Case Study: C-DAC Admission System

This document proposes a modified C-DAC admisson system. Primary focus of the project is to allocate seats to the students for various C-DAC courses at various C-DAC centers based on their C-CAT ranking. Major modification considers eligibility of students for a particular course before giving center/course preferences and limiting only one counciling (with 2 rounds of seat allocation).

The basic data is made available in CSV files. Fields of CSV files are described below.

1. degrees.txt
    a. Degrees (text)
2. courses.csv
    a. Id (int)
    b. Name (text)
    c. Fees (text)
    d. Section (text) -- which CCAT section exam should be appeared for the course A, B or C
3. Eligibilities.csv
    a. Course name (text)
    b. Degree (text) -- qualifying degree for the course
    c. Min percentage (decimal)
4. Centers.csv
    a. Center Id (text) -- used as username for login
    b. Center name (text)
    c. Address (text)
    d. Coordinator (text)
    e. Password (text) -- used as password for login
5. Capacities.csv
    a. Center id (text)
    b. Course name (text)
    c. Capacity (max seats)
    d. Filled capcity (seats allocated after center allocation)
6. Students.csv
    a. Form no (int) -- used as username for login
    b. Name (text) -- used as password for login
    c. Rank a section (-1 indicate student not appeared for the section)
    d. Rank b section (-1 indicate student not appeared for the section)
    e. Rank c section (-1 indicate student not appeared for the section)
    f. Degree (text)
    g. Pecentage (decimal) -- degree score
    h. Allocated preference (int)
    i. Allocated course name (text)
    j. Allocated center id (text)
    k. Payment (decimal)

    l. Reported to center (int)
    m. Prn (text) -- Will be generated at the end as centerid_coursename_srno

7. Preferences.csv
    a. Student form no
    b. Preference no (1, 2, 3, …, 10)
    c. Course name
    d. Center id

System has three types of users i.e. Admin, Student and Center coordinator. Each user has the following relevant functionalities.

1. Student
    a. Register new student (Append student data at the end of Students.csv)
    b. Sign in
    c. List courses (as per eligibility)
    d. List centers
    e. Give preferences (Allowed to give preference only if the student is eligible for that course. Then append preferences in Preferences.csv)
    f. See allocated center/course
    g. Update payment details
2. Admin
    a. Sign in (username & password: admin)
    b. List courses & eligibilities
    c. List centers & capacities
    d. List students
    e. Update student ranks
    f. Allocate centers (Round 1)
    g. Allocate centers (Round 2)
    h. List allocated students
    i. List paid students
    j. List reported (at center) students
    k. Generate PRN
    l. List admitted students (with PRN) for given center
3. Center coordinator
    a. Sign In
    b. List courses (of that center)
    c. List students (allocated to that center)
    d. Update reported status
    e. List admitted students (with PRN)

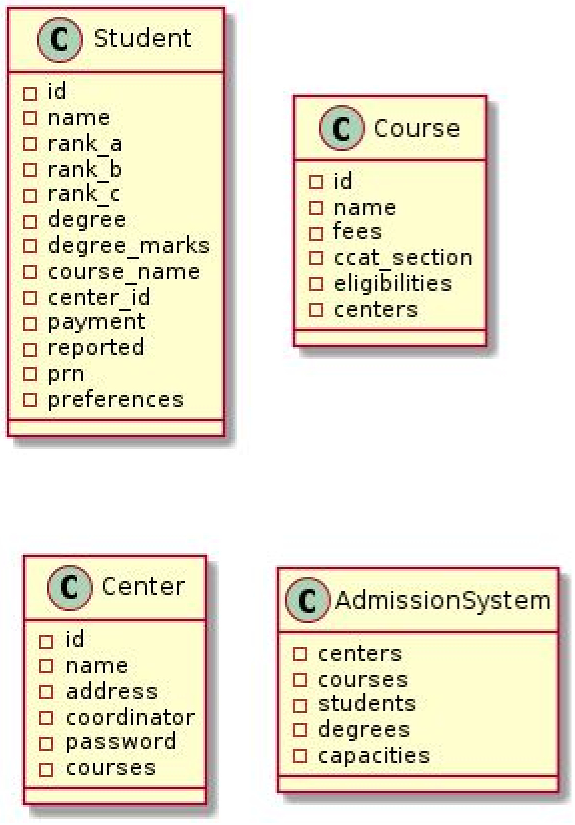
Case Study: C-DAC Admission System

Modified admission process (considered for this case study) is as follows:

1. Student registers for the C-DAC exam along with their academic details and exam category (A, B or C).
2. Students appear for exams of respective sections. The ranks (based on marks) are updated by C-DAC admin. If a student is not qualified/appeared in any section his/her rank is considered as -1.
3. As per valid rank and qualifying degree, students can give max 10 preferences for desired course at desired center as per eligibility criteria. The order of preference matters for center selection. (Data upto this step is already given in students.csv and preferences.csv file. For newly/registered added students (if any), data for these steps should be entered via this program in sequence).
4. C-DAC admin will allocate centers (Round-1) to the students as per their merit/rank (in respective sections) and their preferences. The allocated preference number, center and course should be updated (into students.csv). Also allocated seats should be updated (into capacities.csv).
5. If the center is allocated to the student, he/she should update payment of the first installment (Rs. 11,800/-) (into students.csv).
6. C-DAC admin will discard the seats allocated to unpaid students and allocate centers (Round-2) to the rest of students as per their merit/rank (in respective sections) and their preferences. The allocated preference number, center and course should be updated (into students.csv). Also allocated seats should be updated (into capacities.csv).
7. If the center is allocated to the student, he/she should update payment of the first installment (Rs. 11,800/-) (into students.csv).
8. All students allocated in Round-1 and Round-2 should also update the rest of fee payment (into students.csv).
9. On the first day of the course, the center coordinator will update the reporting (attendance) status of students (into students.csv).
10. Finally C-DAC admin will generate PRN number of all fully paid and reported students and update it (into students.csv).

Relevant data can be clubbed into structure/object/dictionary (depending on language choice). A suggested data in each object is given below. Data types should be chosen wisely depending on language used for implementation.

Case Study: C-DAC Admission System



Please consider the following points during implementing case study.

1. Implement the project as a menu driven console application. Depending on login, the corresponding menu should be visible to the user.
2. The whole data (from all files) must be loaded into the memory in appropriate data structures (collections). Entire processing will happen only in the memory and at the end (before exit) all files should be updated (as per changes done in memory).
3. Sample data is provided into CSV files, so that students need not to waste time for data entry. The data must be loaded from these files (as mentioned above).
4. Since the project focuses on the center/seat allocation algorithm, many menu items are skipped and that data is loaded from CSV files. As a future scope this data can be taken from the user (admin) like degrees, centers, courses, capacities and eligibilities. Also edit and delete features can be added for each of this data.
5. Students and mentors are encouraged to design and plan steps for implentation before coding begins.

Case Study: C-DAC Admission System

Round 1 allocation

- Allocation is done preferencewise i.e. first preference of all students (as per rank) should be allocated, then second preference of all students (as per rank) should be allocated, so on.
- For n 1 to 10 (Allocate nth preference of student)
    - Sort students by rank_a.
    - If n th preference is given by the student and preference is of course of section A and capacity of center is not full and preference is not yet allocated to the student and student fees >= 0
        - Allocate the course to the student.
            - Set preference number for the student.
            - Set the course for the student.
            - Set the center for the student.
        - Increase filled seat count into corresponding course-center capacity.
    - Sort students by rank_b.
    - If n th preference is given by the student and preference is of course of section B and capacity of center is not full and preference is not yet allocated to the student and student fees >= 0
        - Allocate the course to the student.
            - Set preference number for the student.
            - Set the course for the student.
            - Set the center for the student.
        - Increase filled seat count into corresponding course-center capacity.
    - Sort students by rank_c.
    - If n th preference is given by the student and preference is of course of section C and capacity of center is not full and preference is not yet allocated to the student and student fees >= 0
        - Allocate the course to the student.
            - Set preference number for the student.
            - Set the course for the student.
            - Set the center for the student.
        - Increase filled seat count into corresponding course-center capacity.

Round 2 allocation

1. Allocation of the vacant seats due to non-payment of the first installment by few students to whom the center was allocated in Round1.
2. Also students to whom their 2nd or 3rd or next prefernce (other than first preferences) was allocated, can be upgraded if seats are available for their upper preference.
3. Algorithm:

3.1. Mark students to whom the center was allocated in Round1, but not paid first installment (fees=-1).
3.2. Clear allocated preferences of all students (preference=0, course=NA, center=NA).
3.3. Clear filled capacities of all the centers/courses (filled=0 for all capacities).
3.4. Call Round1 allocation algorithm.

Case Study: C-DAC Admission System

Menu driven approach and expected functionality

It is suggested to write two functions save_students_csv(), save_capacities_csv() and save_preferences_csv() to write students and capacities array/vector/list back to the respective files. Note that you need to write the whole array/vector/list into csv file (editing individual records in csv is a tedious task).

The application should have nested menus. The first level menu is Register new student and Sign In. Based on sign in information keep track of current login i.e. current student or current center coordinator or admin. The rest of the menus should be displayed as per login.

Register new student

1. Input student details (name, degree, percentage) from the user and append to the students array. Ensure that degree must exist into degrees.txt file.
2. Then call save_students_csv().

Student -> List courses

1. Compare student degree and marks for each course eligibility and display only those courses for which student is eligible.

Student -> List centers

1. List all centers

Student -> Give preferences

1. Ask the user for the number of preferences he wants to enter.
2. Show him courses he is eligible for along with centers that offer the course. Student will select preferences in the order.
3. Add them into the student (current student object) preferences (array/vector).
4. Then call save_preferences_csv()

Student -> See allocated center/course

1. Display allocated center & course (from current student object)

Student -> Update payment details

1. Display paid fees (from current student object)
2. Input total fees paid by student and set into student (current student object)
3. Then call save_students_csv().

Admin -> List courses & eligibility

1. Display all courses along with their eligibilities

Case Study: C-DAC Admission System

Admin -> List centers & capacities

1. List all centers along with their capacities for each course

Admin -> List students

1. List all the students.

Admin -> Update student ranks

1. Input form no, rank a, rank b and rank c from user and update into students array.
2. Then call save_students_csv().

Admin -> List allocated students

1. List students to whom some center is allocated - sorted by course name (1st level), center name (2nd level) and name (3rd level).

Admin -> List paid students

1. List students who have paid the fees (1st installment or full fees) - sorted by course name (1st level), center name (2nd level), name (3rd level) and fees paid (4th level).

Admin -> List reported students

1. List students who have reported to the center - sorted by course name (1st level), center name (2nd level) and name (3rd level).

Admin -> Generate PRN

1. Sort students by course name (1st level), center name (2nd level) and name (3rd level).
2. Generate PRN number for the students who are reported to the center and also paid full course fees (as given in Course object).
3. PRN format should be CourseName_CenterId_SrNo e.g. PG-DAC_SIP_1, PG-DAC_SIP_2, PG-DAC_SIP_3, ...
4. Call save_students_csv()

Admin -> List admitted students

1. Input course name and center id from user.
2. List students in sorted order of their name for the course and center.

Center coordinator -> List courses

1. List all courses offered at center (current center object)

Center coordinator -> List students

1. Input course name from user
2. List all students for current center given course (sorted by name)

Case Study: C-DAC Admission System

Center coordinator -> Update reported status

1. Input form no of student and his reported (attendance) status.
2. If student is allocated to the center (any course) then update reported status in student object
3. Otherwise give error message, indicating student is not allocated to that center
4. Call save_students_csv()

Center coordinator -> List admitted students

1. Input course name from user.
2. List students in sorted order of their name for the course and center.

Happy Programming!