BİL 102 – Computer Programming Project

Student Automation

Last Submission Date of the Analysis Report: May 14th, 2014 at 15:00
Last Submission Date of the Implementation: May 25th, 2013 at 23:55

1. Description of the Project

You have two choices for the project. You can write the first part of the project alone or you can create a group of two with one of your friends and write both of the parts of the project. **(The tasks of the second part are written underlined.)**

In this project, you will implement a student automation system which will carry out some tasks about courses, students and instructors.

The system will

- create and save its own database of binary files according to the add, remove and modify processes
- check the existence of the binary database files on start of the program
- read existing csv[1] files which are created by hand for initialization and create related database files, if no database file exists
- create an empty database, if no initialization file exists (in this case, the program will end giving an error for the systems which are consisting of the first part only)

The system will execute in different modes; student mode, instructor mode and system mode. These modes will be recognized by checking the user name and password of the user. The user name and the password will be passed as command line arguments to the main function. The program will ask the user to enter the username and password again after starting if no argument is passed to the main function or the username or password is wrong. Your system will carry out the following tasks.

Student Mode

- **Add a course:** Lists the available courses for the student and enables the student to choose courses to enroll. A course is available if
  - the course is open and the add/remove mode of the course is open,
  - the student fulfill the prerequisites
- **Remove a course:** Shows the course list of the student and enables the student to remove a course from the list. The student can remove the course if the course is open and the add/remove mode of the course is open.

1 http://en.wikipedia.org/wiki/Comma-separated_values

- **List courses:** Shows the enrolled courses list of the student. Asks the student to save the list. The list will be saved into "<student number>_courses.csv" file.

- **Create transcript:** Shows the grades list of the previous courses of the student and asks to save the transcript. The transcript will be saved into "<student number>_transcript.txt" file in a proper format.

Instructor Mode

- **Open a course:** Shows the list of the courses which are given by the instructor and are not open. Enables the instructor to open a course for enrollment.

- **Finalize a course:** Shows the list of the courses which are given by the instructor and are open. Enables the instructor to finish a course. A course can only be finalized if all of the students which are enrolled to the course are graded. The system must do the disenrollment task for all of the students which are enrolled to the course.

- **Assign grades:** Shows the list of the courses which are given by the instructor and are open. Asks the instructor to choose a course. Lists the students which are enrolled to the course and asks to choose a student to assign a grade for the course. (Grading task can be done only if the course is open and add/remove mode of the course is not open.) The grades will be floating point numbers.

- **Edit grades:** Shows the list of the courses which are given by the instructor and are open. Asks the instructor to choose a course. Lists the students which are enrolled to the course and asks to choose a student to edit the grade for the course. If a grade is not assigned to the student for the course before, the system will ask the instructor to assign a grade first. (Editing grades task can be done only if the course is open and add/remove mode of the course is not open.)

- **List open courses:** Shows the list of the courses which are given by the instructor and are open. Asks the user to save the list. The list will be saved into a "<name_surname>_openCourses.csv" file.

- **List grades:** Shows the list of the courses which are given by the instructor and are open. Asks the instructor to choose a course. Lists the students which are enrolled to the course with their grades. Asks the user to save the list. The list will be saved into a "<course ID>_grades.csv" file.

System Mode

There will be only one user which can enter the system in system mode. If there is no system user, the system will ask the user to add a system user and the current user will be the newly added system user. The system will not allow deleting the system user.

- **Add user:** Creates a new user. The new user can be a student or an instructor.

1 http://en.wikipedia.org/wiki/Comma-separated_values

- **Remove a user:** Removes a user. Asks the username and password of the user which will be removed. Remove process will be done in two modes.
    - **Removing a student:** A student can be removed only if there is no open enrollment for the student. If the student has completed enough number (you can specify any number) of courses, the student will be removed from the system and added to the graduated students list with the graduation grade. Otherwise the student will be removed from the system and added to the cancelled students list. All of the data of the student must be removed.
    - **Removing an Instructor:** An instructor can only be removed if there is not any open course of the instructor. The system must remove all of the data of the instructor and add the name and surname of the instructor to the previous instructors list.
- **Modify a user:** Enables to edit properties of a user.
- **Create a course:** Enables the system to create a new course. The instructor of the course will be determined here.
- **Remove a course:** Shows the list of the courses. Enables the system to remove a course. (A course can be removed only if it is not opened.)
- **Modify a course:** Shows the list of the courses. Enables the system to edit the properties of the course.

2. Data

You should design your database effectively including the following information.

- **Students:** TC Number (unique, 11 digits), name, surname, student number (10 digits), start year, enrolled courses, previous courses and grades
- **Instructors:** TC Number(unique, 11 digits), name, surname, open courses
- **Courses:** ID (unique), name, instructors, prerequisites (one or more courses), creator (a user), a variable which shows if the course is open or not, mode (add/remove is open or not)
- **Users:** ID (unique, must be same with the TC Number of the user), type (student instructor, system), user name (unique), password

3. Data Files

- **Initialization Files**

  (Do not change the name and format of the initialization files.)

1 http://en.wikipedia.org/wiki/Comma-separated_values

- **Students.csv:** A csv file of students' personal information. (TCNo, name, surname, studentNo, startYear)
- **Instructors.csv:** A csv file of instructors' personal information. (TCNo, name, surname)
- **Users.csv:** A csv file of users' information. (ID, userType, userName, passWord)

- **Database Files**

  Your system will create its own database files by using the initialization files or the add, remove and modify processes which are mentioned in 1st part.

4. Design and Implementation Details

- You will provide a full program.
- Functions will **not** accept the structures as call-by-value arguments.
- Using conditional preprocessor directives (#ifdef, #ifndef, #if, #endif, #else, #elif), provide 2 working modes: normal and debug. In debug mode, printout some critical variables to track the states of the system. In normal mode, do not make any unnecessary printouts.
- Check all of the inputs from the users. Ask the user to reenter the erroneous values.
- You should use dynamic arrays and your program will always free unused memory.
- The system will consider only one grade for each course.

5. Excellence Bonus

Bonus points will be given according to your program's algorithmic design and user friendliness. Do not insert unnecessary complexity to your code. You must create proper structures having enough number of components to represent more information. Also you should have enough number of functions and you should avoid from code repetition. You can add more meaningful functionalities to get bonus points.

6. Project Submission

You will submit 2 document printouts and 3 coding files.

- **Analysis Report**
  - Indicate if you will work as a group or alone.

1 http://en.wikipedia.org/wiki/Comma-separated_values

- A detailed description of the project with your own words (How will it work? What are the restrictions? More functionalities.)
- Define inputs and outputs of the system
- A structure chart for the project (refer to chapter 4.5 and 12.3 of the text book)
- Function prototypes and explanations (you may make some changes later but the main functionalities should be determined here)

- **Implementation and Test Report**
  - Explain the implementation of your program
  - Compare your resulting system with your goals written in your analysis report, mention differences (if there are any differences) and explain causes of the differences
  - Test your system and add screen shots of the console and the output files explaining the outputs
  - Append your code to this report
- **Coding Files**
  - PR_<student number>_main.c
  - PR_<student number>.c
  - PR_<student number>.h

1 http://en.wikipedia.org/wiki/Comma-separated_values