----------Structures----------

1.Menu driven student program

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

struct student {
   char name[50];
   int rollnumber;
   float Marks;
};

struct student students[100];
int student_count = 0;

void add_record() {
   if (student_count < 100) {
      printf("Enter the name of the student: ");
      getchar();
      fgets(students[student_count].name, 50, stdin);
      printf("Enter the roll number: ");
      scanf("%d", &students[student_count].rollnumber);
      printf("Enter the marks: ");
      scanf("%f", &students[student_count].Marks);
      student_count += 1;
   } else {
      printf("Maximum capacity reached!\n");
   }
```

```
}

void display_record() {
    if (student_count == 0) {
        printf("No details to display.\n");
    } else {
        for (int i = 0; i < student_count; i++) {
            printf("Name: %s Roll Number: %d Marks: %.2f\n", students[i].name, students[i].rollnumber,
students[i].Marks);
        }
    }
}

void display_using_rollnumber() {
    int rn;
    printf("Enter the Roll Number: ");
    scanf("%d", &rn);

    int found = 0;
    for (int i = 0; i < student_count; i++) {
        if (students[i].rollnumber == rn) {
            printf("Name: %sRoll Number: %d Marks: %.2f\n", students[i].name, students[i].rollnumber,
students[i].Marks);
            found = 1;
            break;
        }
    }

    if (!found) {
```

```
        printf("No student found with Roll Number %d.\n", rn);
    }
}

void average_marks() {
    if (student_count == 0) {
        printf("No valid records!\n");
    } else {
        float avg = 0.0;
        float sum = 0.0;
        for (int i = 0; i < student_count; i++) {
            sum += students[i].Marks;
        }
        avg = sum / student_count;
        printf("The average marks obtained is %.2f\n", avg);
    }
}

int main() {
    bool is_on = true;
    while (is_on) {
        int user_input;
        printf("\nEnter the option\n1. Add Student\n2. Display All students\n3. Find students by roll number\n4. Calculate Average Marks\n5. Exit\n");
        scanf("%d", &user_input);

        switch (user_input) {
            case 1:
                add_record();
```

```
            break;
        case 2:
            display_record();
            break;
        case 3:
            display_using_rollnumber();
            break;
        case 4:
            average_marks();
            break;
        case 5:
            is_on = false;
            break;
        default:
            printf("Invalid option, please try again.\n");
            break;
      }
   }
   return 0;
}
```

2.

Employee details Printing

```
#include<stdio.h>
#include<string.h>
struct emp{
   char name[50];
   int id;
```

```
    float salary;
    char dept[10];
};
void display_high_salary_employees(struct emp employees[],int n,int threshold){
    for(int i=0;i<n;i++){
        if(employees[i].salary>threshold){
            printf("\n");
            printf("Name   : %s\n",employees[i].name);
            printf("id     : %d\n",employees[i].id);
            printf("salary : %.2f\n",employees[i].salary);
            printf("Department : %s\n",employees[i].dept);
        }
    }
}

int main(){
    int n;
    printf("Enter the number of employees :");
    scanf("%d",&n);
    struct emp employee[n];
    for(int i=0;i<n;i++){
        printf("Enter the details for employee %d\n",i+1);

        printf("Enter the employee name :");
        getchar();
        fgets(employee[i].name,50,stdin);
        printf("Enter the employee id :");
        scanf("%d",&employee[i].id);
        printf("Enter the employee salary :");
```

```
        scanf("%f",&employee[i].salary);

        printf("Enter the emplyee dept A/B/C :");
        getchar();
        fgets(employee[i].dept,3,stdin);
        printf("\n");
    }
    int threshold;
    printf("Enter the treshold :");
    scanf("%d",&threshold);
    display_high_salary_employees(employee, n, threshold);
    return 0;
}
```

3.

Book store inventory

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

struct Book {
    char title[100];
    char author[100];
    char isbn[20];
    float price;
};

void searchByTitle(struct Book inventory[], int n, const char* title) {
    for (int i = 0; i < n; i++) {
```

```
        if (strcmp(inventory[i].title, title) == 0) {
            printf("\nBook Found:\n");
            printf("Title: %s\n", inventory[i].title);
            printf("Author: %s\n", inventory[i].author);
            printf("ISBN: %s\n", inventory[i].isbn);
            printf("Price: %.2f\n", inventory[i].price);
            return;
        }
    }
    printf("Book not found.\n");
}

int main() {
    int n;
    printf("Enter the number of books: ");
    scanf("%d", &n);

    struct Book inventory[n];

    for (int i = 0; i < n; i++) {
        printf("Enter details for book %d:\n", i + 1);
        printf("Title: ");
        getchar();
        fgets(inventory[i].title, 100, stdin);
        inventory[i].title[strcspn(inventory[i].title, "\n")] = '\0';

        printf("Author: ");
        fgets(inventory[i].author, 100, stdin);
        inventory[i].author[strcspn(inventory[i].author, "\n")] = '\0';
```

```
        printf("ISBN: ");
        fgets(inventory[i].isbn, 20, stdin);
        inventory[i].isbn[strcspn(inventory[i].isbn, "\n")] = '\0';

        printf("Price: ");
        scanf("%f", &inventory[i].price);
        getchar();
    }

    char searchTitle[100];
    printf("\nEnter the title of the book to search: ");
    fgets(searchTitle, 100, stdin);
    searchTitle[strcspn(searchTitle, "\n")] = '\0';

    searchByTitle(inventory, n, searchTitle);

    return 0;
}
```

4.

Valid date checking

```
#include<stdio.h>
#include<stdbool.h>
struct dates{
    int day;
    int month;
    int year;
```

```
};
int isLeapYear(int year){
   if((year%400==0 )||(year%4==0 && year%100!=0)){
      return true;
   }else{
      return false;
   }
}

int validate_dates(struct dates date){
   if(date.year<1){
      return 0;
   }
   if (date.month < 1 || date.month > 12) {
      return 0;
   }
   int daysinMonth;
   switch (date.month) {
      case 1:
      case 3:
      case 5:
      case 7:
      case 8:
      case 10:
      case 12:
         daysinMonth = 31;
         break;

      case 4:
```

```
        case 6:
        case 9:
        case 11:
            daysinMonth = 30;
            break;

        case 2:
            if (isLeapYear(date.year)) {
                daysinMonth = 29;
            } else {
                daysinMonth = 28;
            }
            break;

        default:
            return 0;
    }
    if(date.day<1 || date.month>daysinMonth){
        return false;
    }
    return true;
}

int main(){
    int n;
    printf("Enter the no of dates :");
    scanf("%d",&n);
    struct dates date[n];
    for(int i=0;i<n;i++){
```

```
        printf("Enter the day :");
        scanf("%d",&date[i].day);
        printf("Enter the month:");
        scanf("%d",&date[i].month);
        printf("Enter the year :");
        scanf("%d",&date[i].year);
        if (validate_dates(date[i])) {
            printf("Date %d/%d/%d is valid.\n", date[i].day, date[i].month, date[i].year);
        } else {
            printf("Date %d/%d/%d is invalid.\n", date[i].day, date[i].month, date[i].year);
        }
    }
}
```

5.

Complex numbers

```
#include <stdio.h>

struct Complex {
    float real;
    float imag;
};

struct Complex addComplex(struct Complex num1, struct Complex num2) {
    struct Complex result;
    result.real = num1.real + num2.real;
    result.imag = num1.imag + num2.imag;
    return result;
```

```
}

struct Complex subtractComplex(struct Complex num1, struct Complex num2) {
    struct Complex result;
    result.real = num1.real - num2.real;
    result.imag = num1.imag - num2.imag;
    return result;
}

struct Complex multiplyComplex(struct Complex num1, struct Complex num2) {
    struct Complex result;
    result.real = num1.real * num2.real - num1.imag * num2.imag;
    result.imag = num1.real * num2.imag + num1.imag * num2.real;
    return result;
}

void displayComplex(struct Complex num) {
    if (num.imag >= 0) {
        printf("%.2f + %.2fi\n", num.real, num.imag);
    } else {
        printf("%.2f - %.2fi\n", num.real, -num.imag);
    }
}

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

    printf("Enter the real and imaginary parts of the first complex number (a + bi):\n");
    printf("Real part: ");
```

```
    scanf("%f", &num1.real);
    printf("Imaginary part: ");
    scanf("%f", &num1.imag);

    printf("Enter the real and imaginary parts of the second complex number (a + bi):\n");
    printf("Real part: ");
    scanf("%f", &num2.real);
    printf("Imaginary part: ");
    scanf("%f", &num2.imag);

    result = addComplex(num1, num2);
    printf("\nSum: ");
    displayComplex(result);

    result = subtractComplex(num1, num2);
    printf("\nDifference: ");
    displayComplex(result);

    result = multiplyComplex(num1, num2);
    printf("\nProduct: ");
    displayComplex(result);

    return 0;
}
```

6.

```
#include<stdio.h>
#include<stdbool.h>
struct det{
```

```
    int acc_no;
    char name[30];
    float balance;
};
void Deposite(struct det *details){
    int amount;
    printf("Enter the amount :");
    scanf("%d",&amount);
    details->balance+=amount;
    printf("Sucessfully added!\n");
}
void withdraw(struct det *details){
    int amount;
    printf("Enter the amount :");
    scanf("%d",&amount);
    details->balance-=amount;
    printf("Sucessfully withdrawed!\n");
}
void view_balance(struct det *details){
    printf("the balance is : %.2f\n",details->balance);
}
int main(){
    struct det details;
    printf("Enter the name of bank Holder :");
    getchar();
    fgets(details.name,30,stdin);
    printf("enter the Account number :");
    scanf("%d",&details.acc_no);
    printf("enter the balance :");
```

```
    scanf("%f",&details.balance);
    printf("\n");
    int user_option;
    bool is_on=true;
    while(is_on){
        printf("1.Deposite\n2.Withdraw\n3.View Balance\n4.Exit\n");
        printf("Enter your option :");
        scanf("%d",&user_option);
        if(user_option==1){
            Deposite(&details);
        }else if(user_option==2){
            withdraw(&details);
        }else if(user_option==3){
            view_balance(&details);
        }else if(user_option==4){
            is_on=false;
        }else{
          printf("enter a valid option!\n");
        }
    }

}
```

7. Car Inventory System

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

struct Car {
    char make[30];
```

```
    char model[30];
    int year;
    float price;
};

void printCar(struct Car car) {
    printf("\nMake: %s\n", car.make);
    printf("Model: %s\n", car.model);
    printf("Year: %d\n", car.year);
    printf("Price: %.2f\n", car.price);
}

void printCarsInPriceRange(struct Car cars[], int n, float minPrice, float maxPrice) {
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (cars[i].price >= minPrice && cars[i].price <= maxPrice) {
            printCar(cars[i]);
            found = 1;
        }
    }
    if (!found) {
        printf("No cars found in the specified price range.\n");
    }
}

int main() {
    int n;
    printf("Enter the number of cars: ");
    scanf("%d", &n);
```

```
struct Car cars[n];

for (int i = 0; i < n; i++) {
    printf("\nEnter details for car %d:\n", i + 1);

    printf("Enter make: ");
    getchar();
    fgets(cars[i].make, sizeof(cars[i].make), stdin);
    cars[i].make[strcspn(cars[i].make, "\n")] = '\0';

    printf("Enter model: ");
    fgets(cars[i].model, sizeof(cars[i].model), stdin);
    cars[i].model[strcspn(cars[i].model, "\n")] = '\0';

    printf("Enter year: ");
    scanf("%d", &cars[i].year);

    printf("Enter price: ");
    scanf("%f", &cars[i].price);
}

float minPrice, maxPrice;
printf("\nEnter minimum price: ");
scanf("%f", &minPrice);

printf("Enter maximum price: ");
scanf("%f", &maxPrice);
printf("\nCars within the price range %.2f - %.2f:\n", minPrice, maxPrice);
```

```
    printCarsInPriceRange(cars, n, minPrice, maxPrice);

    return 0;
}
```

8.Library Management

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

struct Book {
    char title[50];
    char author[50];
    int pub_year;
    char status[10];
};

void issueBook(struct Book *book) {
    if (strcmp(book->status, "available") == 0) {
        strcpy(book->status, "issued");
        printf("The book \"%s\" has been issued successfully!\n", book->title);
    } else {
        printf("The book \"%s\" is already issued and cannot be issued again.\n", book->title);
    }
}

void returnBook(struct Book *book) {
    if (strcmp(book->status, "issued") == 0) {
        strcpy(book->status, "available");
```

```
        printf("The book \"%s\" has been returned successfully!\n", book->title);
    } else {
        printf("The book \"%s\" is not issued, so it cannot be returned.\n", book->title);
    }
}

void displayBook(struct Book book) {
    printf("\nTitle: %s\n", book.title);
    printf("Author: %s\n", book.author);
    printf("Publication Year: %d\n", book.pub_year);
    printf("Status: %s\n", book.status);
}

int main() {
    int n;
    printf("Enter the number of books in the library: ");
    scanf("%d", &n);

    struct Book library[n];
    for (int i = 0; i < n; i++) {
        printf("\nEnter details for book %d:\n", i + 1);

        printf("Enter title: ");
        getchar();
        fgets(library[i].title, sizeof(library[i].title), stdin);
        library[i].title[strcspn(library[i].title, "\n")] = '\0';

        printf("Enter author: ");
        fgets(library[i].author, sizeof(library[i].author), stdin);
```

```
    library[i].author[strcspn(library[i].author, "\n")] = '\0';

    printf("Enter publication year: ");
    scanf("%d", &library[i].pub_year);


    strcpy(library[i].status, "available");
}

int choice;
bool is_on = true;
while (is_on) {
    printf("\nLibrary Management System\n");
    printf("1. Issue a Book\n");
    printf("2. Return a Book\n");
    printf("3. View Book Details\n");
    printf("4. Exit\n");

    printf("Enter your choice: ");
    scanf("%d", &choice);

    if (choice == 1) {
        int bookIndex;
        printf("Enter the book number to issue (1 to %d): ", n);
        scanf("%d", &bookIndex);

        if (bookIndex < 1 || bookIndex > n) {
            printf("Invalid book number!\n");
        } else {
```

```
            issueBook(&library[bookIndex - 1]);
        }
    } else if (choice == 2) {
        int bookIndex;
        printf("Enter the book number to return (1 to %d): ", n);
        scanf("%d", &bookIndex);

        if (bookIndex < 1 || bookIndex > n) {
            printf("Invalid book number!\n");
        } else {
            returnBook(&library[bookIndex - 1]);
        }
    } else if (choice == 3) {
        int bookIndex;
        printf("Enter the book number to view details (1 to %d): ", n);
        scanf("%d", &bookIndex);

        if (bookIndex < 1 || bookIndex > n) {
            printf("Invalid book number!\n");
        } else {
            displayBook(library[bookIndex - 1]);
        }
    } else if (choice == 4) {
        is_on = false;
    } else {
        printf("Invalid choice, please try again.\n");
    }
}
```

```
    return 0;
}

9. Student Grades
#include <stdio.h>
#include <string.h>

#define MAX_GRADES 5

struct Student {
    char name[50];
    int roll_no;
    float grades[MAX_GRADES];
};

void calculateGrades(struct Student student) {
    float sum = 0;
    float highest = student.grades[0];
    float lowest = student.grades[0];
    for (int i = 0; i < MAX_GRADES; i++) {
        sum += student.grades[i];

        if (student.grades[i] > highest) {
            highest = student.grades[i];
        }
        if (student.grades[i] < lowest) {
            lowest = student.grades[i];
        }
    }
```

```
    float average = sum / MAX_GRADES;

    // Print the results
    printf("\nStudent Name: %s\n", student.name);
    printf("Roll Number: %d\n", student.roll_no);
    printf("Highest Grade: %.2f\n", highest);
    printf("Lowest Grade: %.2f\n", lowest);
    printf("Average Grade: %.2f\n", average);
}

int main() {
    int n;
    printf("Enter the number of students: ");
    scanf("%d", &n);

    struct Student students[n];

    for (int i = 0; i < n; i++) {
        printf("\nEnter details for student %d:\n", i + 1);

        printf("Enter student's name: ");
        getchar();
        fgets(students[i].name, sizeof(students[i].name), stdin);
        students[i].name[strcspn(students[i].name, "\n")] = '\0';

        printf("Enter roll number: ");
        scanf("%d", &students[i].roll_no);
```

```
        printf("Enter %d grades for student:\n", MAX_GRADES);
        for (int j = 0; j < MAX_GRADES; j++) {
            printf("Grade %d: ", j + 1);
            scanf("%f", &students[i].grades[j]);
        }
    }
    for (int i = 0; i < n; i++) {
        calculateGrades(students[i]);
    }

    return 0;
}
```

10.

```
#include<stdio.h>
#include<string.h>
struct pro{
    char name[30];
    int quantity;
    float price;
};
int main(){
   int n;
   printf("Enter the number of products in the catelouge :");
   scanf("%d",&n);
   struct pro product[n];
   for(int i=0;i<n;i++){
       printf("Enter the name for product %d :",i+1);
       getchar();
```

```
    fgets(product[i].name,30,stdin);
    printf("Enter the quantity :");
    scanf("%d",&product[i].quantity);
    printf("Enter the price :");
    scanf("%f",&product[i].price);
  }
  static int total_sales=0;
  for(int i=0;i<n;i++){
    int individual_sale;
    printf("\nEnter the quantity sold for %s :",product[i].name);
    scanf("%d",&individual_sale);
    total_sales+=(individual_sale*product[i].price);
    printf("\nSale for %d for %s is successful!",individual_sale,product[i].name);
  }
  printf("\nThe total sales done is Rs.%d.",total_sales);
  return 0;
}
```

11.Point distance calculation

```
#include<stdio.h>
#include<math.h>
struct points{
   int x;
   int y;
};
float calculate_distance(struct points p1,struct points p2){
   return sqrt(pow(p2.x - p1.x, 2) + pow(p2.y - p1.y, 2));
}
int main(){
```

```
    struct points point1,point2;
    printf("enter the coordinated for point 1 :");
    scanf("%d%d",&point1.x,&point2.y);
    printf("enter the coordinated for point 2 :");
    scanf("%d%d",&point2.x,&point2.y);
    float result=calculate_distance(point1,point2);
    printf("the distance between 2 points is %.2f",result);
}
```

12.Rectangle Properties

```
#include <stdio.h>
struct Rectangle{
    float length;
    float width;
} ;

float calculateArea(struct Rectangle rect) {
    return rect.length * rect.width;
}

float calculatePerimeter(struct Rectangle rect) {
    return 2 * (rect.length + rect.width);
}

int main() {
    struct Rectangle rect;
    printf("Enter the length of the rectangle: ");
    scanf("%f", &rect.length);
    printf("Enter the width of the rectangle: ");
```

```
    scanf("%f", &rect.width);
    float area = calculateArea(rect);
    float perimeter = calculatePerimeter(rect);
    printf("Area of the rectangle: %.2f\n", area);
    printf("Perimeter of the rectangle: %.2f\n", perimeter);

    return 0;
}

13
Movie Sorting:
#include <stdio.h>
#include <string.h>
struct Movie{
    char title[100];
    char director[100];
    int releaseYear;
    float rating;
};

void sortMoviesByRating(struct Movie movies[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (movies[j].rating < movies[j + 1].rating) {
                struct Movie temp = movies[j];
                movies[j] = movies[j + 1];
                movies[j + 1] = temp;
            }
        }
```

```
    }
}

int main() {
    int n;
    printf("Enter the number of movies: ");
    scanf("%d", &n);
    struct Movie movies[n];
    for (int i = 0; i < n; i++) {
        printf("Enter details for movie %d:\n", i + 1);

        printf("Title: ");
        getchar();
        fgets(movies[i].title, 100, stdin);
        movies[i].title[strcspn(movies[i].title, "\n")] = '\0';

        printf("Director: ");
        fgets(movies[i].director, 100, stdin);
        movies[i].director[strcspn(movies[i].director, "\n")] = '\0';

        printf("Release Year: ");
        scanf("%d", &movies[i].releaseYear);

        printf("Rating: ");
        scanf("%f", &movies[i].rating);
    }
    sortMoviesByRating(movies, n);
    printf("\nMovies sorted by rating (highest to lowest):\n");
    for (int i = 0; i < n; i++) {
```

```
        printf("Title: %s, Director: %s, Release Year: %d, Rating: %.1f\n",
                movies[i].title, movies[i].director, movies[i].releaseYear, movies[i].rating);
    }

    return 0;
}
```

14.

Highest temperature

```
#include<stdio.h>
#include<math.h>
struct date{
    int day;
    int month;
    int year;
};
struct weather{
    struct date dt;
    int temp;
    int humidity;
};
int find_max_temp(struct weather data[],int n){
    float max=-INFINITY;
    for(int i=0;i<n;i++){
        if(data[i].temp>max){
            max=data[i].temp;
        }
    }
    return max;
```

```
}
int main(){
   int n;
   printf("Enter the number of days :");
   scanf("%d",&n);
   struct weather data[n];
   for(int i=0;i<n;i++){
      printf("\n Enter the details %d ",i+1);
      printf("Enter the day :");
      scanf("%d",&data[i].dt.day);
      printf("Enter the month :");
      scanf("%d",&data[i].dt.month);
      printf("Enter the year :");
      scanf("%d",&data[i].dt.year);
      printf("Enter the temperature :");
      scanf("%d",&data[i].temp);
      printf("Enter the humidity :");
      scanf("%d",&data[i].humidity);
   }
   int result=find_max_temp(data,n);
   for(int i=0;i<n;i++){
      if(data[i].temp==result){
         printf("The highest temperature was recorded on
:%d/%d/%d",data[i].dt.day,data[i].dt.month,data[i].dt.year);
      }
   }
   return 0;
}
```

15

FractionArithmetic

```
#include <stdio.h>

typedef struct {
    int numerator;
    int denominator;
} Fraction;

int gcd(int a, int b) {
    while (b != 0) {
        int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

Fraction simplify(Fraction f) {
    int divisor = gcd(f.numerator, f.denominator);
    f.numerator /= divisor;
    f.denominator /= divisor;
    if (f.denominator < 0) {
        f.numerator = -f.numerator;
        f.denominator = -f.denominator;
    }
    return f;
}
```

```
Fraction addFractions(Fraction f1, Fraction f2) {
   Fraction result;
   result.numerator = f1.numerator * f2.denominator + f2.numerator * f1.denominator;
   result.denominator = f1.denominator * f2.denominator;
   return simplify(result);
}

Fraction subtractFractions(Fraction f1, Fraction f2) {
   Fraction result;
   result.numerator = f1.numerator * f2.denominator - f2.numerator * f1.denominator;
   result.denominator = f1.denominator * f2.denominator;
   return simplify(result);
}

Fraction multiplyFractions(Fraction f1, Fraction f2) {
   Fraction result;
   result.numerator = f1.numerator * f2.numerator;
   result.denominator = f1.denominator * f2.denominator;
   return simplify(result);
}

Fraction divideFractions(Fraction f1, Fraction f2) {
   Fraction result;
   result.numerator = f1.numerator * f2.denominator;
   result.denominator = f1.denominator * f2.numerator;
   return simplify(result);
}
```

```
int main() {
    Fraction f1, f2, result;
    printf("Enter the numerator and denominator of the first fraction: ");
    scanf("%d %d", &f1.numerator, &f1.denominator);
    printf("Enter the numerator and denominator of the second fraction: ");
    scanf("%d %d", &f2.numerator, &f2.denominator);

    result = addFractions(f1, f2);
    printf("Addition: %d/%d\n", result.numerator, result.denominator);

    result = subtractFractions(f1, f2);
    printf("Subtraction: %d/%d\n", result.numerator, result.denominator);

    result = multiplyFractions(f1, f2);
    printf("Multiplication: %d/%d\n", result.numerator, result.denominator);

    result = divideFractions(f1, f2);
    printf("Division: %d/%d\n", result.numerator, result.denominator);

    return 0;
}
```

16

laptop inventory

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

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

```
    char model[50];
    char processor[50];
    int ram;
    float price;
} Laptop;

void listLaptopsWithinPriceRange(Laptop laptops[], int n, float minPrice, float maxPrice) {
    printf("Laptops within the price range %.2f to %.2f:\n", minPrice, maxPrice);
    int found = 0;
    for (int i = 0; i < n; 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);
            found = 1;
        }
    }
    if (!found) {
        printf("No laptops found in this price range.\n");
    }
}

int main() {
    int n;
    printf("Enter the number of laptops: ");
    scanf("%d", &n);

    Laptop laptops[n];
    for (int i = 0; i < n; i++) {
```

```
    printf("Enter details for laptop %d:\n", i + 1);
    printf("Brand: ");
    getchar(); // Consume newline left by previous input
    fgets(laptops[i].brand, 50, stdin);
    laptops[i].brand[strcspn(laptops[i].brand, "\n")] = '\0';

    printf("Model: ");
    fgets(laptops[i].model, 50, stdin);
    laptops[i].model[strcspn(laptops[i].model, "\n")] = '\0';

    printf("Processor: ");
    fgets(laptops[i].processor, 50, stdin);
    laptops[i].processor[strcspn(laptops[i].processor, "\n")] = '\0';

    printf("RAM (in GB): ");
    scanf("%d", &laptops[i].ram);

    printf("Price: ");
    scanf("%f", &laptops[i].price);
}

float minPrice, maxPrice;
printf("Enter the minimum price: ");
scanf("%f", &minPrice);
printf("Enter the maximum price: ");
scanf("%f", &maxPrice);

listLaptopsWithinPriceRange(laptops, n, minPrice, maxPrice);
```

```
    return 0;
}
```

17

student attendence

```
#include <stdio.h>

typedef struct {
    int studentID;
    int totalClasses;
    int classesAttended;
} Attendance;

void calculateAndDisplayAttendance(Attendance students[], int n) {
    printf("Student Attendance Data:\n");
    printf("Student ID\tTotal Classes\tClasses Attended\tAttendance Percentage\n");
    for (int i = 0; i < n; i++) {
        float percentage = ((float)students[i].classesAttended / students[i].totalClasses) * 100;
        printf("%d\t\t%d\t\t%d\t\t\t%.2f%%\n",
               students[i].studentID, students[i].totalClasses, students[i].classesAttended, percentage);
    }
}

int main() {
    int n;
    printf("Enter the number of students: ");
    scanf("%d", &n);

    Attendance students[n];
```

```
    for (int i = 0; i < n; i++) {
        printf("Enter details for student %d:\n", i + 1);
        printf("Student ID: ");
        scanf("%d", &students[i].studentID);

        printf("Total Classes: ");
        scanf("%d", &students[i].totalClasses);

        printf("Classes Attended: ");
        scanf("%d", &students[i].classesAttended);

        if (students[i].classesAttended > students[i].totalClasses) {
            printf("Error: Classes attended cannot be greater than total classes. Please re-enter the data.\n");
            i--; // Retry for the same student
        }
    }

    calculateAndDisplayAttendance(students, n);

    return 0;
}
```

18

flight information

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

typedef struct {
    char flightNumber[10];
```

```
    char departure[50];
    char destination[50];
    float duration;
} Flight;

void displayFlightsByDuration(Flight flights[], int n, float maxDuration) {
    printf("Flights with a duration less than %.2f hours:\n", maxDuration);
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (flights[i].duration < maxDuration) {
            printf("Flight Number: %s, Departure: %s, Destination: %s, Duration: %.2f hours\n",
                flights[i].flightNumber, flights[i].departure, flights[i].destination, flights[i].duration);
            found = 1;
        }
    }
    if (!found) {
        printf("No flights found with a duration less than %.2f hours.\n", maxDuration);
    }
}

int main() {
    int n;
    printf("Enter the number of flights: ");
    scanf("%d", &n);

    Flight flights[n];
    for (int i = 0; i < n; i++) {
        printf("Enter details for flight %d:\n", i + 1);
```

```
    printf("Flight Number: ");
    getchar(); // Consume newline left by previous input
    fgets(flights[i].flightNumber, 10, stdin);
    flights[i].flightNumber[strcspn(flights[i].flightNumber, "\n")] = '\0';

    printf("Departure: ");
    fgets(flights[i].departure, 50, stdin);
    flights[i].departure[strcspn(flights[i].departure, "\n")] = '\0';

    printf("Destination: ");
    fgets(flights[i].destination, 50, stdin);
    flights[i].destination[strcspn(flights[i].destination, "\n")] = '\0';

    printf("Duration (in hours): ");
    scanf("%f", &flights[i].duration);
  }

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

  displayFlightsByDuration(flights, n, maxDuration);

  return 0;
}
```

19

polynomial representation

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

```
typedef struct {
    int coefficient;
    int exponent;
} Term;

void addPolynomials(Term p1[], int n1, Term p2[], int n2, Term result[], int *resSize) {
    int i = 0, j = 0, k = 0;
    while (i < n1 && j < n2) {
        if (p1[i].exponent == p2[j].exponent) {
            result[k].coefficient = p1[i].coefficient + p2[j].coefficient;
            result[k].exponent = p1[i].exponent;
            i++; j++;
        } else if (p1[i].exponent > p2[j].exponent) {
            result[k] = p1[i];
            i++;
        } else {
            result[k] = p2[j];
            j++;
        }
        k++;
    }
    while (i < n1) result[k++] = p1[i++];
    while (j < n2) result[k++] = p2[j++];
    *resSize = k;
}

void displayPolynomial(Term poly[], int size) {
    for (int i = 0; i < size; i++) {
```

```
        printf("%d(x^%d)", poly[i].coefficient, poly[i].exponent);
        if (i < size - 1) printf(" + ");
    }
    printf("\n");
}

int main() {
    Term p1[] = {{3, 4}, {2, 3}, {1, 0}};
    Term p2[] = {{5, 3}, {1, 2}, {2, 0}};
    Term result[10];
    int resSize = 0;

    addPolynomials(p1, 3, p2, 3, result, &resSize);

    printf("Resultant Polynomial: ");
    displayPolynomial(result, resSize);
    return 0;
}
```

20

medical records

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

typedef struct {
    char name[50];
    int age;
    char diagnosis[50];
    char treatment[50];
```

```
} MedicalRecord;

void searchByDiagnosis(MedicalRecord records[], int n, char diagnosis[]) {
   printf("Patients with diagnosis '%s':\n", diagnosis);
   int found = 0;
   for (int i = 0; i < n; i++) {
      if (strcmp(records[i].diagnosis, diagnosis) == 0) {
         printf("Name: %s, Age: %d, Treatment: %s\n", records[i].name, records[i].age,
records[i].treatment);
         found = 1;
      }
   }
   if (!found) printf("No patients found.\n");
}

int main() {
   MedicalRecord records[] = {
      {"Alice", 30, "Diabetes", "Insulin Therapy"},
      {"Bob", 45, "Hypertension", "Medication"},
      {"Carol", 50, "Diabetes", "Diet Control"}
   };
   char diagnosis[50];
   printf("Enter diagnosis to search: ");
   scanf("%s", diagnosis);
   searchByDiagnosis(records, 3, diagnosis);
   return 0;
}
```

Game scores

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

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

void displayTopScorers(Player players[], int n) {
    for (int i = 0; i < n; i++) {
        int maxScore = players[i].score;
        int topIndex = i;
        for (int j = i + 1; j < n; j++) {
            if (strcmp(players[i].game, players[j].game) == 0 && players[j].score > maxScore) {
                maxScore = players[j].score;
                topIndex = j;
            }
        }
        printf("Game: %s, Top Scorer: %s, Score: %d\n", players[topIndex].game, players[topIndex].name,
maxScore);
        i = topIndex;
    }
}

int main() {
    Player players[] = {
        {"Alice", "Chess", 85}, {"Bob", "Chess", 95},
```

```
        {"Carol", "Tennis", 70}, {"Dave", "Tennis", 80}
    };
    displayTopScorers(players, 4);
    return 0;
}
```

22

city information

```
#include <stdio.h>

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

void calculatePopulationDensity(City cities[], int n) {
    printf("City Information:\n");
    for (int i = 0; i < n; i++) {
        float density = cities[i].population / cities[i].area;
        printf("City: %s, Population Density: %.2f people/sq.km\n", cities[i].name, density);
    }
}

int main() {
    City cities[] = {
        {"New York", 8419600, 789.4},
        {"Los Angeles", 3980400, 1214.9},
        {"Chicago", 2716000, 588.7}
```

```
    };
    calculatePopulationDensity(cities, 3);
    return 0;
}
```

23

vehicle registration

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

typedef struct {
    char registrationNumber[20];
    char owner[50];
    char make[50];
    int year;
} Vehicle;

void listVehiclesByYear(Vehicle vehicles[], int n, int year) {
    printf("Vehicles registered in %d:\n", year);
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (vehicles[i].year == year) {
            printf("Registration: %s, Owner: %s, Make: %s\n",
                vehicles[i].registrationNumber, vehicles[i].owner, vehicles[i].make);
            found = 1;
        }
    }
    if (!found) printf("No vehicles found for the year %d.\n", year);
}
```

```
int main() {
    Vehicle vehicles[] = {
        {"KA01AB1234", "Alice", "Toyota", 2019},
        {"KA02CD5678", "Bob", "Honda", 2020},
        {"KA03EF9101", "Carol", "Ford", 2019}
    };
    int year;
    printf("Enter the year: ");
    scanf("%d", &year);
    listVehiclesByYear(vehicles, 3, year);
    return 0;
}
```

24

restaurent menu.

```
#include <stdio.h>

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

void displayMenu(MenuItem menu[], int n) {
    printf("Restaurant Menu:\n");
    for (int i = 0; i < n; i++) {
        printf("Item: %s, Category: %s, Price: %.2f\n",
               menu[i].name, menu[i].category, menu[i].price);
```

```
    }
}

int main() {
    MenuItem menu[] = {
        {"Pasta", "Main Course", 250.00},
        {"Burger", "Fast Food", 150.00},
        {"Ice Cream", "Dessert", 100.00}
    };
    displayMenu(menu, 3);
    return 0;
}
```

25.

sports team

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

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

void displayTeamsBySport(SportsTeam teams[], int n, char sport[]) {
    printf("Teams playing '%s':\n", sport);
    int found = 0;
    for (int i = 0; i < n; 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 for this sport.\n");
}

int main() {
    SportsTeam teams[] = {
        {"Team A", "Football", 11, "John Doe"},
        {"Team B", "Basketball", 5, "Jane Smith"},
        {"Team C", "Football", 11, "Sam Brown"}
    };
    char sport[50];
    printf("Enter sport to search for teams: ");
    scanf("%s", sport);
    displayTeamsBySport(teams, 3, sport);
    return 0;
}
```

27.

student mark analysis

```
#include <stdio.h>

typedef struct {
    char name[50];
    int marks[5];
```

```
    float total;
    float percentage;
} Student;

void calculateMarks(Student students[], int n) {
    for (int i = 0; i < n; i++) {
        students[i].total = 0;
        for (int j = 0; j < 5; j++) {
            students[i].total += students[i].marks[j];
        }
        students[i].percentage = (students[i].total / 500) * 100;
    }
}

void displayMarks(Student students[], int n) {
    for (int i = 0; i < n; i++) {
        printf("Student: %s, Total: %.2f, Percentage: %.2f%%\n", students[i].name, students[i].total,
students[i].percentage);
    }
}

int main() {
    Student students[] = {
        {"Alice", {85, 90, 88, 92, 86}, 0, 0},
        {"Bob", {78, 82, 80, 75, 88}, 0, 0}
    };
    calculateMarks(students, 2);
    displayMarks(students, 2);
    return 0;
```

```
}
```

28.

Ecommerce product

```
#include <stdio.h>

typedef struct {
    int productId;
    char name[50];
    char category[50];
    float price;
    int stock;
} Product;

void updateStock(Product *product, int quantity) {
    product->stock += quantity;
}

float calculateStockValue(Product product) {
    return product.price * product.stock;
}

int main() {
    Product product = {101, "Laptop", "Electronics", 50000.00, 10};

    int quantity;
    printf("Enter quantity to add to stock: ");
    scanf("%d", &quantity);
    updateStock(&product, quantity);
```

```
    printf("Updated Stock: %d, Total Stock Value: %.2f\n", product.stock, calculateStockValue(product));
    return 0;
}
```

29.

music album

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

typedef struct {
    char albumName[50];
    char artist[50];
    char genre[50];
    int releaseYear;
} MusicAlbum;

void displayAlbumsByGenre(MusicAlbum albums[], int n, char genre[]) {
    printf("Albums of genre '%s':\n", genre);
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (strcmp(albums[i].genre, genre) == 0) {
            printf("Album: %s, Artist: %s, Release Year: %d\n", albums[i].albumName, albums[i].artist,
albums[i].releaseYear);
            found = 1;
        }
    }
    if (!found) printf("No albums found for this genre.\n");
}
```

```
int main() {
    MusicAlbum albums[] = {
        {"Album1", "Artist1", "Pop", 2020},
        {"Album2", "Artist2", "Rock", 2019},
        {"Album3", "Artist1", "Pop", 2021}
    };
    char genre[50];
    printf("Enter genre to search for albums: ");
    scanf("%s", genre);
    displayAlbumsByGenre(albums, 3, genre);
    return 0;
}
```

30.

cinema ticke booking

```
#include <stdio.h>

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

float totalRevenue = 0;

void bookTicket(Ticket *ticket, float price) {
    ticket->price = price;
    totalRevenue += price;
```

```
}

void displayTotalRevenue() {
    printf("Total Revenue Generated: %.2f\n", totalRevenue);
}

int main() {
    Ticket ticket1 = {"Movie1", 1, 0};
    Ticket ticket2 = {"Movie2", 2, 0};

    bookTicket(&ticket1, 150.00);
    bookTicket(&ticket2, 200.00);

    displayTotalRevenue();
    return 0;
}
```

31.

university courses

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

typedef struct {
    char courseCode[10];
    char courseName[50];
    char instructor[50];
    int credits;
} Course;
```

```
void listCoursesByInstructor(Course courses[], int n, char instructor[]) {
    printf("Courses taught by '%s':\n", instructor);
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (strcmp(courses[i].instructor, instructor) == 0) {
            printf("Course: %s, Code: %s, Credits: %d\n", courses[i].courseName, courses[i].courseCode,
courses[i].credits);
            found = 1;
        }
    }
    if (!found) printf("No courses found for this instructor.\n");
}

int main() {
    Course courses[] = {
        {"CS101", "Introduction to Computer Science", "Dr. Smith", 3},
        {"CS102", "Data Structures", "Dr. Johnson", 4},
        {"CS103", "Algorithms", "Dr. Smith", 4}
    };
    char instructor[50];
    printf("Enter instructor name to list courses: ");
    scanf("%s", instructor);
    listCoursesByInstructor(courses, 3, instructor);
    return 0;
}
```