Day 12 programs

1.Student Information:

Define a structure to store student information, including name, roll number, and marks in three subjects.

Write a program to input data for 5 students and display the details along with their average marks.

```
#include <stdio.h>


struct Student {
    char name[50];
    int rollNo;
    float marks[3];
};


int main() {
    struct Student students[5];
    int i;


    for (i = 0; i < 5; i++) {
        printf("Enter name, roll number, and marks for student %d:\n", i + 1);
        scanf("%s %d %f %f %f", students[i].name, &students[i].rollNo,
              &students[i].marks[0], &students[i].marks[1], &students[i].marks[2]);
    }


    printf("\nStudent Details:\n");
    for (i = 0; i < 5; i++) {
        float avg = (students[i].marks[0] + students[i].marks[1] + students[i].marks[2]) /
3;
        printf("Name: %s, Roll No: %d, Average Marks: %.2f\n",
```

```
            students[i].name, students[i].rollNo, avg);
    }

    return 0;
}
```

2.Employee Details:

Create a structure to store employee details like name, ID, salary, and department.

Write a function to display the details of employees whose salary is above a certain threshold.

```
#include <stdio.h>

struct Employee {
    char name[50];
    int id;
    float salary;
    char department[30];
};

void displayHighEarners(struct Employee employees[], int n, float threshold) {
    printf("\nEmployees earning above %.2f:\n", threshold);
    for (int i = 0; i < n; i++) {
        if (employees[i].salary > threshold) {
            printf("Name: %s, ID: %d, Salary: %.2f, Department: %s\n",
                employees[i].name, employees[i].id, employees[i].salary,
employees[i].department);
        }
    }
}
```

```
int main() {
    struct Employee employees[3];
    for (int i = 0; i < 3; i++) {
        printf("Enter details for employee %d:\n", i + 1);
        printf("Name: ");
        scanf("%s", employees[i].name);
        printf("ID: ");
        scanf("%d", &employees[i].id);
        printf("Salary: ");
        scanf("%f", &employees[i].salary);
        printf("Department: ");
        scanf("%s", employees[i].department);
    }

    float threshold;
    printf("Enter salary threshold: ");
    scanf("%f", &threshold);
    displayHighEarners(employees, 3, threshold);

    return 0;
}
```

3.Book Store Inventory:

Define a structure to represent a book with fields for title, author, ISBN, and price. Write a program to manage an inventory of books and allow searching by title.

```
#include <stdio.h>
#include <string.h>
```

```
struct Book {
    char title[50];
    char author[50];
    char isbn[20];
    float price;
};

void searchByTitle(struct Book books[], int n, char *title) {
    for (int i = 0; i < n; i++) {
        if (strcmp(books[i].title, title) == 0) {
            printf("Book Found: %s by %s (ISBN: %s, Price: %.2f)\n",
                   books[i].title, books[i].author, books[i].isbn, books[i].price);
            return;
        }
    }
    printf("Book not found.\n");
}

int main() {
    struct Book books[3];
    for (int i = 0; i < 3; i++) {
        printf("Enter details for book %d:\n", i + 1);
        printf("Title: ");
        scanf("%s", books[i].title);
        printf("Author: ");
        scanf("%s", books[i].author);
        printf("ISBN: ");
        scanf("%s", books[i].isbn);
        printf("Price: ");
```

```
        scanf("%f", &books[i].price);
    }

    char title[50];
    printf("Enter title to search: ");
    scanf("%s", title);
    searchByTitle(books, 3, title);

    return 0;
}
```

4.Date Validation:

Create a structure to represent a date with day, month, and year.

Write a function to validate if a given date is correct (consider leap years).

```
#include <stdio.h>

struct Date {
    int day, month, year;
};

int isLeapYear(int year) {
    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
}

int validateDate(struct Date date) {
    if (date.year < 0 || date.month < 1 || date.month > 12 || date.day < 1)
        return 0;
```

```
    int daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    if (isLeapYear(date.year))
        daysInMonth[1] = 29;

    return date.day <= daysInMonth[date.month - 1];
}

int main() {
    struct Date date;
    printf("Enter date (dd mm yyyy): ");
    scanf("%d %d %d", &date.day, &date.month, &date.year);

    if (validateDate(date)) {
        printf("Valid date.\n");
    } else {
        printf("Invalid date.\n");
    }

    return 0;
}
```

5.Complex Numbers:

Define a structure to represent a complex number with real and imaginary parts.

Implement functions to add, subtract, and multiply two complex numbers.

```
#include <stdio.h>

struct Complex {
    float real;
    float imag;
```

```
};

int main() {
    struct Complex num1, num2, result;

    printf("Enter first complex number (real and imaginary): ");
    scanf("%f %f", &num1.real, &num1.imag);

    printf("Enter second complex number (real and imaginary): ");
    scanf("%f %f", &num2.real, &num2.imag);

    // Perform addition
    result.real = num1.real + num2.real;
    result.imag = num1.imag + num2.imag;
    printf("Addition: %.2f + %.2fi\n", result.real, result.imag);

    // Perform subtraction
    result.real = num1.real - num2.real;
    result.imag = num1.imag - num2.imag;
    printf("Subtraction: %.2f + %.2fi\n", result.real, result.imag);

    // Perform multiplication
    result.real = num1.real * num2.real - num1.imag * num2.imag;
    result.imag = num1.real * num2.imag + num1.imag * num2.real;
    printf("Multiplication: %.2f + %.2fi\n", result.real, result.imag);

    return 0;
}
```

6.Bank Account:

Design a structure to store information about a bank account, including account number, account holder name, and balance.

Write a function to deposit and withdraw money, and display the updated balance.

```
#include <stdio.h>

struct BankAccount {
    int accountNumber;
    char holderName[50];
    float balance;
};

int main() {
    struct BankAccount account = {98765, "Amal Neerath", 2500.0};
    int choice;
    float amount;

    do {
        printf("\nAccount No: %d, Holder: %s, Balance: %.2f\n",
               account.accountNumber, account.holderName, account.balance);
        printf("1. Deposit\n2. Withdraw\n3. Exit\nChoose an option: ");
        scanf("%d", &choice);

        if (choice == 1) {
            printf("Enter deposit amount: ");
            scanf("%f", &amount);
            account.balance += amount;
            printf("Deposited %.2f. New Balance: %.2f\n", amount, account.balance);
        } else if (choice == 2) {
```

```
            printf("Enter withdrawal amount: ");
            scanf("%f", &amount);
            if (amount > account.balance) {
                printf("Insufficient funds.\n");
            } else {
                account.balance -= amount;
                printf("Withdrew %.2f. New Balance: %.2f\n", amount, account.balance);
            }
        } else if (choice != 3) {
            printf("Invalid choice. Try again.\n");
        }
    } while (choice != 3);

    printf("Goodbye!\n");
    return 0;
}
```

7.Car Inventory System:

Create a structure for a car with fields like make, model, year, and price.

Write a program to store details of multiple cars and print cars within a specified price range.

```
#include <stdio.h>

struct Car {
    char make[50];
    char model[50];
    int year;
    float price;
};
```

```
void printCarsInRange(struct Car cars[], int n, float minPrice, float maxPrice) {
    printf("\nCars in price range %.2f - %.2f:\n", minPrice, maxPrice);
    for (int i = 0; i < n; i++) {
        if (cars[i].price >= minPrice && cars[i].price <= maxPrice) {
            printf("Make: %s, Model: %s, Year: %d, Price: %.2f\n",
                   cars[i].make, cars[i].model, cars[i].year, cars[i].price);
        }
    }
}

int main() {
    struct Car cars[3];
    for (int i = 0; i < 3; i++) {
        printf("Enter details for car %d:\n", i + 1);
        printf("Make: ");
        scanf("%s", cars[i].make);
        printf("Model: ");
        scanf("%s", cars[i].model);
        printf("Year: ");
        scanf("%d", &cars[i].year);
        printf("Price: ");
        scanf("%f", &cars[i].price);
    }

    float minPrice, maxPrice;
    printf("Enter minimum and maximum price: ");
    scanf("%f %f", &minPrice, &maxPrice);
    printCarsInRange(cars, 3, minPrice, maxPrice);
```

```
    return 0;
}
```

8.Library Management:

Define a structure for a library book with fields for title, author, publication year, and status (issued or available).

Write a function to issue and return books based on their status.

```
#include <stdio.h>

struct LibraryBook {
    char title[50];
    char author[50];
    int isIssued; // 0 for available, 1 for issued
};

int main() {
    struct LibraryBook book = {"The Great Gatsby", "F. Scott Fitzgerald", 0};
    int choice;

    do {
        printf("\nBook: %s by %s\nStatus: %s\n",
               book.title, book.author, book.isIssued ? "Issued" : "Available");
        printf("1. Issue Book\n2. Return Book\n3. Exit\nChoose an option: ");
        scanf("%d", &choice);

        if (choice == 1 && !book.isIssued) {
            book.isIssued = 1;
```

```
            printf("Book issued successfully.\n");
        } else if (choice == 2 && book.isIssued) {
            book.isIssued = 0;
            printf("Book returned successfully.\n");
        } else if (choice != 3) {
            printf(book.isIssued ? "Book already issued.\n" : "Book is already
available.\n");
        }
    } while (choice != 3);

    printf("Goodbye!\n");
    return 0;
}
```

9.Student Grades:

Create a structure to store a student's name, roll number, and an array of grades.

Write a program to calculate and display the highest, lowest, and average grade for each student.

```
#include <stdio.h>

struct Student {
    char name[50];
    int rollNo;
    float grades[5];
};

int main() {
    struct Student student = {"Alice", 101, {85, 90, 78, 92, 88}};
    float sum = 0, highest = student.grades[0], lowest = student.grades[0];
```

```
    for (int i = 0; i < 5; i++) {
        sum += student.grades[i];
        if (student.grades[i] > highest) highest = student.grades[i];
        if (student.grades[i] < lowest) lowest = student.grades[i];
    }

    printf("Student: %s, Roll No: %d\n", student.name, student.rollNo);
    printf("Highest Grade: %.2f, Lowest Grade: %.2f, Average Grade: %.2f\n",
           highest, lowest, sum / 5);

    return 0;
}
```

10.Product Catalog:

Define a structure to represent a product with fields for product ID, name, quantity, and price.

Write a program to update the quantity of products after a sale and calculate the total sales value.

```
#include <stdio.h>

struct Product {
    int id;
    char name[50];
    int quantity;
    float price;
};

int main() {
```

```
    struct Product product = {101, "Laptop", 50, 750.0};
    int sold;

    printf("Product: %s (ID: %d)\nStock: %d, Price: %.2f\n",
           product.name, product.id, product.quantity, product.price);

    printf("Enter quantity sold: ");
    scanf("%d", &sold);

    if (sold > product.quantity) {
        printf("Not enough stock available.\n");
    } else {
        product.quantity -= sold;
        printf("Updated stock: %d\n", product.quantity);
        printf("Total sale value: %.2f\n", sold * product.price);
    }

    return 0;
}
```

Additional Programs on structures

1.Point Distance Calculation:

Define a structure for a point in 2D space (x, y).

Write a function to calculate the distance between two points.

```
#include <stdio.h>
#include <math.h>

struct Point {
```

```
    int x, y;
};

float calculateDistance(struct Point p1, struct Point p2) {
    return sqrt(pow(p2.x - p1.x, 2) + pow(p2.y - p1.y, 2));
}

int main() {
    struct Point p1 = {0, 0}, p2 = {3, 4};
    printf("Distance: %.2f\n", calculateDistance(p1, p2));
    return 0;
}
```

2.Rectangle Properties:

Create a structure for a rectangle with length and width.

Write functions to calculate the area and perimeter of the rectangle.

```
#include <stdio.h>

struct Rectangle {
    int length, width;
};

int calculateArea(struct Rectangle r) {
    return r.length * r.width;
}

int calculatePerimeter(struct Rectangle r) {
    return 2 * (r.length + r.width);
```

```
}

int main() {
    struct Rectangle r = {5, 3};
    printf("Area: %d, Perimeter: %d\n", calculateArea(r), calculatePerimeter(r));
    return 0;
}
```

3.Movie Details:

Define a structure to store details of a movie, including title, director, release year, and rating.

Write a program to sort movies by their rating.

```
#include <stdio.h>

struct Movie {
    char title[50];
    char director[50];
    int year;
    float rating;
};

int main() {
    struct Movie movies[3] = {
        {"Movie1", "Director1", 2020, 8.2},
        {"Movie2", "Director2", 2018, 7.5},
        {"Movie3", "Director3", 2021, 9.0}
    };
```

```
    struct Movie temp;

    // Sorting movies by rating (bubble sort)
    for (int i = 0; i < 2; i++) {
        for (int j = i + 1; j < 3; j++) {
            if (movies[i].rating < movies[j].rating) {
                temp = movies[i];
                movies[i] = movies[j];
                movies[j] = temp;
            }
        }
    }

    // Display sorted movies
    printf("Movies sorted by rating:\n");
    for (int i = 0; i < 3; i++) {
        printf("Title: %s, Director: %s, Year: %d, Rating: %.1f\n",
               movies[i].title, movies[i].director, movies[i].year, movies[i].rating);
    }

    return 0;
}
```

4.Weather Report:

Create a structure to store daily weather data, including date, temperature, and humidity.

Write a program to find the day with the highest temperature.

```
#include <stdio.h>
```

```
struct Weather {
    int day;
    float temperature;
};

int main() {
    struct Weather weather[7] = {{1, 22.5}, {2, 25.0}, {3, 30.0}, {4, 28.5}, {5, 31.0}, {6, 29.5}, {7, 32.0}};
    int hottestDay = 0;

    for (int i = 1; i < 7; i++) {
        if (weather[i].temperature > weather[hottestDay].temperature) {
            hottestDay = i;
        }
    }

    printf("Hottest day: Day %d, Temperature: %.2f\n", weather[hottestDay].day, weather[hottestDay].temperature);
    return 0;
}
```

5.Fraction Arithmetic:

Define a structure for a fraction with numerator and denominator.

Write functions to add, subtract, multiply, and divide two fractions.

```
#include <stdio.h>

struct Fraction {
    int numerator, denominator;
```

```
};

struct Fraction add(struct Fraction a, struct Fraction b) {
    struct Fraction result;
    result.numerator = a.numerator * b.denominator + b.numerator * a.denominator;
    result.denominator = a.denominator * b.denominator;
    return result;
}

void printFraction(struct Fraction f) {
    printf("%d/%d\n", f.numerator, f.denominator);
}

int main() {
    struct Fraction f1 = {1, 2}, f2 = {1, 3};
    struct Fraction result = add(f1, f2);
    printFraction(result);
    return 0;
}
```

6.Laptop Inventory:

Create a structure to represent a laptop with fields for brand, model, processor, RAM, and price.

Write a program to list laptops within a specific price range.

```
#include <stdio.h>

struct Laptop {
    char brand[50];
```

```
    char model[50];
    char processor[50];
    int ram; // in GB
    float price;
};

int main() {
    struct Laptop laptops[3] = {
        {"Dell", "XPS 13", "Intel i7", 16, 1200.0},
        {"HP", "Pavilion", "Intel i5", 8, 800.0},
        {"Apple", "MacBook Pro", "M1", 8, 1500.0}
    };

    float minPrice, maxPrice;

    printf("Enter the price range (min max): ");
    scanf("%f %f", &minPrice, &maxPrice);

    printf("Laptops in the price range %.2f - %.2f:\n", minPrice, maxPrice);

    for (int i = 0; i < 3; i++) {
        if (laptops[i].price >= minPrice && laptops[i].price <= maxPrice) {
            printf("Brand: %s, Model: %s, Processor: %s, RAM: %dGB, Price: %.2f\n",
                laptops[i].brand, laptops[i].model, laptops[i].processor, laptops[i].ram,
laptops[i].price);
        }
    }

    return 0;
```

```
}
```

7.Student Attendance:

Define a structure to store attendance data, including student ID, total classes, and classes attended.

Write a program to calculate and display the attendance percentage for each student.

```
#include <stdio.h>

struct Student {
    int id, totalClasses, attendedClasses;
};

float calculateAttendance(struct Student s) {
    return (float) s.attendedClasses / s.totalClasses * 100;
}

int main() {
    struct Student student = {1, 50, 45};
    printf("Attendance: %.2f%%\n", calculateAttendance(student));
    return 0;
}
```

8.Flight Information:

Create a structure for a flight with fields for flight number, departure, destination, and duration.

Write a program to display flights that are less than a specified duration.

```c
#include <stdio.h>

struct Flight {
    int flightNumber;
    char departure[50];
    char destination[50];
    float duration; // in hours
};

int main() {
    struct Flight flights[3] = {
        {101, "New York", "London", 7.5},
        {102, "Los Angeles", "Tokyo", 11.0},
        {103, "Paris", "Rome", 2.0}
    };

    float maxDuration;

    printf("Enter maximum flight duration (in hours): ");
    scanf("%f", &maxDuration);

    printf("Flights with a duration less than %.2f hours:\n", maxDuration);

    for (int i = 0; i < 3; i++) {
        if (flights[i].duration < maxDuration) {
            printf("Flight Number: %d, Departure: %s, Destination: %s, Duration: %.2f
hours\n",
                   flights[i].flightNumber, flights[i].departure, flights[i].destination,
flights[i].duration);
        }
```

```
    }

    return 0;
}
```

9.Polynomial Representation:

Define a structure to represent a term of a polynomial (coefficient and exponent).

Write functions to add and multiply two polynomials.

```
#include <stdio.h>

struct Term {
    int coefficient;
    int exponent;
};

void addPolynomials(struct Term poly1, struct Term poly2) {
    if (poly1.exponent == poly2.exponent) {
        printf("Sum: %d x^%d\n", poly1.coefficient + poly2.coefficient, poly1.exponent);
    } else {
        printf("Polynomials have different exponents and can't be added directly.\n");
    }
}

void multiplyPolynomials(struct Term poly1, struct Term poly2) {
    int resultCoefficient = poly1.coefficient * poly2.coefficient;
    int resultExponent = poly1.exponent + poly2.exponent;

    printf("Product: %d x^%d\n", resultCoefficient, resultExponent);
```

```
}

int main() {
    struct Term poly1 = {3, 2};  // 3x^2
    struct Term poly2 = {4, 2};  // 4x^2

    addPolynomials(poly1, poly2);  // Add polynomials
    multiplyPolynomials(poly1, poly2);  // Multiply polynomials

    return 0;
}
```

10.Medical Records:

Create a structure for a patient's medical record with fields for name, age, diagnosis, and treatment.

Write a program to search for patients by diagnosis.

```
#include <stdio.h>
#include <string.h>

struct MedicalRecord {
    char name[50];
    int age;
    char diagnosis[50];
    char treatment[100];
};

int main() {
    struct MedicalRecord patients[3] = {
```

```
        {"John Doe", 45, "Flu", "Antiviral medication"},
        {"Jane Smith", 60, "Diabetes", "Insulin therapy"},
        {"Alice Brown", 30, "Flu", "Rest and hydration"}
    };

    char diagnosis[50];

    // Ask for the diagnosis to search
    printf("Enter diagnosis to search for: ");
    scanf("%s", diagnosis);

    int found = 0;

    // Search for matching diagnosis
    for (int i = 0; i < 3; i++) {
        if (strcmp(patients[i].diagnosis, diagnosis) == 0) {
            printf("Name: %s, Age: %d, Treatment: %s\n", patients[i].name,
patients[i].age, patients[i].treatment);
            found = 1;
        }
    }

    if (!found) {
        printf("No patients found with the diagnosis '%s'.\n", diagnosis);
    }

    return 0;
}
```

11.Game Scores:

Define a structure to store player information, including name, game played, and score.

Write a program to display the top scorer for each game.

```
#include <stdio.h>

struct Player {
    char name[50];
    char game[50];
    int score;
};

int main() {
    struct Player players[3] = {{"Alice", "Soccer", 10}, {"Bob", "Basketball", 15}, {"Charlie", "Soccer", 12}};
    int maxScore = 0;
    for (int i = 0; i < 3; i++) {
        if (players[i].score > maxScore) {
            maxScore = players[i].score;
        }
    }

    printf("Top scorers: ");
    for (int i = 0; i < 3; i++) {
        if (players[i].score == maxScore) {
            printf("%s ", players[i].name);
        }
    }
```

```
    return 0;
}
```

12.City Information:

Create a structure to store information about a city, including name, population, and area.

Write a program to calculate and display the population density of each city.

```
#include <stdio.h>

struct City {
    char name[50];
    int population;
    float area;
};

float calculateDensity(struct City c) {
    return c.population / c.area;
}

int main() {
    struct City city = {"CityX", 1000000, 500.0};
    printf("Population Density: %.2f\n", calculateDensity(city));
    return 0;
}
```

13.Vehicle Registration:

Define a structure for vehicle registration details, including registration number, owner, make, and year.

Write a program to list all vehicles registered in a given year.

```
#include <stdio.h>

struct Vehicle {
    char regNumber[20];
    char owner[50];
    int year;
};

int main() {
    struct Vehicle vehicles[3] = {{"AB123", "Alice", 2019}, {"CD456", "Bob", 2020}, {"EF789", "Charlie", 2021}};
    int year = 2020;

    printf("Vehicles registered in %d:\n", year);
    for (int i = 0; i < 3; i++) {
        if (vehicles[i].year == year) {
            printf("%s, Owner: %s\n", vehicles[i].regNumber, vehicles[i].owner);
        }
    }
    return 0;
}
```

14.Restaurant Menu:

Create a structure to represent a menu item with fields for name, category, and price.

Write a program to display menu items in a specific category.

```
#include <stdio.h>
```

```
struct MenuItem {
    char name[50];
    char category[20];
    float price;
};

int main() {
    struct MenuItem menu[3] = {{"Burger", "Main Course", 5.0}, {"Pasta", "Main Course", 7.5}, {"Ice Cream", "Dessert", 3.0}};
    char category[] = "Main Course";

    printf("Items in %s category:\n", category);
    for (int i = 0; i < 3; i++) {
        if (strcmp(menu[i].category, category) == 0) {
            printf("%s - %.2f\n", menu[i].name
```

15.Sports Team:

Define a structure for a sports team with fields for team name, sport, number of players, and coach.

Write a program to display all teams playing a specific sport.

```
#include <stdio.h>
#include <string.h>

struct SportsTeam {
    char teamName[50];
    char sport[50];
    int numPlayers;
    char coach[50];
};
```

```
int main() {
    struct SportsTeam teams[3] = {
        {"Eagles", "Football", 11, "John Doe"},
        {"Sharks", "Basketball", 5, "Jane Smith"},
        {"Tigers", "Football", 11, "Mike Johnson"}
    };

    char sport[50];

    // Ask for the sport to search
    printf("Enter sport to search for: ");
    scanf("%s", sport);

    // Display teams that play the specified sport
    int found = 0;
    for (int i = 0; i < 3; i++) {
        if (strcmp(teams[i].sport, sport) == 0) {
            printf("Team: %s, Coach: %s, Players: %d\n", teams[i].teamName,
teams[i].coach, teams[i].numPlayers);
            found = 1;
        }
    }

    if (!found) {
        printf("No teams found playing the sport '%s'.\n", sport);
    }

    return 0;
```

```
}
```

16.Student Marks Analysis:

Create a structure to store student marks in different subjects.

Write a program to calculate the total and percentage of marks for each student.

```
#include <stdio.h>

struct Student {
    char name[50];
    int marks[5];
};

float calculateTotal(struct Student s) {
    float total = 0;
    for (int i = 0; i < 5; i++) {
        total += s.marks[i];
    }
    return total;
}

float calculatePercentage(struct Student s) {
    return calculateTotal(s) / 5;
}

int main() {
    struct Student student = {"John", {80, 90, 85, 88, 92}};
    printf("Total: %.2f, Percentage: %.2f%%\n", calculateTotal(student),
calculatePercentage(student));
```

```
    return 0;
}
```

17.E-commerce Product:

Define a structure for an e-commerce product with fields for product ID, name, category, price, and stock.

Write a program to update the stock and calculate the total value of products in stock.

```
#include <stdio.h>

struct Product {
    int productID;
    char name[50];
    float price;
    int stock;
};

int main() {
    struct Product product = {101, "Laptop", 750.0, 50};
    int sold;

    // Display product details
    printf("Product: %s\nPrice: %.2f\nStock: %d\n", product.name, product.price, product.stock);

    // Ask for the number of products sold
    printf("Enter quantity sold: ");
    scanf("%d", &sold);
```

```
    // Update stock
    if (sold > product.stock) {
        printf("Not enough stock.\n");
    } else {
        product.stock -= sold;
        printf("Updated stock: %d\n", product.stock);
    }

    // Calculate total value of remaining stock
    float totalValue = product.price * product.stock;
    printf("Total value of products in stock: %.2f\n", totalValue);

    return 0;
}
```

18.Music Album:

Create a structure to store details of a music album, including album name, artist, genre, and release year.

Write a program to display albums of a specific genre.

```
#include <stdio.h>

struct MusicAlbum {
    char name[50];
    char artist[50];
    char genre[20];
    int year;
};
```

```
int main() {
    struct MusicAlbum albums[3] = {{"Album1", "Artist1", "Pop", 2021}, {"Album2", "Artist2", "Rock", 2020}, {"Album3", "Artist3", "Pop", 2022}};
    char genre[] = "Pop";

    printf("Albums in %s genre:\n", genre);
    for (int i = 0; i < 3; i++) {
        if (strcmp(albums[i].genre, genre) == 0) {
            printf("%s by %s\n", albums[i].name, albums[i].artist);
        }
    }
    return 0;
}
```

19.Cinema Ticket Booking:

Define a structure for a cinema ticket with fields for movie name, seat number, and price.

Write a program to book tickets and display the total revenue generated.

```
#include <stdio.h>

struct Ticket {
    char movieName[50];
    int seatNumber;
    float price;
};

int main() {
    struct Ticket tickets[2] = {{"Movie1", 1, 10.0}, {"Movie2", 2, 12.0}};
    float totalRevenue = 0.0;
```

```
    for (int i = 0; i < 2; i++) {
        totalRevenue += tickets[i].price;
    }

    printf("Total revenue: %.2f\n", totalRevenue);
    return 0;
}
```

20.University Courses:

Create a structure to store course details, including course code, name, instructor, and credits.

Write a program to list all courses taught by a specific instructor.

```
#include <stdio.h>

struct Course {
    char code[10];
    char name[50];
    char instructor[50];
    int credits;
};

int main() {
    struct Course courses[3] = {{"CS101", "Programming", "Dr. A", 4}, {"CS102", "Data Structures", "Dr. B", 3}, {"CS103", "Algorithms", "Dr. A", 4}};
    char instructor[] = "Dr. A";

    printf("Courses by %s:\n", instructor);
    for (int i = 0; i < 3; i++) {
```

```
        if (strcmp(courses[i].instructor, instructor) == 0) {
            printf("%s\n", courses[i].name);
        }
    }

    return 0;
}
```