Overall Design and Workflow

Overall Design:

There are five required classes when I first read the requirements. Course, user, student, admin, and main.

In the requirements, it is clear that we must first create a Course class to store all the courses. I created this class first since it has the least connection with others, meaning no need to worry about inheritance. This class includes all the variables, same in the myCourseList.csv file, to create the Course object.

Next, I created the superclass User and its child classes Student and Admin inherited from it. They each hold different methods according to their abilities, and the superclass User has the viewAllCourse method that is used in both Student and Admin. Student and Admin also have their according interface StudentInterface and AdminInterface that shows all the methods for each of the classes.

The FileOperation class was designed to hold all serialization and deserialization methods, including those for courses and students, and also for reading files. I encountered a lot of trouble dealing with the files, and I also learned how serialization work. Throughout the debugging and testing stage, I realized that each time the file is deserialized, it becomes corrupted but can still be accessed in the same way. I made it static so it is accessible using the class name straight.

Workflow:

1. Main method
    a. Test if course.ser or student.ser has been created, therefore whether or not we read files or deserialize files.
        i. Read files
        ii. Deserialize of student and course
2. Take user input, check whether the user is admin or student
    a. Admin: ask for username and password, which could not be modified
        i. Show admin menu, and ask for user input to choose what the admin wants to do; operated in a while loop and when exit: boolean = false
            - Choose from 1-11
            - depending on the choice number, methods are called from admin, and operation down on the courseList
            - Operations continue until exit=false
    b. Student: ask for username and password, check if the student exists
        i. Show student menu, and ask for user input to choose what the student wants to do; operated in a while loop and when exit: boolean = false
            - Choose from 1-6
            - depending on the choice number, methods are called from student, and operation down on the courseList
            - Operations continue until exit=false
3. Serialization of student and course