CSCI 24000 – Fall 2017
Assignment #4 – Stacks Upon Stacks

Due: 11/6/2017

This fourth assignment will allow you to explore four different concepts in one program. You will be creating a program that makes use of: an abstract data type (ADT), file input/output (I/O), classes & objects, and the creation of a Unified Modeling Language (UML) Class diagram in an object-oriented (OO) programming language (Java). This will also serve as an extension of your Assignment #3 – we will reuse code where possible.

For this assignment, you are tasked with the responsibility of reading in a list of students from a text file, creating a Student object for each student, pushing the Student object onto a Stack (ADT), and then providing an option to print the contents of the Stack by *popping* the students off the Stack one-by-one (You should print each Student on a separate line in the console – see Sample Output). This program will be written in **Java** and must compile and run on **Tesla** (tesla.cs.iupui.edu).

Your program will be menu driven in which you will provide the following prompts to the user:

```
1. Load Students (From File)
2. Print Stack
3. Exit Program
```

We will assume the each student has the following data available:

- `First name`
- `Last name`
- `Address`
    - `Address (Line 1)`
    - `Address (Line 2)`
    - `City`
    - `State`
    - `Zip Code`
- `Student ID`
- `GPA`

The text file containing sample student data will be provided to you. The filename will be **students.txt** – this file can be found (and downloaded) on Canvas. The file will contain **ten** (10) students.

- The format of the file will be as follows:

```
FirstName,LastName,StreetAddress,Address2,City,State,ZipCode,ID,GPA<end of line>
```

- Example Student:

```
Test,Testerson,123 Lane,Apt. 2,Indianapolis,Indiana,46220,123456789,3.75
```

Development Process:

All development must take place on the **master branch** in a **private** GitHub repository. You must add Me (rrybarcz) and all four (4) TA's as collaborators. It is required that you commit and push often! You also need to provide useful comments on your commits. We will be checking to make sure that you are actively pushing changes to your repository – failure to do so will result in a loss of points. You are also required to include the Honor Pledge and Digital Signature in each one of your source files (any file that you have written "code" in – you need to include the Honor Pledge and Digital Signature).

You are required to implement the following items as part of this assignment:

- Your program must contain at least **four** (4) classes.
  - Driver
  - Stack
  - Student
  - Address
- Your Stack (ADT) must be generic, thus allowing it to handle any type.
  - You are **NOT** allowed to use the built-in Java Stack.
- You are required to handle all exceptions (I/O and Stack).
- You should not print empty "lines" – for example not every student will have an Address 2 (e.g., an apartment number), you should ignore these lines when printing out the Student.
- You are required to submit an algorithm – your algorithm should outline the step-by-step process you have taken to solve this problem.
- You are also required to provide a UML Class Diagram with this project – you can achieve this by using a commercial tool (e.g., Microsoft Visio), but you are not required to use a commercial tool to complete this component.
  - This diagram should outline the classes (name, attributes, and methods) and relationships between the various classes. This diagram should match the program (code) that you submit.

Below is example output of what your program should display when executed:

```
1. Load Students (From File)
2. Print Stack
3. Exit Program
Enter Your Selection: 1

Students Loaded From File!

1. Load Students (From File)
2. Print Stack
3. Exit Program
Enter Your Selection: 2

ID: 12          Name: Test Testerson      Address: 1740 Parkview Drive Sugar Land,TX 77478   GPA: 3.127
ID: 567567      Name: Test Testerson      Address: 3479 Austin Avenue Brunswick,GA 31520     GPA: 3.256
ID: 685544      Name: Test Testerson      Address: 1225 Gnatty Creek Road Westbury,NY 11590  GPA: 2.985
ID: 709095      Name: Test Testerson      Address: 1125 Joyce Street  Mobile,AL 36693        GPA: 3.995
ID: 576568      Name: Test Testerson      Address: 2656 Creekside Lane Goleta,CA 93117       GPA: 1.598
ID: 564563      Name: Test Testerson      Address: 4489 Tully Street Apt B Detroit,MI 48219  GPA: 2.127
ID: 124789      Name: Test Testerson      Address: 845 Dola Mine Road  Wendell,NC 27591      GPA: 3.958
ID: 333555      Name: Test Testerson      Address: 2707 Waterview Lane Apt 302 Las Vegas,NM 87701    GPA: 2.546
ID: 456789      Name: Test Testerson      Address: 3668 Thunder Road Palo Alto,CA 94306      GPA: 2.978
ID: 123456      Name: Test Testerson      Address: 1843 Glenview Drive Apt. 2 Corpus Christi,TX 78401 GPA: 3.215

1. Load Students (From File)
2. Print Stack
3. Exit Program
Enter Your Selection: 3

Goodbye!
```

You should **always** write your algorithm before you begin actual coding. This is a good practice and demonstrates the importance of software design. If you ask for help we will always ask to see your algorithm before looking at your code. You will also be creating a UML Class Diagram. This should be completed prior to the construction of your actual code – again we may ask to see your UML Class Diagram.

Submission:

All assignments must be submitted on IU GitHub (github.iu.edu) in your master branch by the due date. The name of your IU GitHub repository must be as follows: **csci24000_fall2017_A4**. You should be submitting the source files (e.g., ***.java**), a Makefile (e.g., **makefile**), a UML class diagram for your program, as well as a file (e.g. **algorithm.txt**) outlining your algorithm for accomplishing this assignment. Failure to follow these rules will result in a 0 on your assignment submission.

Blackbelt:

For this project the Blackbelt extensions are as follows:

- Write this program in C++ (+10) – your program must have no memory leaks!

OR

- Java: Allow the user to modify the data and re-save it to the textfile (+5).
- Java: Allow the program to read-in *any* number of Students and store them in the Stack (+5).