature. To do this, you will use a program, which we'll call **tritype**, that has the following requirements specification

> **tritype** takes as input three integer values. The three values are interpreted as representing the lengths of the sides of a triangle. The program prints a message that states whether the triangle is scalene, isosceles, or equilateral.

You are to do the following:

1. Use the specification to develop a set of test cases (a test suite) for **tritype** using two black box testing methods (both described in "EquivalencePartitioningBoundaryValue:"
   - Equivalence Partitioning
   - Boundary Value Analysis

2. Create a file of test cases (reason for test (in quotes), inputs, expected outputs) that consists of one test case per line; the number of the test case will be the line number in the file. For example, suppose I created two test cases:
   Test Case 1: isosceles 2 2 3 isosceles
   Test Case 2: equilateral 4 4 4 equilateral
   The file should contain
   "isosceles" 2 2 3 isosceles
   "equilateral" 4 4 4 equilateral

3. Send the test cases to Sangmin, and he'll send you the **tritype** program for the second part of the assignment. Let him know whether you want the C version or the Java version.

Part 2

Using your copy of the implementation of **tritype** (either in C or in Java), perform the following activities:

1. Measure the *Statement Adequacy* of the test suite you developed in Part 1.
2. Measure the *Multiple Condition Adequacy* of the test suite you developed in Part 1.
3. Measure the *MCDC Adequacy* of the test suite you developed in Part 1. An explanation of MCDC Adequacy can be found in "Test-suite Reduction and Prioritization for Modified Condition/Decision Coverage"
4. Create the control-flow graph for **tritype**
5. Using the control flow graph for **tritype**, compute the cyclomatic complexity of **tritype**.
6. Measure *Basis Path Adequacy* for the test suite you developed in Part 1.

For any of 1,2,3,5, or 6, you can (a) perform the tasks manually or (b) automate the process by writing a program to perform the task or by finding an existing tool that will do it for you.

At the beginning of class on the due date, submit the following:

- Your test suite (typed).
- Statement Adequacy: a list of the statements and, for each statement, an indication of whether the statement was covered; a statement of the coverage achieved by your test suite.
- Multiple Condition Adequacy: a list of the conditions required for multiple condition coverage and, for each condition, an indication of whether the condition was covered; a statement of the coverage achieved by your test suite.
- MCDC Adequacy: a list of the conditions required for MCDC coverage and, for each condition, an indication of whether the condition was covered; a statement of the coverage achieved by your test suite.
- The control-flow graph for **tritype.c**.
- Basis Path Adequacy: The cyclomatic complexity, along with an explanation of how you got it; a statement of the coverage achieved by your test suite.
- A discussion of the results of your experiment (e.g., difficulty of measuring adequacy, expense of measuring adequacy, inadequacy of your initial test suite, ability of the various techniques to help uncover errors).
- A decision about what requirement(s) you will establish for your developers, and why you made this decision. To help with your decision and discussion, you can select a type of software that your company develops.