CPSC 233 – Individual Assignment 7 – Requirements

This assignment is different from previous assignments. In previous assignments, unit tests were provided and you were to create/update a class that would pass the unit tests. In this case, a class is provided and you are required to create unit tests to verify that the implementation is correct.

The class provided is called CreditHistory and contains the following public methods:

- *addRating* which does not return anything and takes a rating (type int) as an argument. It adds the ratings to the list of ratings in the Credit History. Ratings must be between -5 (inclusive) and 5 (inclusive) to be included in the list.
- *getRatings* which returns the list of all ratings stored in the Credit History, in the order that they were added in an ArrayList<Integer) (And ArrayList that contains Integer objects.) Note that you will have to compare the elements in the list with instances of the Integer object. To convert from the primitive int to the object Integer you can use an Integer constructor: for example *new Integer(7)* will create an Integer object that is equivalent to the int 7.
- *trimRatings* will reduce the number of ratings stored to 10. The 10 most recently added ratings will be kept. Other ratings are discarded.
- *numOfRatings* returns the number of ratings currently stored for the Credit History.

Note that the class may also contain instance variables and private methods. But these do not need to be tested and should not be considered when designing tests: If the provided implementation were changed but still met the requirements, your tests should still pass. Your responsibility is to verify that methods perform as required. If you are unsure about the exact requirements of a method, post your questions on the discussion board and further clarifications will be provided there.

To test your tests, four possible solutions of CreditHistory are provided. One of the solutions is correct and the remaining three solutions each contain two or three logic errors. Each solution is in its own folder. The folder Error1 contains errors in the methods addRating and getRatings. The folder Error2 contains errors in the methods addRating and trimRatings. The folder Error3 contains errors in the methods getRatings, trimRatings and numOfRatings.

Submit your unit tests to WebCAT. You may name your test class(es) anything your wish but it should have the word Test in the name. In your submission also include one of the CredityHistory.java versions. Your grade will be based on how thoroughly you test CreditHistory.

A grade of B+ (3.3) in WebCAT is required to qualify for the full team grade. Since the requirements in this assignment are different than previous assignments, if this proves overly challenging, this may be adjusted. We will monitor WebCAT submissions to ensure a B+ is well within reach for this assignment.