Lab Exercise Three - CIS*2430 (Fall 2021)

Due Dates:

Monday Labs	*- November 1st by 11:59pm*
Tuesday Labs	*- November 2nd by 11:59pm*
Wednesday Labs	*- October 27th by 11:59pm*
Thursday Labs	*- October 28th by 11:59pm*
Friday Labs	*- October 29th by 11:59pm*

Extend the class list system you created in Lab Exercise 2 with a subclass and also allow file I/O for user interactions. The original program and its functionality should remain, but several new menu options are to be added:

- The **Student** class will now have a subclass **GraduateStudent**, which inherits all attributes and methods from the original Student class.
- However, all GraduateStudents should have three new variables: supervisor, isPhD (boolean), and undergraduateSchool, where supervisor and isPhD are mandatory.
- The toString method for Student should show the attributes: program, year, and average grade on separate lines.
- The toString method for GraduateStudent should override that for Student, which will not only print program, year, and average grade, but also add additional lines for *PhD/Masters*, *supervisor*, and *undergraduateSchool*, where the decision to print "*PhD"* or *"Masters"* is determined by the value of *isPhD*.
- The menu should now look like the following:

  (1) Enter information for a new Student.
  (2) Enter information for a new GraduateStudent
  (3) Show all student information with each attribute on a separate line
  (4) Print the average of the average grades for all students as well as the total number of students
  (5) Enter a specific program and show all student information for that program
  (6) Load student information from an input file.
  (7) Save all student information to an output file
  (8) End program.

Note (1): In the class list system, all students and graduate students should be stored in a single ArrayList, and Option (3) essentially loops through the list by calling the toString method for each object (either a student or a graduate student).

Note (2): A filename needs to be provided for Options (6) and (7). Loading an input file will create the related student objects and add them to the existing ArrayList, while saving an output file will dump the exisiting ArrayList to the related file. If an input file is empty, a warning message should be given to the user and writing output to an existing file will wipe out its current content.

Note (3): For graduate students, year is how many years they have been working on their graduate degrees.

In the example below, words in italics represents user actions, while bolded words represent program output. In this example, the average grade of the given student is set to the default of 0%.

<u>**Example Usage and output:**</u>

Program displays menu.
User selects option (6).

Please enter the name of the input file:
students.txt

****Reads and parses file****

Program displays menu.
--

The input file should be a text file with the following format:
Program Year avgGrade supervisor isPhD undergraduateSchool
where program, supervisor, and undergraduateSchool are all made of single words and isPhD is either 1 or 0

Examples:
CompSci 4 85.4 Song 1 Guelph
Psych 2 92.1
Biology 1 75.0 Cornell 0 McGill

You **can** assume that the input file is perfectly formatted since it is likely saved previously by your program. However, you **cannot** assume that the input file will exist, which must be error checked.

This lab introduces lots of new material. It may take slightly longer than your previous lab exercises, but the experience you build will be **vital** to help you complete Assignment Two.
If you have ANY questions, never hesitate to ask or email the teaching team. We are here for you!

MARKING RUBRIC (10 marks in total)

2 Marks for the command loop, and options (1), (4) and (5).
1 Mark for the extension to option (3)

3 Marks for option (2) as well as proper implementation of inheritance and structure of GraduateStudent class
2 Marks for option (6) (File Input)
2 Marks for option (7) (File Output)