**Assignment 5: Grades**

**This assignment is to be completed individually. It is not a team project.**

**You must document (using comments in your code) all resources (beyond our official textbook, Sakai, the class discussion forum, or the class website) that you used to help you complete this assignment.** For example, if you referenced a code example from an online website such as Stackoverflow or github you could acknowledge it as follows:

# Next we print 'Hello, World' to the console.

# Based on help from: https://stackoverflow.com/questions/826948/syntax-error-on-print-with-python-3

print("Hello, world!")

You must provide an attribution to any resource youÂ consulted to complete this assignment except for our official textbook, Sakai, class discussion forum, or the class website. Resources that require attribution include — but are not limited to — websites, books, notes from other students, tutors, or help from other people (friends, classmates, etc.). Under no circumstances are you to look at or copy any material related to another student's solution for this assignment.

To repeat, you must attribute **any resource** you consulted to complete this assignment other than our official textbook, Sakai, class discussion forum, or the class website. Failure to provide attribution is a violation of the honor code.

**Overview**

For this assignment, you'll be reading grade data from a file, determining student letter grades, and writing the results to a new file. The details for how your program should work are described in the Basic Requirements and Advanced Requirements sections below.

**Input Data.** A sample input data file of the type that your program should process is available here: [sample\_grade\_input.txt](https://ils.unc.edu/~gotz/courses/560/a/sample_grade_input.txt). This sample file contains fake grade data for 14 students. However, your program should be able to work with a file with an arbitrary number of students. For example, I may test your program with just 3 students, or with 300. Your program should work for any length file.

Each student record spans 3 lines in the data file. The first line contains the type of student: GRAD or UNDERGRAD. This is important because graduate and undergraduate students get graded differently at UNC. The second line contains the student's name. The third line contains the student's overall numerical grade. Your program will need to map these to letter grades according to the following rules:

For the sake of this assignment, GRAD student grades are defined as follows:

|  |  |
| --- | --- |
| **Number Range** | **Letter Grade** |
| 95-100 | H |
| 80-94 | P |
| 70-79 | L |
| 0-69 | F |

For the sake of this assignment, UNDERGRAD student grades are defined as follows:

|  |  |
| --- | --- |
| **Number Range** | **Letter Grade** |
| 90-100 | A |
| 80-89 | B |
| 70-79 | C |
| 60-69 | D |
| 0-59 | F |

Please note that grades for this course are assigned as outlined in the course syllabus. The grading scales specified above are for use only in the context of this assignment.

**Output Data.** Your program should write the results of its computations to an output file. An example of the required output file can be found here: [sample\_grade\_output.txt](https://ils.unc.edu/~gotz/courses/560/a/sample_grade_output.txt). This sample file contains fake grade data for 14 students. Each student record spans 2 lines in the data file. The first line contains the student name. The second line contains the student's letter grade.

**Basic Requirements**

Satisfying all basic requirements perfectly, with no points deducted for any reason, would earn a maximum score of 8 out of 10 for this assignment.

Your program should perform the following:

* Prompt the user to enter the name of the input file, making sure that the file exists and asking the user to re-enter a filename if needed.
* Prompt the user to enter the name of an output file. The output file should be erased/overwritten if an old one with the same name exists.
* Read the input file, assign grades as appropriate for the type of student (GRAD vs. UNDERGRAD), and write the output to file.
* Gracefully handle errors in the input file. In particular, your program should catch errors such as invalid numbers for grades, or student categories that are not "GRAD" or "UNDERGRAD". If found, the program should display an error message telling the user what the problem was. The program should then stop processing the rest of the data file and tell the user to fix the problems before retrying. During grading, your program will be tested with input files that have had errors intentionally introduced. It is suggested that you try this in your own testing to make sure that your program works as expected.

Two kinds of output should be produced by your program. First, an output data file should be created using the same format described in the "Output Data" paragraph of [the Overview section above](https://ils.unc.edu/~gotz/courses/560/a/5.html#overview). Second, your program will produce output on the console in the form of status messages and prompts for user input. The output produced by your program should be nearly identical to [the sample output provided below](https://ils.unc.edu/~gotz/courses/560/a/5.html#output). While the specific data (names, grades, etc.) will vary based on the input provided, the data formats, prompts, and messages printed to the console when using your program should match those in the sample output files to receive full credit.

**Please Note:** Your program should use the flow control and conditional constructs that have been discussed in class to "stop processing the rest of the data file." In this way, the program should continue to run until the end, with your logic telling the program what to do (or what not to do). You should NOT use any special Python functions to "immediately stop" your program (e.g., using sys.exit() or a similar function is not allowed).

**Advanced Requirements**

Satisfying both the basic and advanced requirements perfectly, with no points deducted for any reason, will result in a full 10 out of 10 score for this assignment.

Expand on the basic requirements by extending your program as follows.

* Allow the user if they wish to subject the grades to a curve prior to grading.
* If the user declines to curve the grades, the program should proceed as outlined in the Basic Requirements.
* If the user chooses to curve the grades, the program should ask the user to enter the number grade that should be treated as a "100" for grading purposes.
* Once the should enters a number, the program should use this to curve the grades using linear interpolation. For example:
  + If a user enters "50" as the score to use for the curve, then the following "curved grades" should be used when assigning letters:
    - 25 becomes 50
    - 50 becomes 100
    - 100 becomes 200
  + If a user enters "80" as the score to use for the curve, then the following "curved grades" should be used when assigning letters:
    - 40 becomes 50
    - 80 becomes 100
    - 100 becomes 125
* Pay attention to possible exceptions. Make sure that your code for this feature is robust.

**Sample Output**

An example of the output produced by my solution to this assignment can be found [here](https://ils.unc.edu/~gotz/courses/560/a/5_sample_output.txt). When the user elects to curve the grades, the output looks like [this](https://ils.unc.edu/~gotz/courses/560/a/5_sample_output_with_curve.txt). When an error in the data file is detected, the output looks like [this](https://ils.unc.edu/~gotz/courses/560/a/5_sample_output_exception.txt).

The above files all show what is shown to the user on the screen. There is also the output data file, of course, which contains the assigned letter grades if no error occurs. An example of the data output file is available [here](https://ils.unc.edu/~gotz/courses/560/a/sample_grade_output.txt).

**Grading Criteria**

This assignment will be graded on a 10 point scale. Your grade for this assignment will be based on a combination of factors including:

* Correct functionality (e.g., Does your Python code do what it is supposed to do as outlined in the basic and/or advanced requirements?)
* Clarity of your solution (e.g., Did you solve the problem directly and efficiently? Or is your answer excessively complex and/or inefficient?)
* Coding style (e.g., Is your code readable with comments, meaningful variable names, and good whitespace/indentation?)
* Meets submission requirements (see below)

**Seeking Help**

For general questions about Python, please use the class forum on Piazza to seek assistance. For questions that are personal in nature or that would reveal a solution to the assignment, you ask for help by email or during office hours. However, please note that emailed questions will not receive an immediate response. It is likely that it will take 24-48 hours for me to respond.

**Submitting Your Solution**

**Please Note:** You must name the main python file for your assignment "assignment<number>\_<onyen>.py". For example, for assignment 3 I would name my file assignment3\_gotz.py because my onyen is gotz.

**Using the wrong name for your file may cost you points on your assignment grade.** Please follow this requirement carefully!

Please submit your assignment via Sakai. You should submit a zip file containing your entire project folder. To create the zip file, follow the submission instructions included in the syllabus posted to Sakai.

The due date for this assignment can be found in the Assignments section of Sakai.