Your assignment is to write a program that automates an instructor's grade book. This program processes commands to add student names, record and alter scores, calculate a final semester score and print out the student information in different orders. The instructor (the user) interacts with the program through a simple command-based user interface.

Input

1. The retained student information, generated in previous executions of this program, is input from file "Grades.dat" at the beginning of each execution of the program.
2. The commands from the instructor are input (in response to "user-friendly" prompts from the program) as single letters. The commands are discussed in detail in the Processing instructions below.

Output

3. Responses to user commands are to be written to the screen, as described in the Processing instructions below.
4. The output from command O (Output) should be printed to file "Grades.out", as described in the Processing instructions below.
5. The updated grade information must be saved to file "Grades.dat".

Processing

The user commands are printed to the screen as a menu, and the user responds with a single-letter command. Some commands require the program to prompt the user to enter additional information. After each command is processed, the menu should be redisplayed. The commands are to be processed as described in the following table:

OOD Process

- Understand the requirements
- Draw Use Case Diagram(s)
- Design the business objects
    - Identify the data attributes
    - Identify the classes by storing the data attributes into categories.
    - Identify the methods by drawing a UML diagram for the class.
    - Refine the classes, attributes, and methods.
- Implement the business object and test them
- Implement the presentation layer
    - Define a class if needed to encapsulating the complexity.
- Implement the data layer and test it
    - Define file formats for “policy“, Grades.dat” and “Grades.out”.
    - Define a class for storing and retrieving the data from the file.

Deliverables

You should upload the following files individually to OSC website:

1. A PDF contains use case diagram(s) and all class diagrams.
2. A PDF contains your program execution outputs. You should have multiple execution outputs from various test cases for testing all the commands.
3. A ZIP file contains all your Python source files including data files used in your program.

Command Instructions

Command	Description
S	Set up for new semester. The program must prompt the user for the number of programming assignments (range 0 .. 6), the number of tests (range 0 .. 4), and final exams (range 0 .. 1). It must also prompt the user for the relative weights of programs, tests, and final exam in determining the students' grades. The relative weights are expressed as percentages of the semester grade, and must add up to 100%. The information will be saved to a separate file called "policy".
A	Add a student. The program must prompt student's ID (integer in the range 1 - 9999), and the user for the student's name (last name and first name, strings of at most 20 characters).
P	Record programming assignment score for all students. The program must prompt the user to ask which assignment number is to be recorded. Upon receiving a valid assignment number, the program will prompt the user with each student's name and ID.
T	Record test score for all students. The program must prompt the user to ask which test is to be recorded. Upon receiving a valid test number, the program will prompt the user with each student's name and ID.
F	Record Final exam score for all students. The program must prompt the user with each student's name and ID.
C	Change a grade for a particular student. The program must prompt the user to get the student ID, the new score, and the type of score to change (P, T, or F, as described above). Upon receiving valid information, the program will also prompt the user to enter assignment number, test number, or exam number depending on the type of score.
G	Calculate final score. Add the program scores, test scores and final exam scores according their weights for each student. And store the final score in the student's record.
O	Output the grade data, ordered alphabetically by name (last name) or by student's ID (in increasing order). The name, student's ID, assignment score(s), test score(s), final exam score(s), and the final score must be printed out in the order indicated, to file "Grades.out". Format the data with appropriate labels.
Q	Quit. Save all the student record data to the "Grades.dat" file, and terminate the program.

NOTE: The S (Setup) command must be made (once per semester) before any of the other commands can be executed. If this command is made in a particular execution of the program, the "Grades.dat" input file should be ignored, and the grades data structure should be initialized to the empty state.