CSCI-C 200

**Final Project**

**Description**

You will implement a mini version of the registrar system, with data stored in csv files. The application will include an admin module, a student module and an instructor module. When the admin module runs, an admin will login and be able to add students, add instructors, add courses and view information entered in this module (through different runs). The data added will be saved into csv files. When the student module runs, a student will login and be able to select a course to enroll and see all courses they are enrolled in. When the instructor module runs, an instructor will see all the courses he/she is assigned to teach. All data entered will be saved into csv files (one for student info, one for instructor info, one for course info, and one for enrollment info).

**Requirements and Rubric**

1. Admin Module
2. Admin login control: manually create a csv file to store admin usernames and passwords(at least three admins). This file will be used for admin login control. This is the only csv file created by you manually. All other csv files will be generated through your program. (5’)
3. An admin will only be allowed to enter the wrong username/password combination 5 times. (5’)
4. An authenticated admin will be provided options to
   1. add a new student. Information must include but not be limited to first name, last name, unique username and password. Must ensure that each username is unique among students. Student information will be saved into a csv file. (5’)
   2. add an instructor. Information must include but not be limited to first name, last name, unique username, password, title such as assistant professor/associate professor/professor. Must ensure that each username is unique among instructors. Instructor information will be saved into a csv file. (5’)
   3. add a new course. Information must include not be limited to a unique course number, course title, instructor username. Must ensure that each course number is unique. Course information will be saved into a csv file. (5’)
   4. See all the student information. Outputs must be nicely formatted. (5’)
   5. See all the instructor information. Outputs must be nicely formatted. (5’)
   6. See all the course information. Outputs must be nicely formatted. (5’)
   7. See all the enrollment information. Outputs must be nicely formatted. (5’)
5. Student Module
6. Login control: use the csv file generated in the admin module to authenticate a student. A student will only be allowed to enter the wrong username/password combination 5 times. (5’)
7. An authenticated student will be given an option to
   1. enroll in a course. The student will see a list of the courses entered by admins and select one to enroll. Enrollment information (unique student username, and unique course number) will be saved into the enrollment csv file. (5’)
   2. see all the courses he/she is enrolled in. The data will be retrieved from the enrollment csv file. (5’)
8. Outputs must be nicely formatted. (5’)
9. Instructor Module
10. Login control: use the csv file generated in the admin module to authenticate an instructor and will only be allowed to enter the wrong username/password combination 5 times. (5’)
11. An authenticated instructor will see all the courses he/she is assigned to teach. The data will be retrieved from the course file. (5’)
12. Outputs must be nicely formatted. (5’)
13. For All Modules
14. Use utility file(s) to store utility functions, such as user verification, login control, add a student, add an instructor, add a course, add an enrollment, view students, view instructors, view courses, view enrollments, check uniqueness etc. (5’)
15. Object Oriented programming methods must be used.
    1. The class created must include but is not limited to Student, Instructor and Course. Define meaningful properties and methods in each class. One of the properties of a Course class must be an instructor object. (5’)
    2. Create a parent class called User, and inherited by Student, Instructor, and Admin classes. Inheritance must be well implemented. (5’)
16. Provide an option for a user to exit the program easily. (5’)

**Submission**

Put all your files in one folder, zip up the folder, and submit the zip file. Your zip file name must be *YourFullName*FinalProject.zip.