CS 341 – Fall 2024
Assignment #1 – Keepin' It Classy

Due: 9/13/2024

This first assignment will serve as a refresher for the C++ programming language and allow you to become more comfortable with the tools we will be using this semester to accomplish our goals. This project will serve as the foundation for exploring the many benefits that the C++ programming language can provide and will help you re-familiarize yourself with Object-Oriented concepts that you previously learned in CS 248 (in Java) all the while learning `Make` and `Git/GitHub` in the process.

For this assignment we are going to build a GPA calculator and then sort the entries from lowest to highest, in terms of calculated GPA.

Development Process:

Phase I: (Due: 9/6/2024)

Your program will take a text file containing thirty (30) students (provided in a text file on Canvas named: `Files->Assignments->A1->students.txt`) and begin by reading in the contents of the file. You will use standard File I/O in C++ to accomplish this task (discussed in the lecture slides). The structure of the text file will be as follows `<STUDENT_ID> <GRADE_POINTS> <CREDIT_HOURS>`:

```
IY7G4G 437 123
69RIHC 461 121
```

The file will contain thirty (30) lines of data for you to read-in. Your job will be to create a `driver.cpp` and a `main` function that accomplishes this task. Once you have completed this you can test your program by printing out in the console the contents of the text file. Success will mean you have completed Phase I and are ready for Phase II...

Phase II: (Due: 9/8/2024)

When you are completing Phase I you are likely asking yourself…"*what are we going to do with all this data?*" That is where Phase II comes in – you will take the data collected in Phase I and build a unique instance of a `Student` for each line read-in. Here you will explore OO programming in C++. As part of this phase you will create two files: `Student.cpp` (definition(s)) and `Student.h` (declaration(s)). We will explore, in lecture, how to accomplish this task.

Your `Student` class will contain the following three (3) attributes: `id`, `gradePoints`, `creditHours`. The `id` attribute will be of type string and hold the id associated with the individual Student `<STUDENT_ID>`. The `gradePoints` attribute will be of type integer and hold the grade points value associated with the individual Student `<GRADE_POINTS>`. The `creditHours` attribute will be of type integer and hold the control value associated with the individual Student `<CREDIT_HOURS>`. In addition to these three (3) attributes your class **MUST** also contain appropriate accessor methods along with a default constructor and destructor. You may either use an additional method to construct a `Student` object or create a second constructor to accomplish this.

Once you have constructed all thirty (30) Students you can test your program by creating a `printInfo()` method that prints out each `Student` objects' information. In order to successfully compile your `Student` class and your `driver.cpp` you will need to create a `Makefile` (discussed in the lecture slides). This `Makefile` should generate an executable file named `GPA.exe`. You will then run `GPA.exe` at the command prompt. Success will mean you have completed Phase II and are ready for Phase III. Sample output of Phase II is shown below:

```
****    Welcome to Dr. R's GPA Calculator Program!      ****

NKCQKD: 200     120
NUII9Z: 351     120
IY7G4G: 437     123
69RIHC: 461     121
PN3IZ3: 319     120
08BL01: 441     128
7EAV6J: 382     120
XA6ZLR: 280     120
RQUJEJ: 369     120
4DDOQS: 469     135
N4E9YM: 423     125
N2LPY5: 240     120
KTKBU9: 386     120
8SFRHK: 331     120
2KSO0Y: 446     128
ETA52A: 269     120
MWB39P: 467     120
M68U8C: 356     128
HAW232: 439     120
5LQVGP: 243     120
OZ24YF: 378     120
QLAXZD: 206     120
X0A7V5: 473     120
MM5BK3: 471     142
H2JVDW: 455     120
3ZC0G5: 295     120
CGEMX8: 255     120
DEK1XI: 380     120
J5ANTL: 291     120
74437W: 364     120

Thank you for using my GPA calculator program - goodbye!
```

Phase III: (Due: 9/10/2024)

For the third phase you will be responsible for calculating the GPA and letter grade for each student and storing this information back on the user created Object. In order to accomplish this you will need to modify your `Student` class through the addition of two attributes: `gpa` and `letterGrade`. `gpa` will be of type `double` and calculate the GPA for the student based upon their `gradePoints / creditHours`. The `letterGrade` will be of type `char` and will only account for "full" letters meaning no + or -. You will need to add the necessary accessor methods for both attributes and don't forget to modify your Constructor(s) to account for these new members. An example of the output generated from this phase is shown below:

```
****    Welcome to Dr. R's GPA Calculator Program!      ****

NKCQKD: 1       D
NUII9Z: 2       C
IY7G4G: 3       B
69RIHC: 3       B
PN3IZ3: 2       C
08BL01: 3       B
7EAV6J: 3       B
XA6ZLR: 2       C
RQUJEJ: 3       B
4DDOQS: 3       B
N4E9YM: 3       B
N2LPY5: 2       C
KTKBU9: 3       B
8SFRHK: 2       C
2KSO0Y: 3       B
ETA52A: 2       C
MWB39P: 3       B
M68U8C: 2       C
HAW232: 3       B
5LQVGP: 2       C
OZ24YF: 3       B
QLAXZD: 1       D
X0A7V5: 3       B
MM5BK3: 3       B
H2JVDW: 3       B
3ZC0G5: 2       C
CGEMX8: 2       C
DEK1XI: 3       B
J5ANTL: 2       C
74437W: 3       B

Thank you for using my GPA calculator program - goodbye!
```

Phase IV: (Due: 9/13/2024)

For the fourth and final phase of this project you will be responsible for sorting the students based upon their GPA from ***highest to lowest***. This sort **MUST** be done in-place (meaning that the elements must be sorted in the data container itself not just printed out in the correct order). For this phase you will need to modify your `driver.cpp` to include a method for sorting. You may choose any sorting algorithm you desire to complete this implementation. You also will need to display the GPA properly (e.g., `X.XX` format) for this phase – the `iomanip` library can help accomplish this. An example of the output upon completion of this phase is shown below:

```
****    Welcome to Dr. R's GPA Calculator Program!      ****

                    X0A7V5: 3.94    A
                    MWB39P: 3.89    A
                    69RIHC: 3.81    A
                    H2JVDW: 3.79    A
                    HAW232: 3.66    B
                    IY7G4G: 3.55    B
                    2KSO0Y: 3.48    B
                    4DDOQS: 3.47    B
                    08BL01: 3.45    B
                    N4E9YM: 3.38    B
                    MM5BK3: 3.32    B
                    KTKBU9: 3.22    B
                    7EAV6J: 3.18    B
                    DEK1XI: 3.17    B
                    OZ24YF: 3.15    B
                    RQUJEJ: 3.08    B
                    74437W: 3.03    B
                    NUII9Z: 2.92    B
                    M68U8C: 2.78    B
                    8SFRHK: 2.76    B
                    PN3IZ3: 2.66    C
                    3ZC0G5: 2.46    C
                    J5ANTL: 2.42    C
                    XA6ZLR: 2.33    C
                    ETA52A: 2.24    C
                    CGEMX8: 2.12    C
                    5LQVGP: 2.02    C
                    N2LPY5: 2.00    C
                    QLAXZD: 1.72    C
                    NKCQKD: 1.67    D

   Thank you for using my GPA calculator program - goodbye!
```

If you reach this point you have successfully completed Assignment #1.

Submission:

All assignments must be submitted on Butler GitHub (github.butler.edu) – this will be discussed in the lecture slides if you are unfamiliar with this process. This an **individual** assignment – meaning that each student should submit their OWN work. The name of your Butler GitHub repository must be as follows: **cs341_fall2024**

For this assignment, all development must take place on the **master branch** in your GitHub repository. It is strongly recommended that you commit and push often! Please make sure to commit for each Software Development Phase – this will allow me to check on your progress and provide helpful tips! By applying this process in this assignment and practicing it will help to familiarize you with the workings of a source code repository and its importance in software design and development. We will be using this the entire semester – so best to become very well versed in it now!

The directory structure of the repository must contain the following files:

- **driver.cpp**
- **Student.cpp**
- **Student.h**
- **makefile**

Each source file (.cpp/.h) **must** include the Honor Pledge and digital signature – more details about this can be found in the lecture slides. Additionally, you should use good programming practices throughout your program including comments, variables naming, indenting, etc. A portion of your grade will depend on this!