Assignment 2

Classes and Object

Goals:

- Practice designing, implementing, and documenting a class
- Test the class designed

Reading:

Read chapters 2, chapter 4, and section 8.4 as well as Appendix F

Read and use the *Good Programming Practices* in BrightSpace.

Program:

Implement a class called Student.

- A student has a name
- A student has an address
- A student has a grade point average GPA
- A student has a unique student number that is given 'by the system' when the student is constructed
- A student has a login id that can be obtained (you do **not** need to check for uniqueness of the login id – just follow the rules below on assigning the login id)

Supply appropriate constructors and public methods for the Student class. Provide the following methods:

- getName
- getAddress
- addCourse
- calculateGPA
- getStudentNum
- getLoginId

Implement a test program for the class Student.

As stated in the text book, "The purpose of a test program is to verify that one or more methods have been implemented correctly. A test program calls methods and checks that they return the expected results." So, instead of writing a separate test plan as you did for the first assignment, incorporate the test plan into the main method of your "StudentTester" class. This main method can (and should) call other (static) methods to test the various methods of the Student class. As recommended in the textbook, think through the expected results. Consult also section §3.4 in the textbook on "Unit Testing".

About Your Implementation::

- Comment your classes and methods appropriately using the Javadoc notation.

- Every time that a course is added, the grade point value earned (i.e. the grade points) for that course must be passed as an argument as well as the number of credits units for the course. You don't need to keep track of the courses taken.

- "The Grade Point Average GPA is calculated by taking the grade points a student earned in a given period of time divided by the total number of credits taken." http://www.back2college.com/gpa.htm

- The Student constructor has two String arguments. One string will be the student address (which may contain blanks and is free formatted). The other argument is the student name. Assume that every student name consists of exactly two words: the first word is the "first name" and the second word is the "last name" (we are not allowing "middle names"). There can be blanks in the argument of the student name: remove the extra blanks before and after the first and last name.

- The "student number" assigned to the first student created should be 10000001. Every time that a student is constructed, the (system) assigned "student number" is one greater than the last constructed "student number".

- "Compute" the login-id as follows (all the letters of the login-id are in lower case):
    - Make the first letter of the 'login id' the first letter of the "first name" (and remember that the "first name" is the first word of the student name, normalized to remove extra blanks).
    - Make the middle part of the 'login id' the first 3 letters of the "last name"
        - if the "last name" has fewer than 3 letters, then use all the letters of the "last name"
    - the last part of the 'login id' consists of the last 2 digits of the 'student number' i.e. the two least significant digits

    - Examples
        - "April Schauer" with a 'student number' 10001 has a 'login id' `asch01`
        - "Norma Li" with a 'student number' 105345 has a 'login id' `nli45`
        - " Brock O " with a 'student number' 10005 has a 'login id' `bo05`
        - "Misty Waters" with a 'student number' 10010 has a 'login id' `mwat10`

Submit:

Submit to BrightSpace a single zip (compressed) file:

1. The source code
    - the file Student.java
    - the file StudentTester.java.

Do not submit any byte code nor the HTML files produced by the Javadoc program.

Marking Scheme:

- [55] Student class design and implementation
    - [5] private instance variables
    - [3] Static use of nextStudentNum
    - [10] Student constructor
    - [4] public getName()
    - [4] public getAddress()
    - [4] public getStudentsNum()
    - [7] public addCourse (credits, gradePoint)
    - [6] public calculateGPA
    - [12] public getLoginId()
- [15] Student Tester program
- [15] Comments
- [15] Programming style