A course registration system (CRS) created using object oriented programming.

==========

Fulfills the following requirements:

1. The CRS stores the following information about each course:
   * course name
   * course ID
   * max # of students registered in the course
   * current # of registered students
   * list of names of students being registered in the course
   * course instructor
   * course section #
   * course location
2. The CRS allows three types of users: **Admin, Faculty (Staff)** and **Student**
3. The CRS allows the **Admin** to perform the following tasks:
   * **Course Management**
     + Create a new course, store in file **course.csv**
     + Delete a course
     + Edit a course (except for course ID and name)
     + Display info for a course (by course ID)
     + Register a student (allows admin to add student w/o assigning to a course)
     + Exit
   * **Reports**
     + View all courses (for every course display list of enrolled students' names, ids, # of students registered, max # of students allowed to be registered)
     + View all full courses
     + Write to a file list of full courses
     + View names of students registered to specific course
     + View list of courses a student is registered in (given first & last name of student)
     + Sort courses on current # of students registered
     + Exit
4. The CRS allows the **Student** to perform the following tasks:
   * **Course Management**
     + View all courses
     + View all courses that are NOT full
     + Register in a course **CourseStaff.csv** (student enters course name, section, student's full name) **CourseStudent.csv**
     + Withdraw from a course (student enters their name, course name)
     + View all courses student is registered in
     + Exit
5. The CRS allows the **Faculty** to perform the following tasks:
   * **Course Management**
     + View all courses
     + Register teaching in a course (staff enters course name, section, staff's full name) and store in the file **CourseStaff.csv**
     + Withdraw from a course (staff enters their name, course name)
     + View all courses staff is registered in
     + Exit
6. The CRS must implement the following design:
   * An *Interface* for admin class with signatures of **methods** used by admin.
   * An *Interface* for student class with signatures of methods used by student.
   * An *Interface* for staff class with signatures of methods used by staff.
   * Both **Admin** and **Student, Staff** classes inherit from class **User**. **User** has the following class members:
     + username
     + password
     + first, mid & last name
   * When program is launched, the CRS must read all courses' information from a file and
   * Assume there is only one Admin in the program.