CSL 213 : Data Structures and Program Design I
H/W Assignment

1. Announcement Date: **31st October 2021, 7:00 PM**

2. Due date: Submit online by **11.59 PM on Wednesday 10th November, 2021**

3. The assignment has to be done individually.

4. No Copying / sharing of code is allowed for this assignment. If any such case is identified, the original author and person who has copied, both will be penalised equally and zero marks will be awarded.

5. You need to submit your source files by attaching them to a mail and sending it on **dspd.assignment@gmail.com** by the common deadline. Please attach .c and/or .h and/or .txt files only. No .obj or.exe files should be attached. The subject should be marked as DSPD-1-HW-Assignment: Your enrolment no.

<u>Problem for R1 Batch (Enrolment Numbers BT20CSE001 to BT20CSE025):</u>

CoProtect is a covid vaccination platform available for the citizens of India to book their appointment for vaccination against Covid-19 disease. You need to implement this platform using *array of structures* wherein citizens must be able to register themselves and book an appointment for them to receive the vaccine at the desired centre. Your program should have at least two structures to store the data of citizens and the authorized centres.

The structure for citizens should contain the following fields:

- 5-digit beneficiary id
- Beneficiary full name
- Gender
- Year of birth
- Number of doses taken (0/1/2)

The structure for the vaccination centre should contain the following fields:

- Centre ID
- Centre address
- District
- PIN code
- Center type (government/private) [Only government-run centres are authorized to provide vaccines at free of cost]
- Vaccine types (covaxin/covishield/sputnik/zydus/Pfizer/moderna) that are available with their respective quantity available for the day and the price fixed for each vaccine type.

Implement the following functions to provide the facilities to the user. You can implement other functions as per your need. *All the records for centres and beneficiaries must be kept in sorted fashion (after any operation) based on centre ID and beneficiary ID respectively.*

a. addNewBeneficiary():
    - *Input: Beneficiary full name, gender, year of birth [Beneficiary id should be a unique and system generated number]*
    - *Output: Display all the information after successful addition of record.*
b. addNewCenter():
    - *Input: centre ID, address, district, PIN code, centre type, vaccine type with their respective quantity available for the day.*
    - *Output: Display the message after successful addition of record.*
c. EnlistSortedCenters ():
    - *Input: Key field for the sorted list (Center ID/PIN code/District-wise)*
    - *Output: Display the sorted list of centres based on key field.*
d. getCentre():
    - *Input: Parameter for searching the center (Center ID/District/PIN code/center type/vaccine type)*
    - *Output: Display the list of centers based on search parameter.*

e. BookAppointment():
    - *Input: Beneficiary ID, Center ID, vaccine type*
    - *Output – Display the message after a successful booking.*
f. CancelAppointment():
    - *Input: Beneficiary ID*
    - *Output: Display the message after a successful cancellation.*
g. removeBeneficiary():
    - *Input: Beneficiary ID*
    - *Output: Display the message after the successful removal.*
h. removeCentre():
    - *Input: Center ID*
    - *Output: Display the message after the successful removal.*
i. UpdateBeneficiaryInfo():
    - *Input: Beneficiary ID and the fields to update*
    - *Output: Display the message after a successful update.*
j. UpdateCenterInfo():
    - *Input: Center ID and the fields to update*
    - *Output: Display the message after a successful update.*
k. MaxIncomeCentre():
    - *Input: None*
    - *Output: Print the centre information where the maximum income (in Rs.) has been generated.*
l. VaccineAvailabilityReport():
    - *Input: None*
    - *Output: Generate a report with list of centres (sorted district-wise) and the number of vaccines available (at any given time) for each vaccine type.*
m. DistrictReport():
    - *Input: None*
    - *Output: Total number of vaccines booked in each district.*
n. SeniorCitizenIntersection():
    - *Input:*
        i. *List1 - List of all senior citizens (age > 60) who have booked their 1st dose*
        ii. *List2 – List of senior citizens (age > 60) who have booked their 2nd dose*
    - *Output: List of senior citizens who booked both dosages at the same centre.*

Your program must display the options available for the user on a particular day. The system should take care of the following important things:

- Your database must be dynamic to reflect the changes made by the user on a particular day.
- Once the appointment is booked or cancelled at a particular centre, the change must reflect specifically on the number of doses available at that centre.
- Citizens above 18 years only are allowed to take a vaccine.
- No citizen is allowed to take more than 1 dose on the same day.
- The citizen must be shown the price list of vaccines at the time of booking.

Problem for R2 Batch (Enrolment Numbers BT20CSE026 to BT20CSE050):

An owner of the Hospital in Nagpur needs to employ a hospital management system using array of structures to keep the track of hospital activities. The following structure defines the data maintained for each patient in the hospital:

```
struct patient
{
        int PatientNumber;
        int age;
        char gender;
        char FirstName [20];
        char LastName [20];
        char PhoneNumber [10];
        char ResidentialCity [15];
        char Email [30];
        char Doctor [20];
        char Problem [30];
        int ServiceType;

        struct charges
        {
                char TotalCharge [15];
                char TotalDeposited [15];
                char TotalReturn [15];
        };
}
```

The field Service takes a value 0 or 1 indicating whether the type of service required is Outdoor Patients Department (OPD) or Emergency service.

The key functions to be implemented in hospital management system are:

1. **Add a New Patient Record** which reads a record from user and insert into a list of structures choosing between O.P.D. service and Emergency service. The information to be entered by the user is same in O.P.D. service and Emergency service. The new records must be added in such a way that patients with Service value 0 must be stored first, sorted sequentially with respect to their PatientNumber, followed by patients with Service value 1, again sorted sequentially with respect to their PatientNumber. This order must be preserved throughout.

2. **Search Patient Record** User should be able to search either via. full name of the patient or by patient number. All the information corresponding to the respective patient should be displayed. These include the ones provided while adding a new patient record. If

wrong information about patient full name or number is provided, the program displays a message saying that no records are available. Also, user should be able to view the list of expenditures of the particular patient whose record is sought, including the financial information of total charge, total deposited and total money to return.

3. **Edit Patient Record** User should be able to edit either via. full name of the patient or by patient number. All the information corresponding to the respective patient should be displayed. If wrong information about patient full name or number is provided, the program displays a message saying that no records are available. Editing option should ask for the field/fields to be edited and replace the new value in the record.
4. **List record of patients'** User should be able to list patient records by choosing any one of the four options listed below:
   a. Records of patients in alphabetical order
   b. Records of Emergency patients
   c. Records of O.P.D. patients

   As in the two features mentioned above, user should be able to view financial records corresponding to any particular patient listed according to any one of the four options mentioned above.
5. **Delete Patient Records** this allows user to delete added record of any patient. For this the patient number to be removed is to be provided. Upon 'Enter', user can view the patient record and the financial records of the patient. To delete the record, press 'Enter' and the respective patient record should be deleted.
6. **Print outsider patient data** User should be able to retrieve patient records of all outsider patients being treated at the hospital. (Patients residing outside Nagpur).
7. **Maximum and Minimum charge of Treatment** Write a function which retrieves the record of the patient who was charged:
   a. Highest by the hospital. If there are multiple patients with same maximum amount charged then print all.
   b. Lowest by the hospital. If there are multiple patients with same minimum amount charged then print all.
8. **Funds Return Data** User should be able to list all the patients whose money has to be returned before discharge along with respective amount to be returned.
9. **Search patients by Age** Write a function to find out the patients availing Emergency services in the age range of:
   a. 25-40 years
   b. 40 + years

10. **Search patients by Doctor** User should be able to find out the names of the patients taking treatment from a particular doctor entered by the user.
11. **View Doctors** Write a function to find out doctors treating:
    - **a.** both OPD and Emergency patients.
    - **b.** Treating only Emergency Patients.

Problem for R3 Batch (Enrolment Numbers BT20CSE051 to BT20CSE075):

Write a program in C to manage information of Cars and its owners using array of structures. There are two structures in a program named as Vehicle and Owner. Variables in Vehicle structure are:

Structure of Vehicle:

Attribute Name	Data Type	Example
vehicle-type	char	2wheel, 3 Wheel, 4wheel etc.
maker	char	Honda, Toyota, Suzuki etc.
year-of-manufacture	int	2012 etc.
engine-no	char	BR-29536 etc.
registration-no	char	LEV-0012 etc.
vehicle-price	unsigned int	48000 etc.
owners_ID	int	1,2,3 etc..

An array with the IDs of this car's owners is owners_ID. An owner can have a maximum of 5 cars.
Variables in Owner structure are:
Structure of Owner

Attribute Name	Data Type
owners_ID	int
owner-name	char
father-name	char
address	char
date-of-purchase	int
month-of-purchase	int
year-of-purchase	int
purchase-amount	double

1. Write a function to add a new owner (making sure they don't exist already).
2. Write a function to add a new car to the owner. (if the owner already has 5 vehicles then don’t allow new purchase)
3. Delete a car by ID (also delete car’s entry from owner of this car and if the owner doesn't have another car then delete that owner’s entry too)
4. Write a function to list all the cars of particular owner in ascending order of year of manufacture.
5. Delete all cars from the structure above that have a manufacturing year before 1990. (Also delete car’s entry from owner of this car and if the owner doesn't have another car then delete that owner’s entry too)
6. Write a function to find out owner with maximum and second maximum number of vehicles. (multiple entries should be retrieved).
7. Find out the owners who purchased more than 1, 2-wheeler vehicles in the same year.

8. Write a function to find out the price of the costliest vehicle owned by the owner entered by the user. (User can enter the name of the owner or Owner ID)
9. Find the owners who own more than one vehicle of the brand chosen by the user.
10. List the owners who purchased 3-wheeler in the year provided by the user.

<u>Problem for R4 Batch (Enrolment Numbers BT20CSE076 to BT20CSE100):</u>

Online streaming services such as Netflix keep track of TV shows and movies as well as user's viewing preferences. Use the following entities and their attributes –

1. TV_Shows
    a. Name (primary key)
    b. First telecast date
    c. Total number of episodes
    d. Telecast days (use array)
    e. Original release date (if it has been already telecasted on TV)
    f. Cast
    g. Producer
    h. Writer
    i. Production company
2. Movie
    a. Name (primary key)
    b. release date
    c. Cast
    d. Producer
    e. writer
    f. soundtrack
    g. production_company
    h. budget
    i. box_office collection
    j. Awards (type of award, category, date of ceremony,receipients, result (won/nominated)
3. Cast
    a. name (you can use separate structure to store personal information about star cast) (primary key)
    b. TV shows
    c. Movies
    d. Awards (use separate structure to maintain details such as year, for which TV show or movie, use same fields as award category in movie structure)
4. User
    a. Identification number(primary key)
    b. registration date
    c. subscription details
    d. subscription plan
    e. subscription database (use separate structure to maintain date and amount of subscription transactions till date.)
    f. preferences
    g. viewing details (separate structure should be maintained for storing shows/movie viewed, viewing time,rating given etc.)

Based on above details solve the following questions (use array of structure to store database and all the records should be sorted as per the primary key).

1. **Add /update operation**
    a. Add a new TV show. While adding if any new cast is found update the cast database (same criteria should be followed for adding new movie details). For existing cast update the details of movie or TV show in cast database.
    b. Add new user and while adding if any of his/her preference matches with existing any users preferences show the list and allow him to select from them if he/she wishes.
    c. Add new cast and while adding new cast details cross verify award details with award details in movie and TV show database.
2. **Search operation**
    a. Display the details of all movies and TV shows performed by a cast. A cast name should be inputted by user. The list should be displayed in descending order of year.
    b. Display the list TV shows and movies subscribed by the user . While displaying, give the warning if subscription is about to end as the plan.
    c. Make a list of TV shows and movies produced by the same production company for the given cast details (cast details provided by the user).
    d. List all the type of awards won by the particular movie for different categories. Movie name should be given by user.
3. **Sort operation**
    a. List of all type of awards won/nominated by the cast movie-wise and TV show-wise. The list should be displayed in descending order of year of won/nominated.
    b. Divide the subscribed user in three categories. Users who are subscribed for more than 3 years – platinum user, 1.5-3 years- Gold user and less than 1.5 years- Silver user.
    c. Sort the TV shows based on users viewing time.
4. **Other operations**
    a. Identify the movie viewed by maximum number of users in a particular month of the year.
    b. List the TV shows and movies for the same casting details

<u>Problem for R5 Batch</u>

<u>(Enrolment Numbers BT20CSE101 to BT20CSE128+Ex-students):</u>

Now a day's gaming across network is very popular called as social gaming. It needs to store the database of following entities

1. Player information
    a. player Id (primary key)
    b. name
    c. registration date
    d. games played and won ,scores, prizes etc.(another structure can be used to store details).
    e. game preferences
2. Game master details
    a. Game id(primary key)
    b. name
    c. type (single user /multiple user)
    d. no. of players required.
3. Games played details
    a. game id
    b. players id (multiple if played with friends)
    c. score/awards /active days in a game per user details
    d. date of game start
    e. date of game end
4. Guest user details
    a. id (primary key)
    b. name
    c. date

Based on the above structures solve following questions (use array of structure to store database and all the records should be sorted based on primary key)-

1. **Add/Update operation**
    a. Add new user. While adding new user give him the game preference for games which are mostly played by other users.
    b. Update the game played details when any new user joins the particular game. If the user is not registered, make the user as guest. Give him game preference of other users of the game which he is playing.
    c. If the guest user is playing any game for more than 15 days, don't allow him to play anymore until he/she completes the registration process. Registration process will update players' database.
2. **Search operation**
    a. A player 'P' is playing the game 'G'. Display the list of games played by other players for the game 'G' played by 'P'. The value of 'P' and 'G' given by user.

  b. Display the game wise list of inactive users .Inactive users are the users who have not played the game till end.
  c. Write a function active_users() to display the active user details. Active users are users who are **currently** playing more than ‘K’ number of games . The value of ‘K’ is the input from user.
3. **Sorting Operation**
  a. Display the game details for which players win prizes mostly.
  b. Make a list of top 5 game details which are played for highest number of days.
  c. Display the player’s details who are currently playing any game ‘G’. The list should be printed in descending order.
4. **Delete Operation**
  a. Write a function remove_guest() to delete the all the guest who are active for any game for more than 20 days. While removing make sure all dependent records should be deleted from all the repositories.