Flying Bob: Shayan Chowdhury, Jerry Ye, Addison Huang
APCS2 pd2
Final Project Proposal
2018-05-29

Flying Programming Program

General Description of our Project:

Our final project idea is to create an in-terminal application to help with programming.

Our application will allow for two different types of users that can each login with proper identification from their username or password: Administrators or Students. Usernames and passwords will be stored inside a local CSV file and parsed through to find a match. Each user will be allowed to do different things.

The administrators will have complete access to the program, and will be allowed to create courses, access any user's information, modify anyone's information, and begin creating everyone's schedule using the information available. This will be done through recursive backtracking. The administrator will also be given an option to feed in random amounts of Student and Course data for the purpose of a simulation.

The Student will have very limited access when compared to the administrator. They will be allowed to view their own information, and choose courses that they may wish to take. When choosing, they will repeatedly be given prompts such as "which <Insert subject name> course would you like to take" with various course options from the database. After they a choose a course, they will be inputted into a PriorityQueue for that course; this will be organized by the student's grade level and GPA.

Prioritized To-Do List and Rough Timeline (Deadlines may change with respect to in-class deadlines):

1. Create UML Diagram (Wednesday, May 30th)
2. Create Flowchart of Driver (Thursday, May 31st)
3. Finish a MVP (Monday, June 4th)
    a. Have our UML diagram and flowchart functional
    b. Include Login
4. Add option to allow administrator to seed sample data for purpose of simulation (Tuesday, June 5th)
5. Add Error Handling (Wednesday, June 6th)

Concepts Addressed In Our Project:

- Sort LCourses using heap sort after large amounts of data are seeded in from the administrator

- Sort LStudents using merge sort after large amounts of data are seeded in from the administrator
- Sort student's schedule using quick sort
- Use insertion sort to sort LStudents and LCourses if an administrator manually adds courses/students(data is already mostly sorted)
- Use recursive backtracking to generate schedules that will fit
- Uses a PriorityQueue as a data structure to place students in classes(students will be implementing comparable)
- Writing out to a CSV file and taking in data from a CSV file
- Uses class hierarchy to facilitate creating accounts
- Binary search through students to find specific student as an administrator using a tree
- Uses a stack to reverse a student's viewed schedule to implement printScheduleReverse()

<u>Intended Classes so far (rough draft of UML plan; methods and instance variables are not fully fleshed out as of yet):</u>

- User
    - Instance Variables:
        - Username : String
        - Password : String
    - Methods:
        - Accessors and Mutators
- Student -- extends user implements Comparable
    - Instance Variables:
        - ID : integer
        - Schedule : course[]
        - Grade : integer
        - GPA : integer
    - Methods:
        - chooseClasses()
        - getSchedule()
        - getGPA()
        - printSchedule()
        - printScheduleReverse()
        - All accessor + mutator methods…
- Course implements Comparable
    - Instance Variables:
        - teacher: String
        - period : integer
        - Subject : String
        - maxStudents : integer
        - numStudents : integer

        - waitList: PriorityQueue<Students>
- LCourses
    - Instance Variables:
        - ArrayList<courses>
- LStudents
    - Instance Variables:
        - ArrayList<students>
- Driver
- Administrator -- extends user
    - Methods
        - Method to feed in student data and coursework data
        - Create/delete anything
        - getStudent(int id)