```c
/*Create a structure Vehicle with the following members:
char registrationNumber[15]
char model[30]
int yearOfManufacture
float mileage
float fuelEfficiency
Implement functions to:
Add a new vehicle to the fleet.
Update the mileage and fuel efficiency for a vehicle.
Display all vehicles manufactured after a certain year.
Find the vehicle with the highest fuel efficiency.
Use dynamic memory allocation to manage the fleet of vehicles.*/
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct vehicle
{
    char registrationNumber[15];
char model[30];
int yearOfManufacture;
float mileage;
float fuelEfficiency;
};
void update(char reg[15],struct vehicle *v,int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(reg,v[i].registrationNumber)==0)
        {
            printf("Enter the fuel efficiency\n");
            scanf("%f",&v[i].fuelEfficiency);
            printf("Enter mileage\n");
            scanf("%d",&v[i].mileage);
        }
        else
        {
            printf("Vehicle not found\n");
        }
    }
}
```

```c
void display(int y, struct vehicle *v,int n)
{
    for(int i=0;i<n;i++)
    {
        if(y<v[i].yearOfManufacture)
        {
            printf("Registration number : %s",v[i].registrationNumber);
            printf("Model : %s",v[i].model);
            printf("Year of manufacture : %d\n",v[i].yearOfManufacture);
            printf("Mileage : %.2f",v[i].mileage);
            printf("Fuel efficiency %.2f",v[i].fuelEfficiency);
        }
    }
}
void efficiency(struct vehicle *v,int n)
{
    float fe=0;
    for (int i=0;i<n;i++)
    {
        if(fe<v[i].fuelEfficiency)
        {
           fe=v[i].fuelEfficiency;
        }
        printf("Highest fuel efficiency is %.2f",fe);
    }
}
void main()
{
    int n,y;
    char reg[15];
    printf("Enter the number of vehicles\n");
    scanf("%d",&n);
    struct vehicle *v=(struct vehicle *)malloc(n*sizeof(struct vehicle));
    for(int i=0;i<n;i++)
    {
        printf("Enter reg number\n");
        scanf("%s",v[i].registrationNumber);
        printf("Enter the model\n");
        getchar();
        scanf("%s",v[i].model);
```

```
        printf("Enter the fuel efficiency\n");
        scanf("%f",&v[i].fuelEfficiency);
        printf("Enter mileage\n");
        scanf("%d",&v[i].mileage);
    }
    printf("Enter the reg number to update mileage and efficiency\n");
    scanf("%s",reg);
    update(reg,v,n);
    printf("Enter the year\n");
    scanf("%d",y);
    display(y,v,n);
    efficiency(v,n);
}
```

```
2009
Enter the reg number to update mileage and efficiency
r1
Enter the fuel efficiency
56
Enter mileage
78
Enter the year
2004
Registration number : r2Model : m2Year of manufacture : 2005
Mileage : 0.00Fuel efficiency 45.00Registration number : r3Model : m3Year of manufacture : 2009
Mileage : 0.00Fuel efficiency 70.00Highest fuel efficiency is 56.00Highest fuel efficiency is 56.00Highest fuel efficiency is 70.00
PS D:\projects\quest\C>
```

```
/*Define a structure CarRental with members:
char carID[10]
char customerName[50]
char rentalDate[11] (format: YYYY-MM-DD)
char returnDate[11]
float rentalPricePerDay
Write functions to:
Book a car for a customer by inputting necessary details.
Calculate the total rental price based on the number of rental days.
Display all current rentals.
Search for rentals by customer name.
Implement error handling for invalid dates and calculate the number of
rental day
*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
struct CarRental {
```

```
    char carID[10];
    char customerName[50];
    char rentalDate[11];  // Format: YYYY-MM-DD
    char returnDate[11];  // Format: YYYY-MM-DD
    float rentalPricePerDay;
};

int calculateDays(const char *startDate, const char *endDate) {
    int startYear, startMonth, startDay;
    int endYear, endMonth, endDay;
    sscanf(startDate, "%d-%d-%d", &startYear, &startMonth, &startDay);
    sscanf(endDate, "%d-%d-%d", &endYear, &endMonth, &endDay);
    int startDays = startYear * 365 + startMonth * 30 + startDay;
    int endDays = endYear * 365 + endMonth * 30 + endDay;

    return endDays - startDays;
}
void bookCar(struct CarRental *rentals, int *count) {
    printf("Enter car ID: ");
    scanf("%s", rentals[*count].carID);
    printf("Enter customer name: ");
    scanf(" %[^\n]", rentals[*count].customerName);
    printf("Enter rental date (YYYY-MM-DD): ");
    scanf("%s", rentals[*count].rentalDate);
    printf("Enter return date (YYYY-MM-DD): ");
    scanf("%s", rentals[*count].returnDate);
    printf("Enter rental price per day: ");
    scanf("%f", &rentals[*count].rentalPricePerDay);
    (*count)++;
}
void calculateRentalPrice(struct CarRental *rentals, int count) {
    for (int i = 0; i < count; i++) {
        int days = calculateDays(rentals[i].rentalDate,
rentals[i].returnDate);
        if (days < 0) {
            printf("Error: Invalid dates for car ID %s\n",
rentals[i].carID);
        } else {
            float totalPrice = days * rentals[i].rentalPricePerDay;
            printf("Car ID: %s, Customer: %s, Total Rental Price: %.2f\n",
```

```
                    rentals[i].carID, rentals[i].customerName, totalPrice);
        }
    }
}
void displayRentals(struct CarRental *rentals, int count) {
    printf("Current Rentals:\n");

printf("-----------------------------------------------------------------\n");
    printf("Car ID | Customer Name       | Rental Date | Return Date\n");

printf("-----------------------------------------------------------------\n");
    for (int i = 0; i < count; i++) {
        printf("%6s | %-20s | %11s | %11s\n", rentals[i].carID,
rentals[i].customerName,
               rentals[i].rentalDate, rentals[i].returnDate);
    }

printf("-----------------------------------------------------------------\n");
}
void searchRentalsByCustomer(struct CarRental *rentals, int count, const
char *name) {
    int found = 0;
    for (int i = 0; i < count; i++) {
        if (strcmp(rentals[i].customerName, name) == 0) {
            printf("Car ID: %s, Rental Date: %s, Return Date: %s, Rental
Price Per Day: %.2f\n",
                   rentals[i].carID, rentals[i].rentalDate,
rentals[i].returnDate, rentals[i].rentalPricePerDay);
            found = 1;
        }
    }
    if (!found) {
        printf("No rentals found for customer: %s\n", name);
    }
}

void main() {
    int n = 0, choice;
    struct CarRental rentals[50];
    do {
```

```
        printf("\nCar Rental System\n");
        printf("1. Book a car\n");
        printf("2. Calculate total rental price\n");
        printf("3. Display all rentals\n");
        printf("4. Search rentals by customer name\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                bookCar(rentals, &n);
                break;
            case 2:
                calculateRentalPrice(rentals, n);
                break;
            case 3:
                displayRentals(rentals, n);
                break;
            case 4: {
                char name[50];
                printf("Enter customer name: ");
                scanf(" %[^\n]", name);
                searchRentalsByCustomer(rentals, n, name);
                break;
            }
            case 5:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 5);
}
```

```
Car Rental System
1. Book a car
2. Calculate total rental price
3. Display all rentals
4. Search rentals by customer name
5. Exit
Enter your choice: 2
Car ID: c1, Customer: rahul, Total Rental Price: 580.00

Car Rental System
1. Book a car
2. Calculate total rental price
3. Display all rentals
4. Search rentals by customer name
5. Exit
Enter your choice: 3
Current Rentals:
-------------------------------------------------------------
Car ID | Customer Name       | Rental Date | Return Date
-------------------------------------------------------------
    c1 | rahul               |  2001-07-02 |  2001-08-01
-------------------------------------------------------------

Car Rental System
1. Book a car
2. Calculate total rental price
3. Display all rentals
4. Search rentals by customer name
5. Exit
Enter your choice: 4
Enter customer name: rahul
Car ID: c1, Rental Date: 2001-07-02, Return Date: 2001-08-01, Rental Price Per Day: 20.00

Car Rental System
1. Book a car
2. Calculate total rental price
3. Display all rentals
4. Search rentals by customer name
5. Exit
Enter your choice: 5
```

```
/*Create a structure SensorData with fields:
int sensorID
char timestamp[20] (format: YYYY-MM-DD HH:MM:SS)
float speed
float latitude
float longitude
Functions to:
```

```c
Log new sensor data.
Display sensor data for a specific time range.
Find the maximum speed recorded.
Calculate the average speed over a specific time period.
Store sensor data in a dynamically allocated array and resize it as
needed.*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct SensorData {
    int sensorID;
    char timestamp[20]; // Format: YYYY-MM-DD HH:MM:SS
    float speed;
    float latitude;
    float longitude;
};
void logSensorData(struct SensorData **data, int *size, int *capacity);
void displayDataInRange(struct SensorData *data, int size, const char
*startTime, const char *endTime);
float findMaxSpeed(struct SensorData *data, int size);
float calculateAverageSpeed(struct SensorData *data, int size, const char
*startTime, const char *endTime);
int isTimestampInRange(const char *timestamp, const char *start, const
char *end);
int main() {
    struct SensorData *sensorData = NULL;
    int size = 0, capacity = 0;
    int choice;
    char startTime[20], endTime[20];

    do {
        printf("\nSensor Data Management:\n");
        printf("1. Log New Sensor Data\n");
        printf("2. Display Data in Time Range\n");
        printf("3. Find Maximum Speed\n");
        printf("4. Calculate Average Speed\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
```

```
        switch (choice) {
        case 1:
            logSensorData(&sensorData, &size, &capacity);
            break;
        case 2:
            printf("Enter start time (YYYY-MM-DD HH:MM:SS): ");
            scanf("%s", startTime);
            printf("Enter end time (YYYY-MM-DD HH:MM:SS): ");
            scanf("%s", endTime);
            displayDataInRange(sensorData, size, startTime, endTime);
            break;
        case 3:
            printf("Maximum speed recorded: %.2f\n",
findMaxSpeed(sensorData, size));
            break;
        case 4:
            printf("Enter start time (YYYY-MM-DD HH:MM:SS): ");
            scanf("%s", startTime);
            printf("Enter end time (YYYY-MM-DD HH:MM:SS): ");
            scanf("%s", endTime);
            printf("Average speed: %.2f\n",
calculateAverageSpeed(sensorData, size, startTime, endTime));
            break;
        case 5:
            printf("Exiting...\n");
            break;
        default:
            printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 5);
    free(sensorData);
    return 0;
}
void logSensorData(struct SensorData **data, int *size, int *capacity) {
    if (*size == *capacity) {
        *capacity = (*capacity == 0) ? 2 : (*capacity * 2);
        *data = realloc(*data, (*capacity) * sizeof(struct SensorData));
        if (*data == NULL) {
            printf("Memory allocation failed.\n");
```

```
            exit(1);
        }
    }

    printf("Enter Sensor ID: ");
    scanf("%d", &((*data)[*size].sensorID));
    printf("Enter Timestamp (YYYY-MM-DD HH:MM:SS): ");
    scanf("%s", (*data)[*size].timestamp);
    printf("Enter Speed: ");
    scanf("%f", &((*data)[*size].speed));
    printf("Enter Latitude: ");
    scanf("%f", &((*data)[*size].latitude));
    printf("Enter Longitude: ");
    scanf("%f", &((*data)[*size].longitude));

    (*size)++;
}
void displayDataInRange(struct SensorData *data, int size, const char
*startTime, const char *endTime) {
    printf("\nSensor Data in Range %s to %s:\n", startTime, endTime);
    int found = 0;
    for (int i = 0; i < size; i++) {
        if (isTimestampInRange(data[i].timestamp, startTime, endTime)) {
            printf("Sensor ID: %d, Timestamp: %s, Speed: %.2f, Latitude:
%.2f, Longitude: %.2f\n",
                   data[i].sensorID, data[i].timestamp, data[i].speed,
data[i].latitude, data[i].longitude);
            found = 1;
        }
    }
    if (!found) {
        printf("No data found in the specified range.\n");
    }
}
float findMaxSpeed(struct SensorData *data, int size) {
    if (size == 0) {
        printf("No data available.\n");
        return 0;
    }
    float maxSpeed = data[0].speed;
```

```
    for (int i = 1; i < size; i++) {
        if (data[i].speed > maxSpeed) {
            maxSpeed = data[i].speed;
        }
    }
    return maxSpeed;
}
float calculateAverageSpeed(struct SensorData *data, int size, const char
*startTime, const char *endTime) {
    int count = 0;
    float sum = 0;
    for (int i = 0; i < size; i++) {
        if (isTimestampInRange(data[i].timestamp, startTime, endTime)) {
            sum += data[i].speed;
            count++;
        }
    }
    return (count > 0) ? (sum / count) : 0;
}
int isTimestampInRange(const char *timestamp, const char *start, const
char *end) {
    return strcmp(timestamp, start) >= 0 && strcmp(timestamp, end) <= 0;
}
```

```
Enter your choice: 1
Enter Sensor ID: 3
Enter Timestamp (YYYY-MM-DD HH:MM:SS): 2006-12-01 08:08:08
Enter Speed: Enter Latitude: Enter Longitude:
Sensor Data Management:
1. Log New Sensor Data
2. Display Data in Time Range
3. Find Maximum Speed
4. Calculate Average Speed
5. Exit
Enter your choice: Enter Sensor ID: Enter Timestamp (YYYY-MM-DD HH:MM:SS): Enter Speed: 40
Enter Latitude: 10
Enter Longitude: 11

Sensor Data Management:
1. Log New Sensor Data
2. Display Data in Time Range
3. Find Maximum Speed
4. Calculate Average Speed
5. Exit
Enter your choice: 3
Maximum speed recorded: 2005.00

Sensor Data Management:
1. Log New Sensor Data
2. Display Data in Time Range
3. Find Maximum Speed
4. Calculate Average Speed
5. Exit
Enter your choice: 4
Enter start time (YYYY-MM-DD HH:MM:SS): 2005-01-01 01:01:01
Enter end time (YYYY-MM-DD HH:MM:SS): Average speed: 0.00

Sensor Data Management:
1. Log New Sensor Data
2. Display Data in Time Range
3. Find Maximum Speed
4. Calculate Average Speed
5. Exit
Enter your choice:
```

```
/*Define a structure EnginePerformance with members:
char engineID[10]
float temperature
float rpm
float fuelConsumptionRate
float oilPressure
Functions to:
Add performance data for a specific engine.
```

```c
/*Display all performance data for a specific engine ID.
Calculate the average temperature and RPM for a specific engine.
Identify any engine with abnormal oil pressure (above or below specified
thresholds).
Use linked lists to store and manage performance data entries.*/
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct engine
{
    char engineID[10];
float temperature;
float rpm;
float fuelConsumptionRate;
float oilPressure;
};
void Add(struct engine *e,int count)
{

            printf("Enter engine id\n");
            scanf("%s",e[count].engineID);
            printf("Enter temperature\n");
            scanf("%f",&e[count].temperature);
            printf("Enter rpm\n");
            scanf("%f",&e[count].rpm);
            printf("Enter fuel consumption\n");
            scanf("%f",&e[count].fuelConsumptionRate);
            printf("Enter oil pressure\n");
            scanf("%f",&e[count].oilPressure);

}
void average(struct engine *e,int n)
{
    float st=0,srpm=0;
    for(int i=0;i<n;i++)
    {
        st+=e[i].temperature;
        srpm+=e[i].rpm;
    }
    printf("Average temperature and rpm are %.2f and %.2f\n",st/n,srpm);
```

```
}
void display(struct engine *e,int n,float ut,float lt)
{
    for(int i=0;i<n;i++)
    {
        if(e[i].oilPressure>ut || e[i].oilPressure<lt )
        {
            printf("%s\n",e[i].engineID);
        }
    }
}
void main()
{
    int n,count=0;
    char id[10];
    float ut,lt;
    printf("Enter number of engines\n");
    scanf("%d",&n);
    struct engine *e=(struct engine *)malloc(n*sizeof(struct engine));
    while(count<n)
    {
    Add(e,count);
    count++;
    }
    average(e,n);
    printf("Enter upper and lower limits\n");
    scanf("%f %f",&ut,&lt);
    display(e,n,ut,lt);

}
```

```
Enter oil pressure
80
Average temperature and rpm are 156.67 and 9000.00
Enter upper and lower limits
200 100
e3
PS D:\projects\quest\C>
```

```c
/*Create a structure ServiceRecord with the following:
char serviceID[10]
char vehicleID[15]
char serviceDate[11]
char description[100]
float serviceCost
Functions to:
Add a new service record for a vehicle.
Display all service records for a given vehicle ID.
Calculate the total cost of services for a vehicle.
Sort and display service records by service date.*/
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct service
{
    char serviceID[10];
char vehicleID[15];
char serviceDate[11];
char description[100];
float serviceCost;
};
void display_vid(struct service *s,char vid[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(s[i].vehicleID,vid)==0)
        {
            printf("Vehicle ID : %s\n",s[i].vehicleID);
            printf("Service ID : %s\n",s[i].serviceID);
```

```
            printf("Service date : %s\n",s[i].serviceDate);
            printf("Description : %s\n",s[i].description);
            printf("Service cost : %.2f\n",s[i].serviceCost);
        }
    }
}
void Total(struct service *s,char vid[],int n)
{
    float sum=0;
    for(int i=0;i<n;i++)
    {
        if(strcmp(s[i].vehicleID,vid)==0)
        sum+=s[i].serviceCost;
    }
    printf("Service cost of vehicle %s is %.2f\n",vid,sum);
}
void sort(struct service *s,int n)
{
    struct service temp;
    for(int i=0;i<n-1;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            if(strcmp(s[i].serviceDate,s[i].serviceDate)>0)
            {
                temp=s[i];
                s[i]=s[j];
                s[j]=temp;
            }
        }
    }
    for(int i=0;i<n;i++)
    {
        printf("Vehicle ID : %s\n",s[i].vehicleID);
        printf("Service ID : %s\n",s[i].serviceID);
        printf("Service date : %s\n",s[i].serviceDate);
        printf("Description : %s\n",s[i].description);
        printf("Service cost : %.2f\n",s[i].serviceCost);
        printf("\n");
    }
```

```
}

void main()
{
    int n;
    char vid[15];
    printf("Enter number of vehicles\n");
    scanf("%d",&n);
    struct service *s=(struct service *)malloc(n*sizeof(struct service));
    for(int i=0;i<n;i++)
    {
        printf("Enter service id\n");
        scanf("%s",s[i].serviceID);
        printf("Enter vehicle id\n");
        scanf("%s",s[i].vehicleID);
        printf("Enter service date\n");
        scanf("%s",s[i].serviceDate);
        printf("Enter description\n");
        scanf("%s",s[i].description);
        printf("Enter service cost\n");
        scanf("%f",&s[i].serviceCost);

    }
    printf("Enter vehicle id to search\n");
    scanf("%s",vid);
    display_vid(s,vid,n);
    printf("Enter vehicle id to search\n");
    scanf("%s",vid);
    Total(s,vid,n);
    printf("\n");
    printf("sorted service record\n");
    sort(s,n);
}
```

```
Enter vehicle id to search
v1
Vehicle ID : v1
Service ID : s1
Service date : 2005-08-12
Description : oilchange
Service cost : 500.00
Vehicle ID : v1
Service ID : s3
Service date : 2006-08-14
Description : oilchange
Service cost : 500.00
Enter vehicle id to search
v1
Service cost of vehicle v1 is 1000.00

sorted service record
Vehicle ID : v1
Service ID : s1
Service date : 2005-08-12
Description : oilchange
Service cost : 500.00

Vehicle ID : v2
Service ID : s2
Service date : 2005-09-19
Description : breakchange
Service cost : 451.00

Vehicle ID : v1
Service ID : s3
Service date : 2006-08-14
Description : oilchange
Service cost : 500.00
```

```
/*Define a structure Player with the following members:
char name[50]
int age
char team[30]
```

```c
int matchesPlayed
int totalRuns
int totalWickets
Functions to:
Add a new player to the system.
Update a player’s statistics after a match.
Display the details of players from a specific team.
Find the player with the highest runs and the player with the most
wickets.
Use dynamic memory allocation to store player data in an array and expand
it as needed.*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct player
{
    char name[50];
int age;
char team[30];
int matchesPlayed;
int totalRuns;
int totalWickets;
};
void add(struct player *p)
{
    printf("Enter name of player\n");
    scanf("%s",p->name);
    printf("Enter the age\n");
    scanf("%d",&p->age);
    printf("Enter matches played\n");
    scanf("%d",&p->matchesPlayed);
    printf("Enter total runs\n");
    scanf("%d",&p->totalRuns);
    printf("Enter total wickets\n");
    scanf("%d",&p->totalWickets);

}
void update(struct player *p)
{
    p->matchesPlayed++;
```

```
    printf("Enter new total runs of %s\n",p->name);
    scanf("%d",&p->totalRuns);
    printf("Enter total wickets\n");
    scanf("%d",&p->totalWickets);

}
void  display(struct player *p,char t[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(p[i].team,t)==0)
        {
            printf("Name : %s\n",p[i].name);
        }
    }
}
void high(struct player *p,int n)
{
    int w=0,r=0;
    for(int i=0;i<n;i++)
    {
        if(w<p[i].totalWickets)
        w=p[i].totalWickets;
        if(r<p[i].totalRuns)
        r=p[i].totalRuns;
    }
    for (int i = 0; i < n; i++)
    {
        if(w==p[i].totalWickets)
        printf("Player with highest wicket is %s",p[i].name);
        if(r==p[i].totalRuns)
        printf("Player with highest runs %s",p[i].name);
    }

}
void main()
{
    int n;
    char t[30];
    printf("Enter number of players\n");
```

```
    scanf("%d",&n);
    struct player *p =(struct player *)malloc(n*sizeof(struct player));
    for(int i=0;i<n;i++)
    add(&p[i]);

    printf("\nPOST MATCH UPDATES\n");
    for(int i=0;i<n;i++)
    update(&p[i]);

    printf("\n");
    display(p,t,n);

    printf("\n");
    high(p,n);
    printf("Enter new size \n");
    scanf("%d",&n);
    p=realloc(p,n);
}
```

```
Enter the age
13
Enter matches played
6
Enter total runs
180
Enter total wickets
7
Enter name of player
p3
Enter the age
15
Enter matches played
8
Enter total runs
500
Enter total wickets
1

POST MATCH UPDATES
Enter new total runs of p1
400
Enter total wickets
5
Enter new total runs of p2
240
Enter total wickets
11
Enter new total runs of p3
649
Enter total wickets
2


Player with highest wicket is p2Player with highest runs p3Enter new size
```

```
/*Create a structure Match with members:
char team1[30]
char team2[30]
char date[11] (format: YYYY-MM-DD)
char venue[50]
Functions to:
Schedule a new match between two teams.
```

```c
/*Display all scheduled matches.
Search for matches scheduled on a specific date.
Cancel a match by specifying both team names and the date.
Ensure that the match schedule is stored in an array, with the ability to
dynamically adjust its size*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct match
{
    char team1[30];
char team2[30];
char date[11]; //(format: YYYY-MM-DD)//
char venue[50];
};
void setup(struct match *m,int n)
{

    for(int i=0;i<n;i++)
    {
        printf("Details of match %d\n",i+1);
        printf("Enter team1\n");
        scanf("%s",m[i].team1);
        printf("Enter team2\n");
        scanf("%s",m[i].team2);
        printf("Enter date\n");
        scanf("%s",m[i].date);
        printf("Enter the venue\n");
        scanf("%s",m[i].venue);
    }
}
void display(struct match *m,int n)
{
    printf("\n");
    for(int i=0;i<n;i++)
    {
        printf("MATCH %d\n",i+1);
        printf("Team %s V/S Team %s\n",m[i].team1,m[i].team2);
        printf("Date : %s\n",m[i].date);
        printf("Venue : %s\n",m[i].venue);
```

```
    }
}
void search(struct match *m,int n,char d[])
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(m[i].date,d)==0)
        {
            printf("Team %s V/S Team %s\n",m[i].team1,m[i].team2);
            printf("Date : %s\n",m[i].date);
            printf("Venue : %s\n",m[i].venue);
        }
    }
}
void cancel(struct match *m,char t1[],char t2[],char d[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(m[i].team1,t1)==0 && strcmp(m[i].team2,t2)==0 &&
strcmp(m[i].date,d)==0)
        {
            strcpy(m[i].date,"\0");
            strcpy(m[i].team1,"\0");
            strcpy(m[i].team2,"\0");
            strcpy(m[i].venue,"\0");
            printf("Match %d cancelled\n",i+1);
        }
    }
}
void main()
{
    int n,count=0;
    char d[11],t1[30],t2[30];
    printf("Enter the number of matches\n");
    scanf("%d",&n);
    struct match * m=(struct match *)malloc(n*sizeof(struct match));
    printf("Setup matches\n");
    setup(m,n);
    display(m,n);
    printf("Enter date to search\n");
```

```
    scanf("%s",d);
    search(m,n,d);
    printf("Enter the name of both teams and date to cancel\n");
    scanf("%s",t1);
    scanf("%s",t2);
    scanf("%s",d);
    cancel(m,t1,t2,d,n);
}
```

```
Aluva
Details of match 3
Enter team1
t1
Enter team2
t3
Enter date
2025-01-22
Enter the venue
Kochi

MATCH 1
Team t1 V/S Team t2
Date : 2025-01-10
Venue : kochi
MATCH 2
Team t2 V/S Team t3
Date : 2025-01-12
Venue : Aluva
MATCH 3
Team t1 V/S Team t3
Date : 2025-01-22
Venue : Kochi
Enter date to search
2025-01-10
Team t1 V/S Team t2
Date : 2025-01-10
Venue : kochi
Enter the name of both teams and date to cancel
t2 t3 2025-01-12
Match 2 cancelled
PS D:\projects\quest\C>
```

```
/*Define a structure CountryMedalTally with members:
char country[30]
int gold
int silver
int bronze
Functions to:
```

```c
/*Add a new country's medal tally.
Update the medal count for a country.
Display the medal tally for all countries.
Find and display the country with the highest number of gold medals.
Use an array to store the medal tally, and resize the array dynamically as
new countries are added*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct medal
{char country[30];
int gold;
int silver;
int bronze;

};
void add(struct medal *m,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter name of country\n");
        scanf("%s",&m[i].country);
        printf("Enter number of gold\n");
        scanf("%d",&m[i].gold);
        printf("Enter number of silver\n");
        scanf("%d",&m[i].silver);
        printf("Enter number of bronze\n");
        scanf("%d",&m[i].bronze);
    }
}
void update(struct medal *m,char c[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(m[i].country,c)==0)
        {
            printf("Enter updated counts for |Gold|Silver|Bronze\n");
            scanf("%d %d %d",&m[i].gold,&m[i].silver,&m[i].bronze);
        }
    }
```

```
}
void display(struct medal *m,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Country : %s\n",m[i].country);
        printf("Gold medals : %d\n",m[i].gold);
        printf("Silver medal : %d\n",m[i].silver);
        printf("Bronze medals : %d\n",m[i].bronze);
    }

}
void max(struct medal *m,int n)
{
    int max=0;
    for(int i=0;i<n;i++)
    {
        if(max<m[i].gold)
        max=m[i].gold;
    }
    for(int i=0;i<n;i++)
    {
        if(max==m[i].gold)
        printf("The country with max number of gold medals is
%s\n",m[i].country);
    }
}
void main()
{
    int n;
    char c[30];
    printf("Enter the number of countries\n");
    scanf("%d",&n);
    struct medal *m=(struct medal *)malloc(sizeof(struct medal));
    add(m,n);
    printf("Enter name of country to update medal count\n");
    scanf("%s",c);
    update(m,c,n);
    display(m,n);
    max(m,n);
```

```
}
```

```
Enter name of country to update medal count
India
Enter updated counts for |Gold|Silver|Bronze
14 6 7
Country : India
Gold medals : 14
Silver medal : 6
Bronze medals : 7
Country : China
Gold medals : 5
Silver medal : 6
Bronze medals : 7
Country : USA
Gold medals : 8
Silver medal : 10
Bronze medals : 11
The country with max number of gold medals is India
PS D:\projects\quest\C>
```

```c
/*Create a structure Athlete with fields:
char athleteID[10]
char name[50]
char sport[30]
float personalBest
float lastPerformance
Functions to:
Add a new athlete to the system.
Update an athlete's last performance.
Display all athletes in a specific sport.
Identify and display athletes who have set a new personal best in their
last performance.
Utilize dynamic memory allocation to manage athlete data in an expandable
array*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct athlete
```

```
{
    char athleteID[10];
char name[50];
char sport[30];
float personalBest;
float lastPerformance;
};
void add(struct athlete *a,int n)
{
    for(int  i=0;i<n;i++)
    {
        printf("Enter athlete id\n");
        scanf("%s",a[i].athleteID);
        printf("Enter name\n");
        scanf("%s",a[i].name);
        printf("Enter sports\n");
        scanf("%s",a[i].sport);
        printf("Personal best \n");
        scanf("%f",&a[i].personalBest);
        printf("Last performance\n");
        scanf("%f",&a[i].lastPerformance);
    }
}
void update(struct athlete *a,char nam[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(a[i].name,nam)==0)
        {
            printf("Update last performance of %s : ",a[i].name);
            scanf("%f",&a[i].lastPerformance);
        }
    }
}
void specific(struct athlete *a,char s[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(s,a[i].sport)==0)
        {
```

```
            printf("Athlete name  : %s",a[i].name);
        }
    }
}
void personal(struct athlete *a,int n )
{
    printf("\nAtheletes se a new personal best in their last
performance\n");
    for(int i=0;i<n;i++)
    {
        if(a[i].lastPerformance>a[i].personalBest)
        {
            a[i].personalBest=a[i].lastPerformance;
            printf("Name : %s",a[i].name);
        }
    }
}
void main()
{
int n;
char nam[50],s[30];
printf("Enter the number of athelets\n");
scanf("%d",&n);
struct athlete *a=(struct athlete *)malloc(n*sizeof(struct athlete));
add(a,n);
printf("Enter the name of athelete to update\n");
scanf("%s",nam);
update(a,nam,n);
printf("Enter name of sport\n");
scanf("%s",s);
specific(a,s,n);
personal(a,n);

}
```

```
Enter the name of athelete to update
rohit
Update last performance of rohit : 14
Enter name of sport
100meters
Athlete name  : rahulAthlete name  : rayan
Atheletes se a new personal best in their last performance
Name : rohit
PS D:\projects\guest\C>
```

```c
/*Define a structure Equipment with members:
char equipmentID[10]
char name[30]
char category[20] (e.g., balls, rackets)
int quantity
float pricePerUnit
Functions to:
Add new equipment to the inventory.
Update the quantity of existing equipment.
Display all equipment in a specific category.
Calculate the total value of equipment in the inventory.
Store the inventory data in a dynamically allocated array and ensure
proper resizing when needed*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct equipment
{
    char equipmentID[10];
char name[30];
char category[20]; //(e.g., balls, rackets)//
int quantity;
float pricePerUnit;
};
void add(struct equipment *e,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter equipment id : ");
        scanf("%s",e[i].equipmentID);
```

```
        printf("Enter name : ");
        scanf("%s",e[i].name);
        printf("Enter category : ");
        scanf("%s",e[i].category);
        printf("Enter quantity : ");
        scanf("%d",&e[i].quantity);
        printf("Enter price per unit : ");
        scanf("%f",&e[i].pricePerUnit);
    }
}
void update(struct equipment *e,char nam[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(e[i].name,nam)==0)
        {
            printf("Enter updated quantity : ");
            scanf("%d",&e[i].quantity);
        }
    }
}
void display(struct equipment *e,char c[],int n)
{
    printf("Equipments under %s category\n",c);
    for(int i=0;i<n;i++)
    {
        if(strcmp(e[i].category,c)==0)
        {
        printf("Enter equipment id : %s\n",e[i].equipmentID);
        printf("Enter name : %s\n",e[i].name);
        printf("Enter quantity : %d\n",e[i].quantity);
        printf("Enter price per unit : %.2f\n",e[i].pricePerUnit);
        }
    }
}
void total(struct equipment *e,int n)
{
    int sum=0;
    for(int i=0;i<n;i++)
    {
```

```
        sum=sum+e[i].quantity*e[i].pricePerUnit;
    }
    printf("Total stock value is %d",sum);
}
void main()
{
    int n;
    char nam[30],c[20];
    printf("Enter the number of equipments\n");
    scanf("%d",&n);
    struct equipment *e=(struct equipment *)malloc(n*sizeof(struct
equipment));
    add(e,n);
    printf("Enter the name of equipment to search : ");
    scanf("%s",nam);
    update(e,nam,n);
    printf("Enter the category to search : ");
    scanf("%s",c);
    display(e,c,n);
    total(e,n);
}
```

```
PS D:\projects\quest\C> cd "d:\projects\quest\
Enter the number of equipments
3
Enter equipment id : e1
Enter name : racket
Enter category : bat
Enter quantity : 10
Enter price per unit : 10
Enter equipment id : e2
Enter name : football
Enter category : ball
Enter quantity : 20
Enter price per unit : 30
Enter equipment id : e3
Enter name : basketball
Enter category : ball
Enter quantity : 40
Enter price per unit : 12
Enter the name of equipment to search : racket
Enter updated quantity : 50
Enter the category to search : ball
Equipments under ball category
Enter equipment id : e2
Enter name : football
Enter quantity : 20
Enter price per unit : 30.00
Enter equipment id : e3
Enter name : basketball
Enter quantity : 40
Enter price per unit : 12.00
Total stock value is 1580
PS D:\projects\quest\C>
```

```
/*Define a structure ResearchPaper with the following members:
char title[100]
char author[50]
char journal[50]
int year
char DOI[30]
```

```c
/*Functions to:
Add a new research paper to the database.
Update the details of an existing paper using its DOI.
Display all papers published in a specific journal.
Find and display the most recent papers published by a specific author.
Use dynamic memory allocation to store and manage the research papers in
an array, resizing it as needed.*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct paper
{
   char title[100];
char author[50];
char journal[50];
int year;
char DOI[30];
};
void add(struct paper *p,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter title : ");
        scanf("%s",p[i].title);
        printf("Enter author : ");
        scanf("%s",p[i].author);
        printf("Enter journal : ");
        scanf("%s",p[i].journal);
        printf("Enter year :");
        scanf("%d",&p[i].year);
        printf("Enter DOI : ");
        scanf("%s",p[i].DOI);
    }
}
void update(struct paper * p, char d[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(p[i].DOI,d)==0)
        {
```

```
            printf("Update title : ");
            scanf("%s",p[i].title);
            printf("Update author : ");
            scanf("%s",p[i].author);
            printf("Update journal : ");
            scanf("%s",p[i].journal);
            printf("Update year : ");
            scanf("%d",&p[i].year);
        }
    }
}
void display(struct paper *p,char j[],int n)
{
    printf("\n Papers pblished in %s\n",j);
    for(int i=0;i<n;i++)
    {
        if(strcmp(p[i].journal,j)==0)
        {
            printf("Author : %s\n",p[i].author);
            printf("Title : %s\n",p[i].title);
            printf("Journal : %s\n",p[i].journal);
            printf("DOI : %s\n",p[i].DOI);
            printf("Year : %d\n",p[i].year);
            printf("\n");
        }
    }
}
void auth(struct paper *p,char a[],int n)
{
    int  max=0;
    for(int i=0;i<n;i++)
    {
        if(strcmp(p[i].author,a)==0 && max < p[i].year)
        max=p[i].year;
    }
    for(int i=0;i<n;i++)
    {
        if(strcmp(p[i].author,a)==0 && max==p[i].year)
        {
            printf("Author : %s\n",p[i].author);
```

```
            printf("Title : %s\n",p[i].title);
            printf("Journal : %s\n",p[i].journal);
            printf("DOI : %s\n",p[i].DOI);
            printf("Year : %d\n",p[i].year);
        }
    }

}
void main()
{
    int n;
    char d[30],j[50],a[50];
    printf("Enter the number of papers\n");
    scanf("%d",&n);
    struct paper * p=(struct paper *)malloc(n*sizeof(struct paper));
    add(p,n);
    printf("Enter the DOI to update\n");
    scanf("%s",d);
    update(p,d,n);
    printf("Enter the name of journal\n");
    scanf("%s",j);
    display(p,j,n);
    printf("Enter name of author to search\n");
    scanf("%s",a);
    auth(p,a,n);
}
```

```
Enter author : a2
Enter journal : j2
Enter year :2009
Enter DOI : d2
Enter title : t3
Enter author : a2
Enter journal : j1
Enter year :2012
Enter DOI : d3
Enter the DOI to update
d1
Update title : t4
Update author : a3
Update journal : j3
Update year : 2016
Enter the name of journal
j2

 Papers pblished in j2
Author : a2
Title : t2
Journal : j2
DOI : d2
Year : 2009

Enter name of author to search
a2
Author : a2
Title : t3
Journal : j1
DOI : d3
Year : 2012
PS D:\projects\quest\C>
```

```
/*Create a structure Experiment with members:
char experimentID[10]
char researcher[50]
char startDate[11] (format: YYYY-MM-DD)
char endDate[11]
```

```c
float results[10] (store up to 10 result readings)
Functions to:
Log a new experiment.
Update the result readings of an experiment.
Display all experiments conducted by a specific researcher.
Calculate and display the average result for a specific experiment.
Use a dynamically allocated array for storing experiments and manage
resizing as more data is logged.*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct exp
{
   char experimentID[10];
char researcher[50];
char startDate[11]; //(format: YYYY-MM-DD)//
char endDate[11];
float results[10];
};
void add(struct exp *e,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter exp id : ");
        scanf("%s",e[i].experimentID);
        printf("Enter researcher name : ");
        scanf("%s",e[i].researcher);
        printf("Enter start date : ");
        scanf("%s",e[i].startDate);
        printf("Enter end date : ");
        scanf("%s",e[i].endDate);
        printf("Enter the results : ");
        for(int j=0;j<10;j++)
        scanf("%f",&e[i].results[j]);
    }
}
void update(struct exp *e,char eid[],int n)
{
    for(int i=0;i<n;i++)
    {
```

```
        if(strcmp(e[i].experimentID,eid)==0)
        {
            printf("Update results : ");
            for(int j=0;j<10;j++)
            {
                scanf("%f",&e[i].results[j]);
            }
        }
    }

}
void research(struct exp *e,char r[],int n)
{
    printf("Experiments done by %s",r);
    for(int i=0;i<n;i++)
    {
        if(strcmp(e[i].researcher,r)==0)
        {
            printf("Experiment id : %s\n" ,e[i].experimentID);
        }
    }
}
void average(struct exp *e,char eid[],int n)
{
    float sum=0;
    for(int i=0;i<n;i++)
    {
        if(strcmp(e[i].experimentID,eid)==0)
        {
            for(int j=0;j<10;j++)
            sum+=e[i].results[j];
            printf("The average of results of experiment id %s :
%f\n",e[i].experimentID,sum/10);
        }
    }

}
void main()
{
    int n;
```

```
     char eid[10],r[50];
    printf("Enter the number of experiments\n");
    scanf("%d",&n);
    struct exp *e =(struct exp * )malloc(n*sizeof(struct exp));
    add(e,n);
    printf("Enter experiment id to update\n");
    scanf("%s",eid);
    update(e,eid,n);
    printf("Enter name of researcher to search\n");
    scanf("%s",r);
    research(e,r,n);
    printf("Enter experiment id to find average of results\n");
    scanf("%s",eid);
    average(e,eid,n);
}
```

```
PS D:\projects\quest\C> cd "d:\projects\quest\C\" ; if ($?) { gc
Enter the number of experiments
3
Enter exp id : e1
Enter researcher name : r1
Enter start date : 2025-01-01
Enter end date : 2025-01-09
Enter the results : 1 2 3 4 5 6 7 8 9 10
Enter exp id : e2
Enter researcher name : r2
Enter start date : 2024-10-23
Enter end date : 2025-01-01
Enter the results : 11 12 13 14 15 16 17 18 19 20
Enter exp id : e3
Enter researcher name : r2
Enter start date : 2022-11-15
Enter end date : 2023-03-25
Enter the results : 2 4 8 10 12 14 16 18 20 22
Enter experiment id to update
e1
Update results : 21 22 23 24 25 26 27 28 29 30
Enter name of researcher to search
r2
Experiments done by r2Experiment id : e2
Experiment id : e3
Enter experiment id to find average of results
e3
The average of results of experiment id e3 : 12.600000
PS D:\projects\quest\C>
```

```
/*Define a structure GrantApplication with the following members:
char applicationID[10]
char applicantName[50]
char projectTitle[100]
float requestedAmount
char status[20] (e.g., Submitted, Approved, Rejected)
Functions to:
```

```c
/*Add a new grant application.
Update the status of an application.
Display all applications requesting an amount greater than a specified
value.
Find and display applications that are currently "Approved."
Store the grant applications in a dynamically allocated array, resizing it
as necessary.*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct application
{
    char applicationID[10];
char applicantName[50];
char projectTitle[100];
float requestedAmount;
char status[20] ;//(e.g., Submitted, Approved, Rejected)//
};
void add(struct application *a,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter application id : ");
        scanf("%s",a[i].applicationID);
        printf("Enter application name : ");
        scanf("%s",a[i].applicantName);
        printf("Enter projec title : ");
        scanf("%s",a[i].projectTitle);
        printf("Enter requested amount : ");
        scanf("%f",&a[i].requestedAmount);
        printf("Enter the status : ");
        scanf("%s",a[i].status);
    }
}
void display(struct application *a,float num,int n)
{
    printf("PRojects with budjet above %f\n",num);
    for(int i=0;i<n;i++)
    {
        if(a[i].requestedAmount>num)
```

```
        {

            printf("Application id : %s\n",a[i].applicationID);
            printf("Application name : %s\n",a[i].applicantName);
            printf("Project title : %s\n",a[i].projectTitle);
            printf("Requested amount : %.2f\n",a[i].requestedAmount);
            printf("Status : %s\n",a[i].status);
            printf("\n");
        }
    }
}
void approved(struct application *a,int n)
{
    printf("Approved applications\n");
    for(int i=0;i<n;i++)
    {
        if(strcmp(a[i].status,"Approved")==0)
        {
            printf("Application id : %s\n",a[i].applicationID);
            printf("Application name : %s\n",a[i].applicantName);
            printf("Project title : %s\n",a[i].projectTitle);
            printf("Requested amount : %.2f\n",a[i].requestedAmount);
            printf("\n");

        }
    }
}
void main()
{
    int n;
    float num;
    printf("Enter number of applications : ");
    scanf("%d",&n);
    struct application * a=(struct application *)malloc(n*sizeof(struct
application));
    add(a,n);
    printf("Enter threshold amiunt : ");
    scanf("%f",&num);
    display(a,num,n);
    approved(a,n);
```

```
}
```

```
PROBLEMS     OUTPUT     DEBUG CONSOLE

Enter application id : e2
Enter application name : a2
Enter projec title : t2
Enter requested amount : 4000
Enter the status : Approved
Enter application id : e3
Enter application name : a3
Enter projec title : t3
Enter requested amount : 7000
Enter the status : Approved
Enter threshold amount : 3500

Projects with budjet above 3500.
Application id : e2
Application name : a2
Project title : t2
Requested amount : 4000.00
Status : Approved

Application id : e3
Application name : a3
Project title : t3
Requested amount : 7000.00
Status : Approved


Approved applications
Application id : e2
Application name : a2
Project title : t2
Requested amount : 4000.00

Application id : e3
Application name : a3
Project title : t3
Requested amount : 7000.00
```

```c
/*Create a structure Collaborator with members:
char collaboratorID[10]
char name[50]
char institution[50]
char expertiseArea[30]
int numberOfProjects
Functions to:
Add a new collaborator to the database.
Update the number of projects a collaborator is involved in.
Display all collaborators from a specific institution.
Find collaborators with expertise in a given area.
Use dynamic memory allocation to manage the list of collaborators,
allowing for expansion as more are added*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct collab
{
char collaboratorID[10];
char name[50];
char institution[50];
char expertiseArea[30];
int numberOfProjects;
};
void add(struct collab *c,int  n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter collaboratorID : ");
        scanf("%s",c[i].collaboratorID);
        printf("Enter name : ");
        scanf("%s",c[i].name);
        printf("Enter institution : ");
        scanf("%s",c[i].institution);
        printf("Enter expertiseArea : ");
        scanf("%s",c[i].expertiseArea);
        printf("Enter number of projects : ");
        scanf("%d",&c[i].numberOfProjects);
```

```
    }
}
void update(struct collab *c,char cid[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(c[i].collaboratorID,cid)==0)
        {
            printf("Enter updated value : ");
            scanf("%d",&c[i].numberOfProjects);
        }
    }
}
void display(struct collab *c,char ins[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(c[i].institution,ins)==0)
        {
            printf("Collaborator id : %s\n",c[i].collaboratorID);
            printf("Collaborator name : %s\n",c[i].name);
            printf("\n");

        }
    }
}
void main()
{
    int n;
    char cid[10],ins[50];
    printf("Enter number of collaborators : ");
    scanf("%d",&n);
    struct collab * c=(struct collab *)malloc(n*sizeof(struct collab));
    add(c,n);
    printf("Enter Collaborator ID to update number of projects involves :
");
    scanf("%s",cid);
    update(c,cid,n);
    printf("Enter institution to search : ");
    scanf("%s",ins);
```

```
    display(c,ins,n);
}
PS D:\projects\quest\c> cd "d:\projects\quest\c\" ; if ($?) { gcc tempCodeRunne
Enter number of collaborators : 3
Enter collaboratorID : c1
Enter name : n1
Enter institution : i1
Enter expertiseArea : garage
Enter number of projects : 5
Enter collaboratorID : c2
Enter name : n2
Enter institution : i2
Enter expertiseArea : pond
Enter number of projects : 6
Enter collaboratorID : c3
Enter name : n3
Enter institution : i3
Enter expertiseArea : pond
Enter number of projects : 4
Enter Collaborator ID to update number of projects involves : c3
Enter updated value : 7
Enter institution to search : i2
Collaborator id : c2
Collaborator name : n2
```

```
/*Define a structure ConferenceSubmission with the following:
char submissionID[10]
char authorName[50]
char paperTitle[100]
char conferenceName[50]
char submissionDate[11]
char status[20] (e.g., Pending, Accepted, Rejected)
Functions to:
Add a new conference submission.
Update the status of a submission.
Display all submissions to a specific conference.
Find and display submissions by a specific author.
Store the conference submissions in a dynamically allocated array,
resizing the array as needed when more submissions are added.*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct conference
```

```
{
char submissionID[10];
char authorName[50];
char paperTitle[100];
char conferenceName[50];
char submissionDate[11];
char status[20];
};
void add(struct conference *c,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Enter submission id : ");
        scanf("%s",c[i].submissionID);
        printf("Enter author name : ");
        scanf("%s",c[i].authorName);
        printf("Enter paper title : ");
        scanf("%s",c[i].paperTitle);
        printf("Enter conference name : ");
        scanf("%s",c[i].conferenceName);
        printf("Enter submission date : ");
        scanf("%s",c[i].submissionDate);
        printf("Enter status : ");
        scanf("%s",c[i].status);
    }
}
void update(struct conference *c,char sid[],int n)
{
    for(int i=0;i<n;i++)
    {
        if(strcmp(c[i].submissionID,sid)==0)
        {
            printf("Enter updated value : ");
            scanf("%s",c[i].status);
        }
    }
}
void display(struct conference *c,char cn[],int n)
{
    for(int i=0;i<n;i++)
```

```
    {
        if(strcmp(c[i].conferenceName,cn)==0)
        {
             printf("Submission id : %s\n",c[i].submissionID);
            printf("Author name : %s\n",c[i].authorName);
            printf("Paper title : %s\n",c[i].paperTitle);
            printf("Conference name : %s\n",c[i].conferenceName);
            printf("Submission date : %s\n",c[i].submissionDate);
            printf("Enter status : %s\n",c[i].status);
        }
    }
}
void Author(struct conference *c,char a[],int n)
{
    printf("Submissions made by %s\n",a);
    for(int i=0;i<n;i++)
    {
        if(strcmp(c[i].authorName,a)==0)
        {
            printf("Submission id : %s\n",c[i].submissionID);
            printf("Paper title : %s\n",c[i].paperTitle);
            printf("Conference name : %s\n",c[i].conferenceName);
            printf("Submission date : %s\n",c[i].submissionDate);
            printf("Enter status : %s\n",c[i].status);
        }
    }
}

void main()
{
    int n;
    char sid[10],cn[50],a[50];
    printf("Enter the number of conferences : ");
    scanf("%d",&n);
    struct conference *c= (struct conference *)malloc(sizeof(struct
conference));
    add(c,n);
    printf("Enter the submission id : ");
    scanf("%s",sid);
    update(c,sid,n);
```

```c
    printf("Enter the conference name to search : ");
    scanf("%s",cn);
    display(c,cn,n);
    printf("Enter Author name to search : ");
    scanf("%s",a);
    Author(c,a,n);


}
```

```
Enter submission id : s3
Enter author name : n3
Enter paper title : t3
Enter conference name : cn2
Enter submission date : 2024-09-12
Enter status : Approved
Enter the submission id : s1
Enter updated value : Approved
Enter the conference name to search : cn2
Submission id : s2
Author name : n2
Paper title : t2
Conference name : cn2
Submission date : 2024-10-10
Enter status : Approved
Submission id : s3
Author name : n3
Paper title : t3
Conference name : cn2
Submission date : 2024-09-12
Enter status : Approved
Enter Author name to search : n1
Submissions made by n1
Submission id : s1
Paper title : t1
Conference name : cn1
Submission date : 20025-01-09Approved
Enter status : Approved
PS D:\projects\quest\C>
```