CSE 4283/6283 - Software Testing and Quality Assurance - Spring 2016

Assignment 4 --- Test-Driven Development & Unit Testing

Objective

Use test-driven development to implement a set of software requirements. Write unit tests to provide full coverage of developed code base using multi-faceted independent tests. Create a control flow graph. Groups up to 6.

Scenario

Your team has been tasked to write code on an immensely parallel high-performance-computing architecture to compute complex functions requiring high computational effort. These functions will be executed by researchers from various disciplines and will provide answers to difficult questions. The application will be accessed from the command line and provide users with a list of functions that they can execute. You are to code the functions below adopting a test-driven development approach. We define a unit as a function or method in a class with a single responsibility. The unit tests written should enable 100% coverage of the application. To verify tests, your team will create a control flow graph for one functional requirement. All code and tests should be added and maintained in a GitHub repository.

Requirements:

1. **Body Mass Index** - Input height in feet and inches. Input weight in pounds. Return BMI value and category: Underweight = <18.5; Normal weight = 18.5–24.9; Overweight = 25–29.9; Obesity = BMI of 30 or greater (need to convert height and weight to metric values - see formula linked below)

2. **Retirement** - Input user's current age, annual salary, percentage saved (savings matched by employer - so double amount saved). Input desired retirement savings goal. Output what age savings goal will be met. Output a message if the goal is not met (assume death at 100 years).

3. **Distance Formula** - Input 2 points (x1, y1) (x2, y2) [4 values] and calculate the distance between the points using the distance formula (should be implemented without use of libraries with the exception of a square root library function).
   Distance Formula: $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

4. **Email verifier** - Input a string, determine if it is a valid email address (i.e., some_string1 '@' some_string2 '.' some_string3 that's 3 letters or less) - can assume some_string1 and some_string2 are made up of any alphanumeric character or symbol - some_string3 must be less than 3 letters.

Assignment

Write a report summarizing your efforts. Your report should consist of the following content:

1. **Lessons learned (up to 2 paragraphs) -** Benefits / drawbacks of TDD and unit testing; Would you choose to apply TDD for a project?
2. **TDD Screenshots** - examples of failed then passed tests as evidence of TDD process conformance
3. **GitHub Link to access commit history** - each team member *must* develop and commit at least 1 test to the repository
4. **Setup and use instructions -** include links to download and setup test framework
5. **Testing framework report** or output showing 100% code coverage

6. **Choose one requirement to create a control flow graph**. Label each node and provide a description of processing for each node (i.e., which line of code is the node responsible for)
7. Submit each group member's name and NetID with GitHub account username

Notes & Resources:

- Choose your own programming language and unit testing framework
- BMI Formula - http://extoxnet.orst.edu/faqs/dietcancer/web2/twohowto.html
- Distance Formula - http://www.purplemath.com/modules/distform.htm
- Each function should be tested using 2 or more unit tests
- Provide a brief description for each unit test

Submit your assignment (*.pdf) as an attachment via MyCourses - Due Thursday, May 5 by ****5:00pm**** (late submissions penalized - 20% - no exceptions)

Grading

TDD process conformance - Quality of unit test - Control flow graph - Technical quality of insights and report, grammar/style, professional presentation.

100%