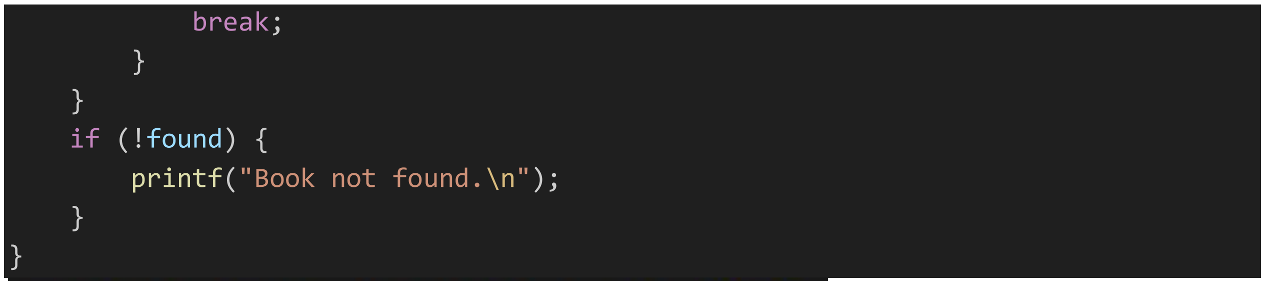
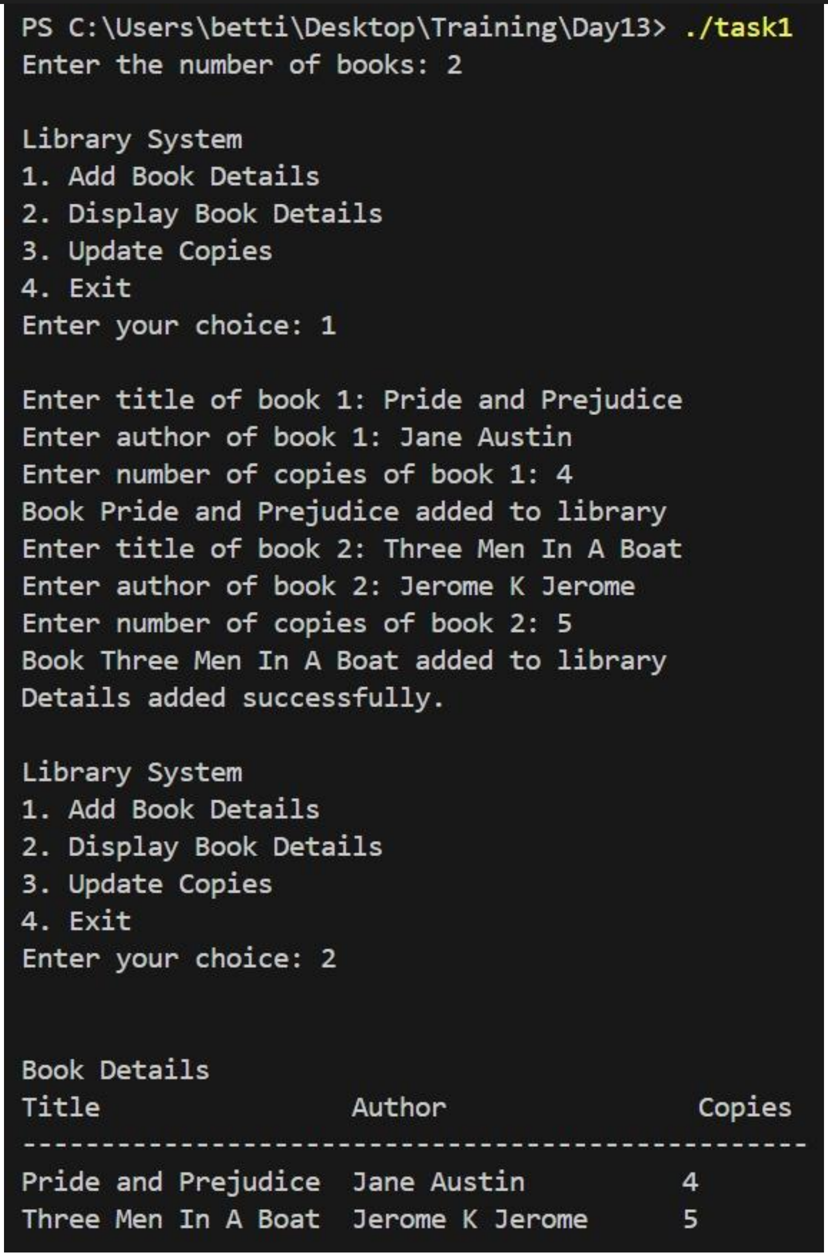
1.

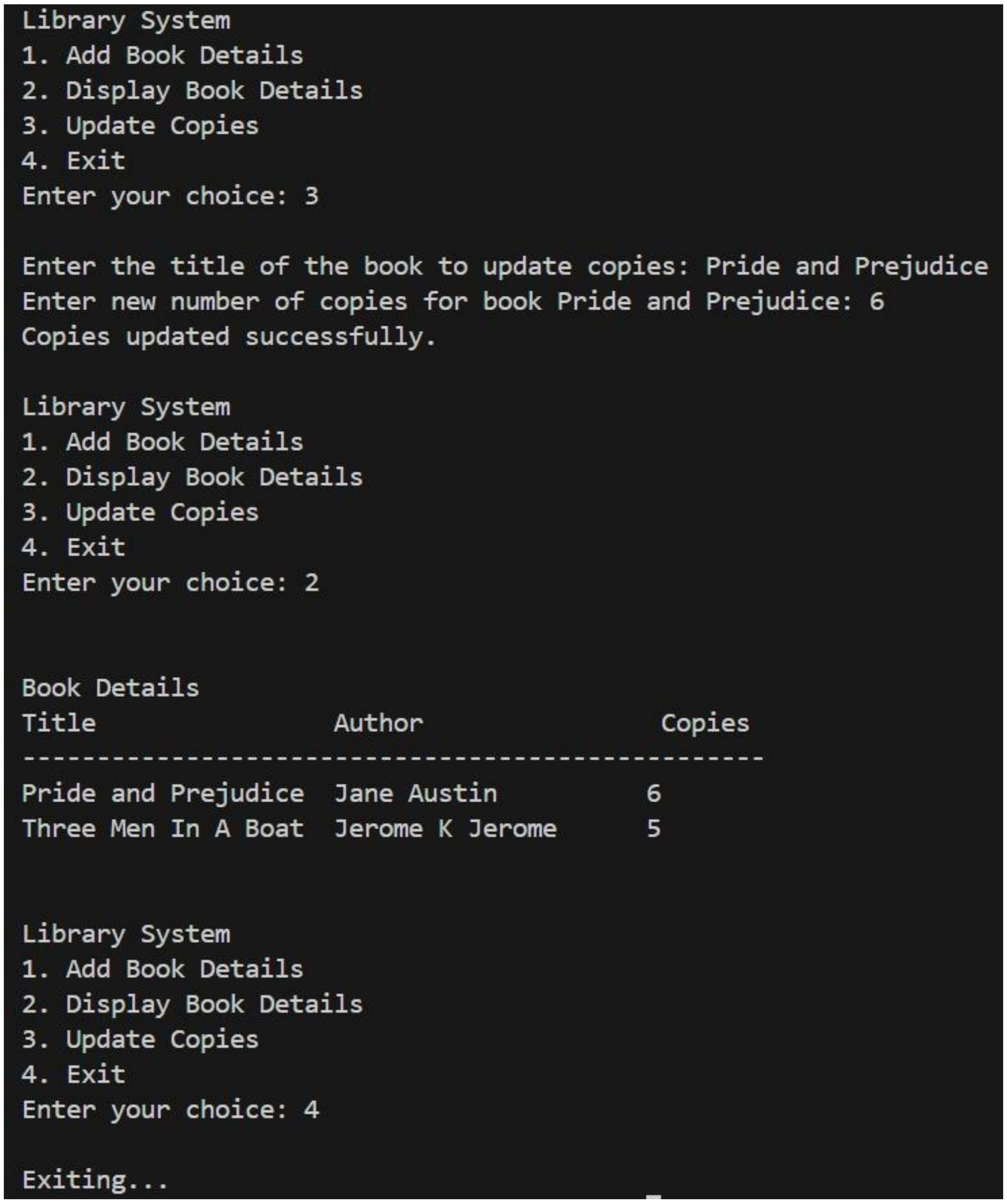
```c
/*Problem 2: Library System with Dynamic Allocation
Objective: Manage a library system where book details are dynamically stored
using pointers inside a structure.
Description:
Define a structure Book with fields:
char *title: Pointer to dynamically allocated memory for the book's title
char *author: Pointer to dynamically allocated memory for the author's name
int *copies: Pointer to the number of available copies (stored dynamically)
Write a program to:
Dynamically allocate memory for n books.
Accept and display book details.
Update the number of copies of a specific book.
Free all allocated memory before exiting.
 */

#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>

struct Book {
    char *title;
    char *author;
    int *copies;
};
void addDetails(struct Book *books, int *noOfBooks);
void displayDetails(struct Book *books, int *noOfBooks);
void updateCopies(struct Book *books, int *noOfBooks, char *title);

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

    struct Book *books = (struct Book *) malloc(noOfBooks * sizeof(struct Book));

    int choice;
    while (1) {
        printf("\nLibrary System\n");
        printf("1. Add Book Details\n");
        printf("2. Display Book Details\n");
        printf("3. Update Copies\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        getchar();
        printf("\n");

        switch (choice) {
            case 1:
                addDetails(books, &noOfBooks);
                break;
            case 2:
                displayDetails(books, &noOfBooks);
                break;
            case 3:
                char userEnteredTitle[100];
                printf("Enter the title of the book to update copies: ");
                fgets(userEnteredTitle, sizeof(userEnteredTitle), stdin);
                userEnteredTitle[strcspn(userEnteredTitle, "\n")] = '\0';
                updateCopies(books, &noOfBooks, userEnteredTitle);
                break;
            case 4:
                printf("Exiting...\n");
                for (int i = 0; i < noOfBooks; i++) {
                    free(books[i].title);
                    free(books[i].author);
                    free(books[i].copies);
                }
                free(books);
                return 0;
            default:
                printf("Invalid choice! Please try again.\n");
        }
    }

    return 0;
}

void addDetails(struct Book *books, int *noOfBooks) {

    for (int i = 0; i < *noOfBooks; i++) {
        books[i].title = (char *) malloc(100 * sizeof(char));
        books[i].author = (char *) malloc(100 * sizeof(char));
        books[i].copies = (int *) malloc(sizeof(int));

        printf("Enter title of book %d: ", i + 1);
        fgets(books[i].title, 100, stdin);
        books[i].title[strcspn(books[i].title, "\n")] = '\0';

        printf("Enter author of book %d: ", i + 1);
        fgets(books[i].author, 100, stdin);
        books[i].author[strcspn(books[i].author, "\n")] = '\0';

        printf("Enter number of copies of book %d: ", i + 1);
        scanf("%d", books[i].copies);
        getchar();

        printf("Book %s added to library\n",books[i].title);
    }
    printf("Details added successfully.\n");
}

void displayDetails(struct Book *books, int *noOfBooks) {
    if(books == NULL) {
        printf("No books added yet.\n");
        return;
    }
    printf("\nBook Details\n");
    printf("Title               Author               Copies\n");
    printf("--------------------------------------------------\n");
    for (int i = 0; i < *noOfBooks; i++) {
        printf("%-20s %-20s %-10d\n", books[i].title, books[i].author,
*(books[i].copies));
    }
    printf("\n");
}

void updateCopies(struct Book *books, int *noOfBooks, char *title) {
    int found = 0;
    for (int i = 0; i < *noOfBooks; i++) {
        if (strcmp(books[i].title, title) == 0) {
            printf("Enter new number of copies for book %s: ", title);
            scanf("%d", books[i].copies);
            getchar();
            printf("Copies updated successfully.\n");
            found = 1;
```



```c
            break;
        }
    }
    if (!found) {
        printf("Book not found.\n");
    }
}
```



```
PS C:\Users\betti\Desktop\Training\Day13> ./task1
Enter the number of books: 2

Library System
1. Add Book Details
2. Display Book Details
3. Update Copies
4. Exit
Enter your choice: 1

Enter title of book 1: Pride and Prejudice
Enter author of book 1: Jane Austin
Enter number of copies of book 1: 4
Book Pride and Prejudice added to library
Enter title of book 2: Three Men In A Boat
Enter author of book 2: Jerome K Jerome
Enter number of copies of book 2: 5
Book Three Men In A Boat added to library
Details added successfully.

Library System
1. Add Book Details
2. Display Book Details
3. Update Copies
4. Exit
Enter your choice: 2


Book Details
Title                Author                Copies
--------------------------------------------------
Pride and Prejudice  Jane Austin          4
Three Men In A Boat  Jerome K Jerome      5
```

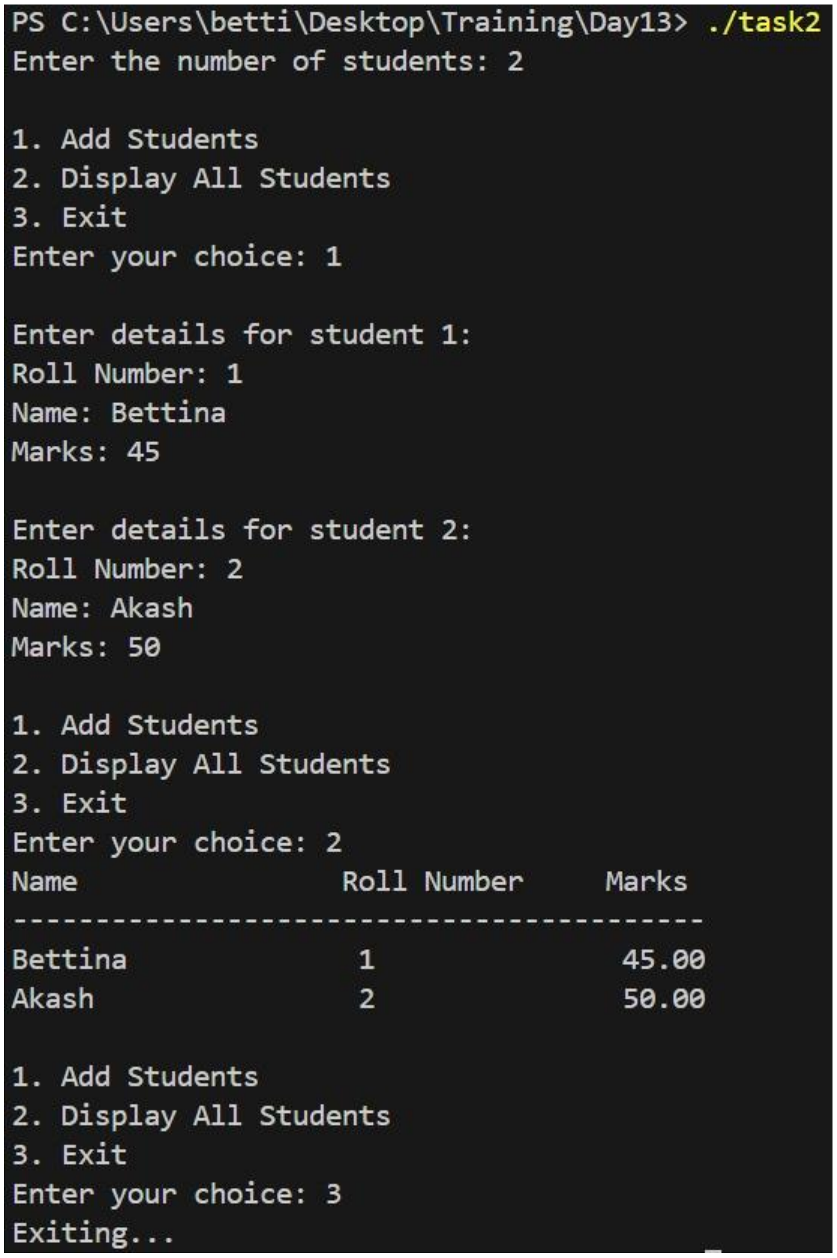


```
Library System
1. Add Book Details
2. Display Book Details
3. Update Copies
4. Exit
Enter your choice: 3

Enter the title of the book to update copies: Pride and Prejudice
Enter new number of copies for book Pride and Prejudice: 6
Copies updated successfully.

Library System
1. Add Book Details
2. Display Book Details
3. Update Copies
4. Exit
Enter your choice: 2


Book Details
Title                Author               Copies
--------------------------------------------------
Pride and Prejudice  Jane Austin         6
Three Men In A Boat  Jerome K Jerome     5


Library System
1. Add Book Details
2. Display Book Details
3. Update Copies
4. Exit
Enter your choice: 4

Exiting...
```

2.

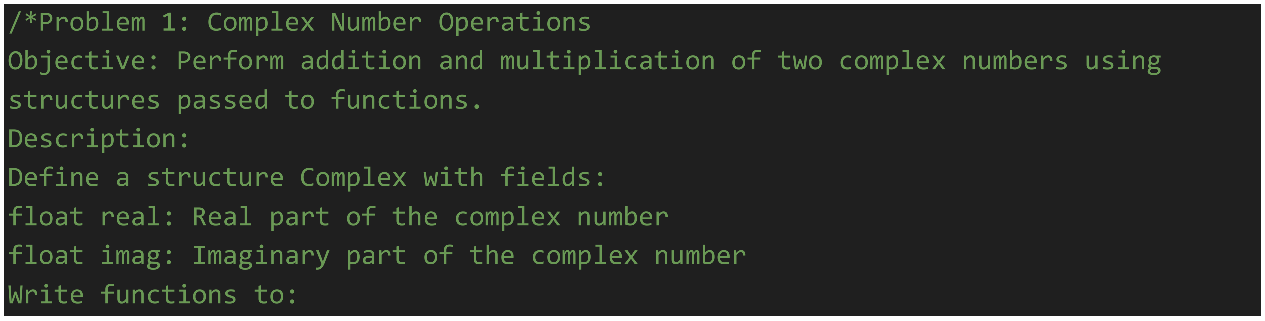
```
/*Problem 1: Dynamic Student Record Management
Objective: Manage student records using pointers to structures and dynamically
allocate memory for student names.
Description:
Define a structure Student with fields:
int roll_no: Roll number
char *name: Pointer to dynamically allocated memory for the student's name
float marks: Marks obtained
```

```c
Write a program to:
Dynamically allocate memory for n students.
Accept details of each student, dynamically allocating memory for their names.
Display all student details.
Free all allocated memory before exiting.*/

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

struct student {
    int rollNumber;
    char *name;
    float marks;
};

void addDetails(struct student *students, int noOfStudents);
void displayAll(struct student *students, int noOfStudents);

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

    struct student *students = (struct student *)malloc(noOfStudents *
sizeof(struct student));
    if (students == NULL) {
    printf("Memory allocation failed for students.\n");
    exit(1);
}

    int choice;
    while(1) {
        printf("\n1. Add Students\n2. Display All Students\n3. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1:
                addDetails(students, noOfStudents);
                break;
            case 2:
                displayAll(students, noOfStudents);
                break;
            case 3:
                printf("Exiting...\n");
                for(int i = 0; i < noOfStudents; i++)
                {
                    free(students[i].name);
                }
                free(students);
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
                break;
        }
    }

    return 0;
}

void addDetails(struct student *students, int noOfStudents) {
    for (int i = 0; i < noOfStudents; i++) {
        students[i].name = (char *)malloc(100 * sizeof(char));
        if (students[i].name == NULL) {
            printf("Memory allocation failed for name.\n");
            exit(1);
        }

        printf("\nEnter details for student %d:\n", i + 1);
        printf("Roll Number: ");
        scanf("%d", &students[i].rollNumber);
        getchar();

        printf("Name: ");
        fgets(students[i].name, 100, stdin);
        students[i].name[strcspn(students[i].name, "\n")] = '\0'; // Remove
trailing newline

        printf("Marks: ");
        scanf("%f", &students[i].marks);
    }
}

void displayAll(struct student *students, int noOfStudents) {
     if(noOfStudents == 0) {
        printf("No students found.\n");
        return;
    }

    printf("Name                Roll Number     Marks\n");
    printf("------------------------------------------\n");

    for (int i = 0; i < noOfStudents; i++) {
        printf("%-20s %-15d %-10.2f\n", students[i].name, students[i].rollNumber,
students[i].marks);
    }
}
```



```
PS C:\Users\betti\Desktop\Training\Day13> ./task2
Enter the number of students: 2

1. Add Students
2. Display All Students
3. Exit
Enter your choice: 1

Enter details for student 1:
Roll Number: 1
Name: Bettina
Marks: 45

Enter details for student 2:
Roll Number: 2
Name: Akash
Marks: 50

1. Add Students
2. Display All Students
3. Exit
Enter your choice: 2
Name                Roll Number     Marks
------------------------------------------
Bettina               1               45.00
Akash                 2               50.00

1. Add Students
2. Display All Students
3. Exit
Enter your choice: 3
Exiting...
```

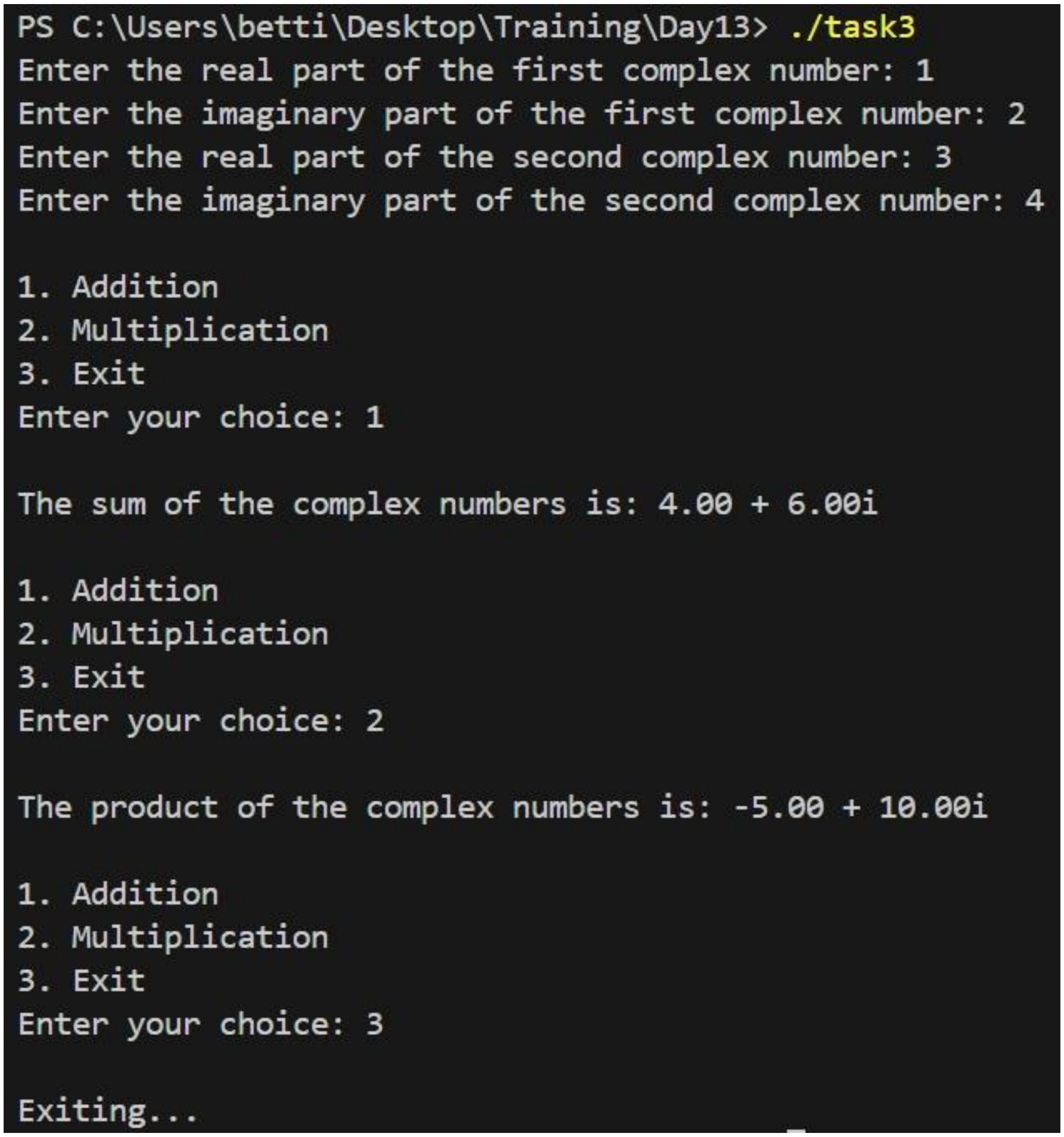
3.



```
/*Problem 1: Complex Number Operations
Objective: Perform addition and multiplication of two complex numbers using
structures passed to functions.
Description:
Define a structure Complex with fields:
float real: Real part of the complex number
float imag: Imaginary part of the complex number
Write functions to:
```

```c
Add two complex numbers and return the result.
Multiply two complex numbers and return the result.
Pass the structures as arguments to these functions and display the results.*/

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

struct Complex {
    float real;
    float imag;
};

struct Complex addComplex(struct Complex, struct Complex);
struct Complex multiplyComplex(struct Complex, struct Complex);

int main() {
    struct Complex num1, num2;
    printf("Enter the real part of the first complex number: ");
    scanf("%f", &num1.real);
    printf("Enter the imaginary part of the first complex number: ");
    scanf("%f", &num1.imag);
    printf("Enter the real part of the second complex number: ");
    scanf("%f", &num2.real);
    printf("Enter the imaginary part of the second complex number: ");
    scanf("%f", &num2.imag);

    int choice;
    while(1) {
        printf("\n1. Addition\n2. Multiplication\n3. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        struct Complex result;
        switch(choice) {
            case 1:
                result = addComplex(num1, num2);
                printf("\nThe sum of the complex numbers is: %.2f + %.2fi\n",
result.real, result.imag);
                break;
            case 2:
                result = multiplyComplex(num1, num2);
                printf("\nThe product of the complex numbers is: %.2f + %.2fi\n",
result.real, result.imag);
```

```c
                break;
            case 3:
                printf("\nExiting...\n");
                exit(0);
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }
    return 0;
}

struct Complex addComplex(struct Complex a, struct Complex b) {
    struct Complex sum;
    sum.real = a.real + b.real;
    sum.imag = a.imag + b.imag;
    return sum;
}

struct Complex multiplyComplex(struct Complex a, struct Complex b) {
    struct Complex product;
    product.real = a.real * b.real - a.imag * b.imag;
    product.imag = a.real * b.imag + a.imag * b.real;
    return product;
}
```



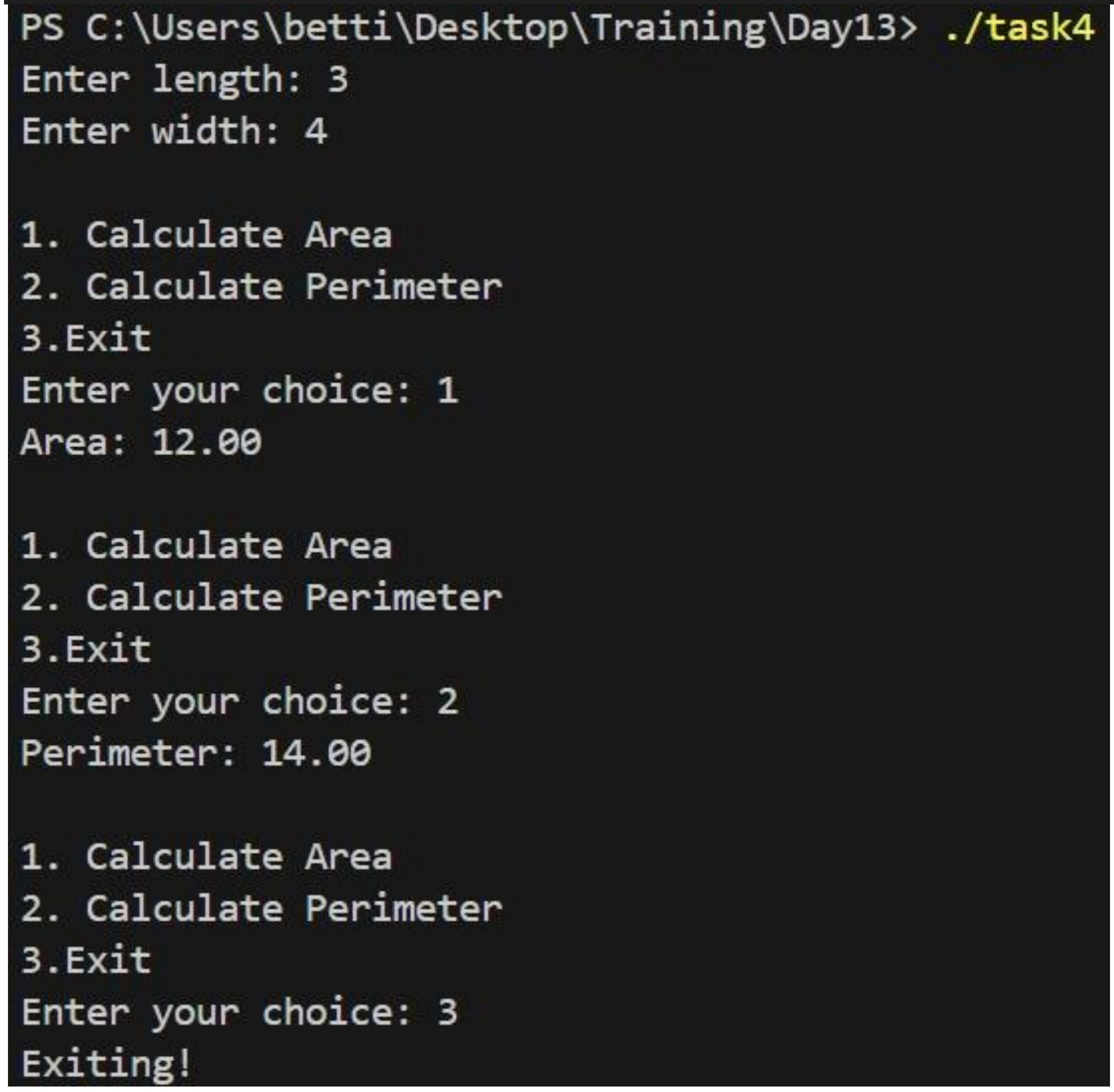
```
PS C:\Users\betti\Desktop\Training\Day13> ./task3
Enter the real part of the first complex number: 1
Enter the imaginary part of the first complex number: 2
Enter the real part of the second complex number: 3
Enter the imaginary part of the second complex number: 4

1. Addition
2. Multiplication
3. Exit
Enter your choice: 1

The sum of the complex numbers is: 4.00 + 6.00i

1. Addition
2. Multiplication
3. Exit
Enter your choice: 2

The product of the complex numbers is: -5.00 + 10.00i

1. Addition
2. Multiplication
3. Exit
Enter your choice: 3

Exiting...
```

4.

```
/*Problem 2: Rectangle Area and Perimeter Calculator
Objective: Calculate the area and perimeter of a rectangle by passing a structure
to functions.
Description:
Define a structure Rectangle with fields:
float length: Length of the rectangle
float width: Width of the rectangle
Write functions to:
```

```
Calculate and return the area of the rectangle.
Calculate and return the perimeter of the rectangle.
Pass the structure to these functions by value and display the results in main.*/

#include <stdio.h>

struct Rectangle {
    float length;
    float width;
};

float rectangleArea(struct Rectangle);
float rectanglePerimeter(struct Rectangle);

int main() {
    struct Rectangle dimensions;
    printf("Enter length: ");
    scanf("%f", &dimensions.length);
    printf("Enter width: ");
    scanf("%f", &dimensions.width);

    float result;
    int choice;

    while(1) {
        printf("\n1. Calculate Area\n2. Calculate Perimeter\n3.Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1:
                result = rectangleArea(dimensions);
                printf("Area: %.2f\n", result);
                break;
            case 2:
                result = rectanglePerimeter(dimensions);
                printf("Perimeter: %.2f\n", result);
                break;
            case 3:
                printf("Exiting!\n");
                return 0;
            default:
                printf("Invalid choice. Please enter 1 or 2.\n");
```

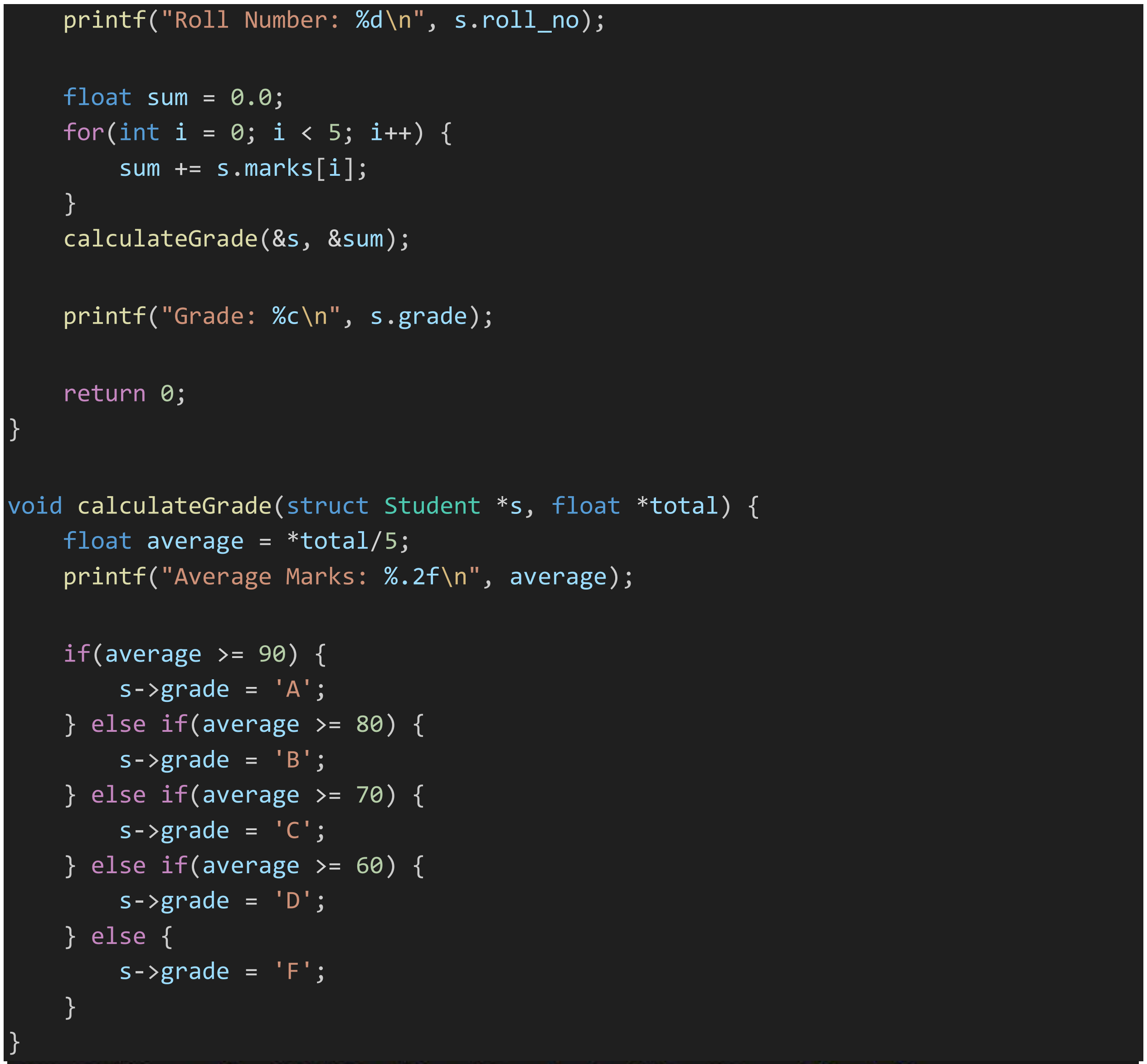
```c
                break;
        }
    }

    return 0;
}

float rectangleArea(struct Rectangle rectangle) {
    return rectangle.length * rectangle.width;
}

float rectanglePerimeter(struct Rectangle rectangle) {
    return 2*(rectangle.length + rectangle.width);
}
```



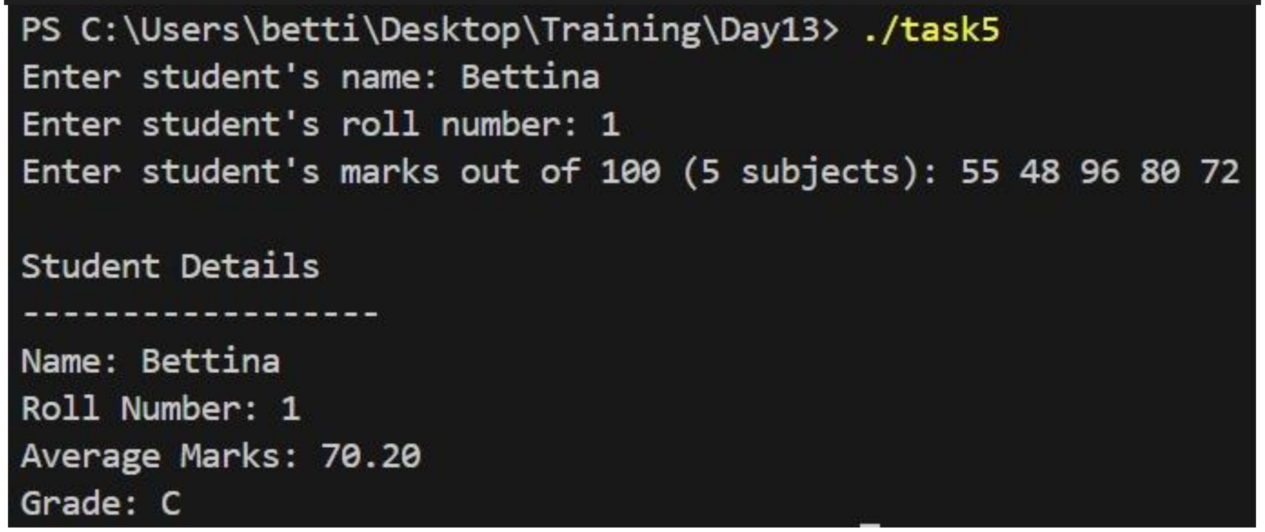
```
PS C:\Users\betti\Desktop\Training\Day13> ./task4
Enter length: 3
Enter width: 4

1. Calculate Area
2. Calculate Perimeter
3.Exit
Enter your choice: 1
Area: 12.00

1. Calculate Area
2. Calculate Perimeter
3.Exit
Enter your choice: 2
Perimeter: 14.00

1. Calculate Area
2. Calculate Perimeter
3.Exit
Enter your choice: 3
Exiting!
```

5.

```c
/*Problem 3: Student Grade Calculation
Objective: Calculate and assign grades to students based on their marks by
passing a structure to a function.
Description:
Define a structure Student with fields:
char name[50]: Name of the student
int roll_no: Roll number
float marks[5]: Marks in 5 subjects
char grade: Grade assigned to the student
Write a function to:
Calculate the average marks and assign a grade (A, B, etc.) based on predefined
criteria.
Pass the structure by reference to the function and modify the grade field.*/

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

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

void calculateGrade(struct Student *, float *);

int main() {
    struct Student s;
    printf("Enter student's name: ");
    scanf("%s", s.name);
    printf("Enter student's roll number: ");
    scanf("%d", &s.roll_no);
    printf("Enter student's marks out of 100 (5 subjects): ");

    for(int i = 0; i < 5; i++) {
        scanf("%f", &s.marks[i]);
    }

    printf("\nStudent Details\n");
    printf("------------------\n");
    printf("Name: %s\n", s.name);
```



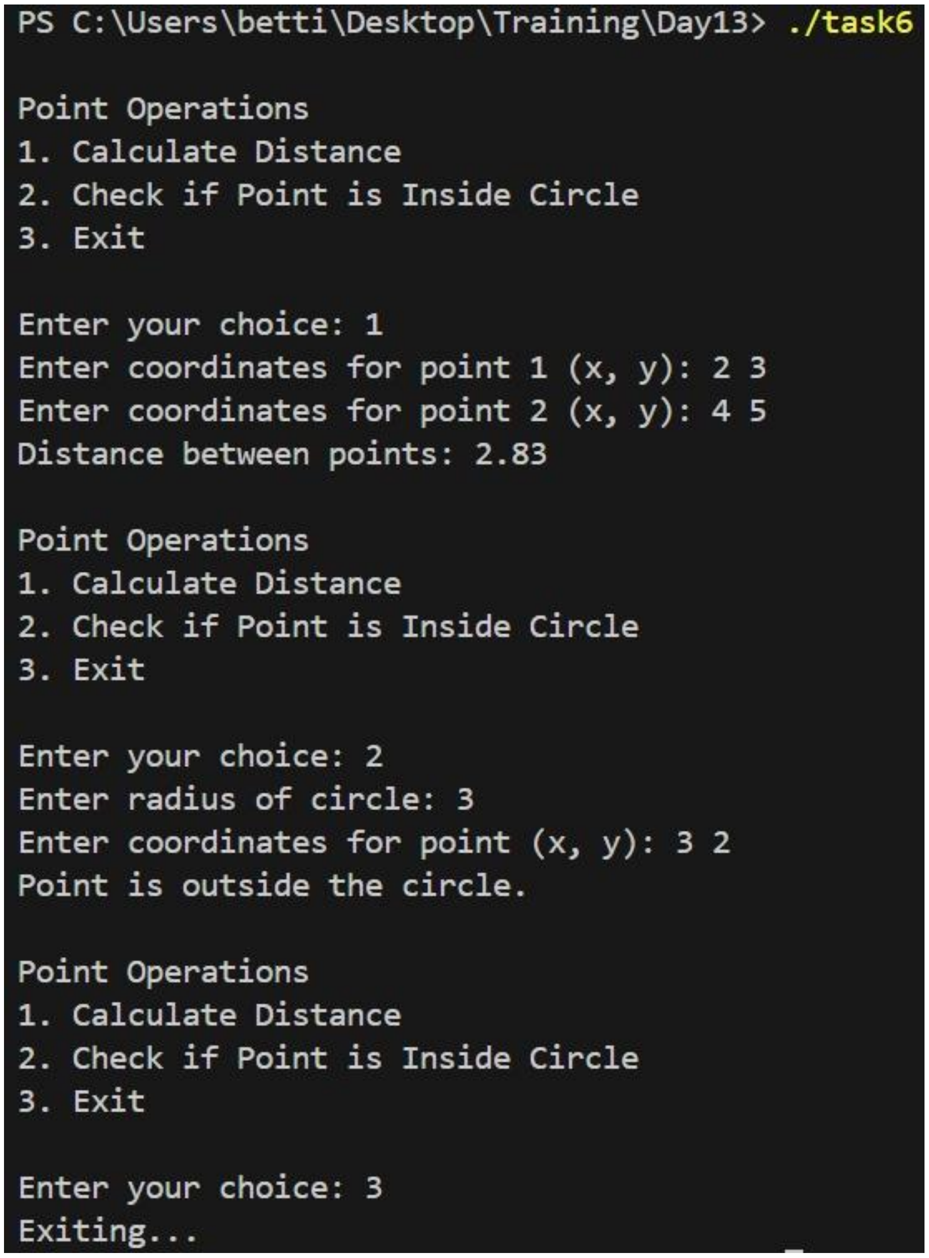
```
    printf("Roll Number: %d\n", s.roll_no);

    float sum = 0.0;
    for(int i = 0; i < 5; i++) {
        sum += s.marks[i];
    }
    calculateGrade(&s, &sum);

    printf("Grade: %c\n", s.grade);

    return 0;
}

void calculateGrade(struct Student *s, float *total) {
    float average = *total/5;
    printf("Average Marks: %.2f\n", average);

    if(average >= 90) {
        s->grade = 'A';
    } else if(average >= 80) {
        s->grade = 'B';
    } else if(average >= 70) {
        s->grade = 'C';
    } else if(average >= 60) {
        s->grade = 'D';
    } else {
        s->grade = 'F';
    }
}
```



```
PS C:\Users\betti\Desktop\Training\Day13> ./task5
Enter student's name: Bettina
Enter student's roll number: 1
Enter student's marks out of 100 (5 subjects): 55 48 96 80 72

Student Details
------------------
Name: Bettina
Roll Number: 1
Average Marks: 70.20
Grade: C
```

6.

```c
/*Problem 4: Point Operations in 2D Space
Objective: Calculate the distance between two points and check if a point lies
within a circle using structures.
Description:
Define a structure Point with fields:
float x: X-coordinate of the point
float y: Y-coordinate of the point
Write functions to:
Calculate the distance between two points.
Check if a given point lies inside a circle of a specified radius (center at
origin).
Pass the Point structure to these functions and display the results.*/

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

struct Point {
    float x;
    float y;
};

float calculateDistance(struct Point, struct Point);
bool isPointInsideCircle(struct Point, float radius);

int main() {
    struct Point p1, p2, p;
    float distance, radius;

    int choice;
    while (1) {
        printf("\nPoint Operations\n");
        printf("1. Calculate Distance\n");
        printf("2. Check if Point is Inside Circle\n");
        printf("3. Exit\n");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter coordinates for point 1 (x, y): ");
                scanf("%f %f", &p1.x, &p1.y);
                printf("Enter coordinates for point 2 (x, y): ");
                scanf("%f %f", &p2.x, &p2.y);
                distance = calculateDistance(p1, p2);
                printf("Distance between points: %.2f\n", distance);
                break;
            case 2:
                printf("Enter radius of circle: ");
                scanf("%f", &radius);
                printf("Enter coordinates for point (x, y): ");
                scanf("%f %f", &p.x, &p.y);
                if (isPointInsideCircle(p, radius)) {
                    printf("Point is inside the circle.\n");
                } else {
                    printf("Point is outside the circle.\n");
                }
                break;
            case 3:
                printf("Exiting...\n");
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }
    return 0;
}

float calculateDistance(struct Point a, struct Point b) {
    float dx = a.x - b.x;
    float dy = a.y - b.y;
    return sqrt(dx * dx + dy * dy);
}

bool isPointInsideCircle(struct Point c, float r) {
    return (c.x * c.x + c.y * c.y <= r * r);
}
```

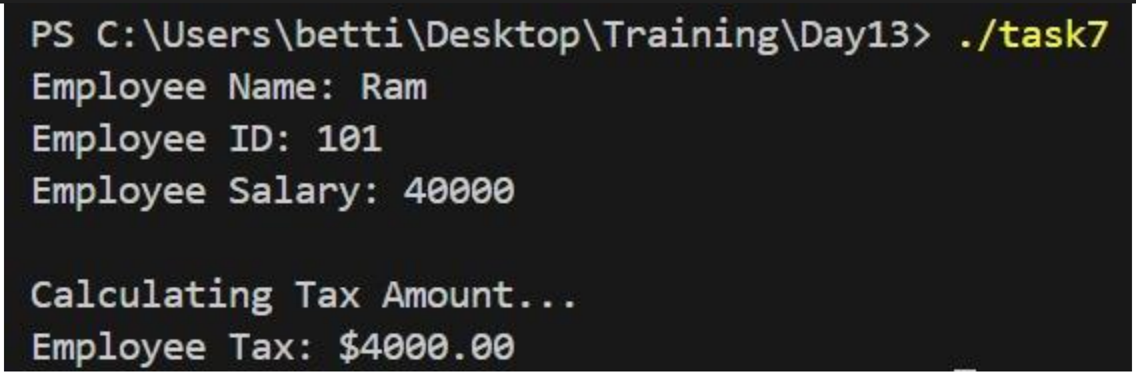


```
PS C:\Users\betti\Desktop\Training\Day13> ./task6

Point Operations
1. Calculate Distance
2. Check if Point is Inside Circle
3. Exit

Enter your choice: 1
Enter coordinates for point 1 (x, y): 2 3
Enter coordinates for point 2 (x, y): 4 5
Distance between points: 2.83

Point Operations
1. Calculate Distance
2. Check if Point is Inside Circle
3. Exit

Enter your choice: 2
Enter radius of circle: 3
Enter coordinates for point (x, y): 3 2
Point is outside the circle.

Point Operations
1. Calculate Distance
2. Check if Point is Inside Circle
3. Exit

Enter your choice: 3
Exiting...
```

7.

```
/*Problem 5: Employee Tax Calculation
Objective: Calculate income tax for an employee based on their salary by passing
a structure to a function.
Description:
Define a structure Employee with fields:
char name[50]: Employee name
int emp_id: Employee ID
float salary: Employee salary
```

```c
float tax: Tax to be calculated (initialized to 0)
Write a function to:
Calculate tax based on salary slabs (e.g., 10% for salaries below $50,000, 20%
otherwise).
Modify the tax field of the structure.
Pass the structure by reference to the function and display the updated tax in
main.*/

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

struct Employee {
    char name[50];
    int emp_id;
    float salary;
    float tax;
};

float calculateTax(struct Employee*);

int main() {
    struct Employee employee;
    printf("Employee Name: ");
    scanf("%s", employee.name);
    printf("Employee ID: ");
    scanf("%d", &employee.emp_id);
    printf("Employee Salary: ");
    scanf("%f", &employee.salary);
    if (employee.salary <= 0) {
        printf("Invalid salary. It should be a positive number.\n");
        return 1;
    }

    employee.tax = 0;
    printf("\nCalculating Tax Amount...\n");

    float employeeTax = calculateTax(&employee);
    printf("Employee Tax: $%.2f\n", employeeTax);
    return 0;
}

float calculateTax(struct Employee* emp) {
    float empTax;
    if (emp->salary <= 50000) {
        empTax = emp->salary * 0.10;
    } else {
        empTax = emp->salary * 0.20;
    }
    emp->tax = empTax;
    return empTax;
}
```



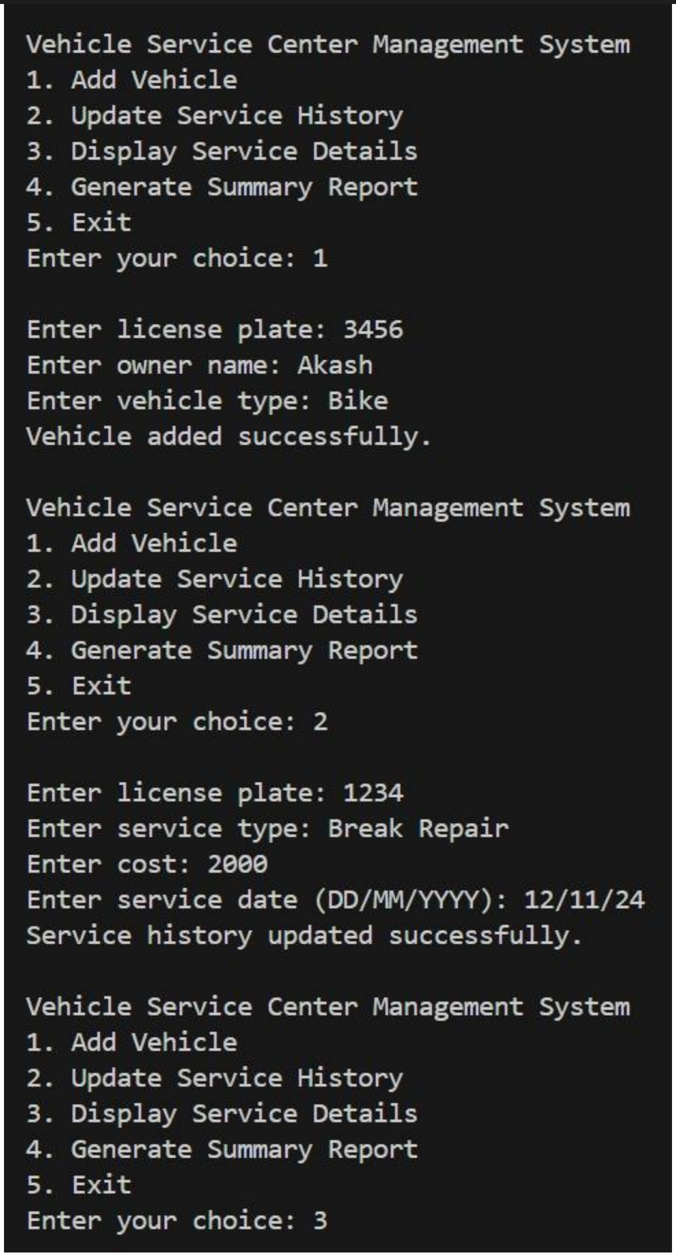
```
PS C:\Users\betti\Desktop\Training\Day13> ./task7
Employee Name: Ram
Employee ID: 101
Employee Salary: 40000

Calculating Tax Amount...
Employee Tax: $4000.00
```

8.

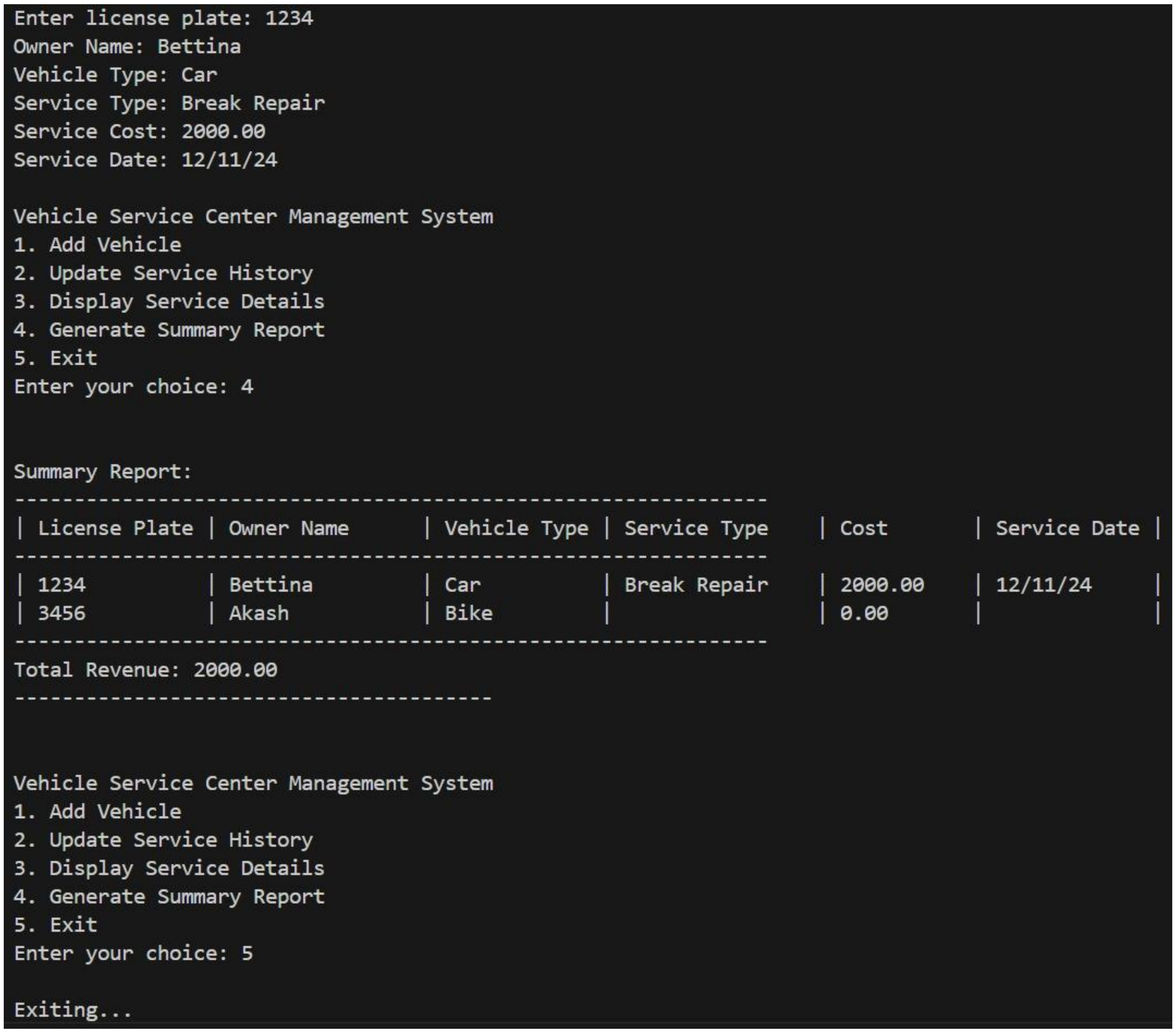
```c
/*Problem Statement: Vehicle Service Center Management
Objective: Build a system to manage vehicle servicing records using nested
structures.
Description:
Define a structure Vehicle with fields:
char license_plate[15]: Vehicle's license plate number
char owner_name[50]: Owner's name
char vehicle_type[20]: Type of vehicle (e.g., car, bike)
Define a nested structure Service inside Vehicle with fields:
char service_type[30]: Type of service performed
float cost: Cost of the service
char service_date[12]: Date of service
Implement the following features:
Add a vehicle to the service center record.
Update the service history for a vehicle.
Display the service details of a specific vehicle.
Generate and display a summary report of all vehicles serviced, including total
revenue.*/


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

```c
struct Vehicle {
    char license_plate[15];
    char owner_name[50];
    char vehicle_type[20];
    struct Service {
        char service_type[30];
        float cost;
        char service_date[12];
    } service;
};

void add_vehicle(struct Vehicle*, int*);
void update_service_history(struct Vehicle*, int);
void display_service_details(struct Vehicle*, int);
void generate_report(struct Vehicle*, int);

int main() {
    struct Vehicle vehicles[100];
    int no_of_vehicles = 0;

    int choice = 0;
    while(1) {
        printf("\nVehicle Service Center Management System\n");
        printf("1. Add Vehicle\n");
        printf("2. Update Service History\n");
        printf("3. Display Service Details\n");
        printf("4. Generate Summary Report\n");
        printf("5. Exit\n");

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

        printf("\n");

        switch(choice) {
            case 1:
                add_vehicle(vehicles, &no_of_vehicles);
                break;
            case 2:
                update_service_history(vehicles, no_of_vehicles);
                break;
            case 3:
                display_service_details(vehicles, no_of_vehicles);
                break;
            case 4:
                generate_report(vehicles, no_of_vehicles);
                break;
            case 5:
                printf("Exiting...\n");
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
                break;
        }
    }
    return 0;
}

void add_vehicle(struct Vehicle* vehicles, int* no_of_vehicles) {
    printf("Enter license plate: ");
    scanf("%s", vehicles[*no_of_vehicles].license_plate);

    printf("Enter owner name: ");
    scanf(" %[^\n]s", vehicles[*no_of_vehicles].owner_name);

    printf("Enter vehicle type: ");
    scanf("%s", vehicles[*no_of_vehicles].vehicle_type);

    strcpy(vehicles[*no_of_vehicles].service.service_type, "");
    vehicles[*no_of_vehicles].service.cost = 0.0;
    strcpy(vehicles[*no_of_vehicles].service.service_date, "");

    (*no_of_vehicles)++;
    printf("Vehicle added successfully.\n");
}

void update_service_history(struct Vehicle* vehicles, int no_of_vehicles) {
    if (no_of_vehicles == 0) {
        printf("No vehicles available.\n");
        return;
    }

    char license_plate[15];
    printf("Enter license plate: ");
    scanf("%s", license_plate);
    getchar();

    int found = 0;
    for (int i = 0; i < no_of_vehicles; i++) {
        if (strcmp(vehicles[i].license_plate, license_plate) == 0) {
            printf("Enter service type: ");
            scanf("%[^\n]", vehicles[i].service.service_type);

            getchar();

            printf("Enter cost: ");
            scanf("%f", &vehicles[i].service.cost);

            printf("Enter service date (DD/MM/YYYY): ");
            scanf("%s", vehicles[i].service.service_date);

            found = 1;
            printf("Service history updated successfully.\n");
            break;
        }
    }

    if (!found) {
        printf("Vehicle with license plate %s not found.\n", license_plate);
    }
}

void display_service_details(struct Vehicle* vehicles, int no_of_vehicles) {
    if (no_of_vehicles == 0) {
        printf("No vehicles available.\n");
        return;
    }

    char license_plate[15];
    printf("Enter license plate: ");
    scanf("%s", license_plate);

    int found = 0;
    for (int i = 0; i < no_of_vehicles; i++) {
        if (strcmp(vehicles[i].license_plate, license_plate) == 0) {
            printf("Owner Name: %s\n", vehicles[i].owner_name);
            printf("Vehicle Type: %s\n", vehicles[i].vehicle_type);
            if (strcmp(vehicles[i].service.service_type, "") == 0) {
                printf("No service history available for this vehicle.\n");
            } else {
                printf("Service Type: %s\n", vehicles[i].service.service_type);
                printf("Service Cost: %.2f\n", vehicles[i].service.cost);
                printf("Service Date: %s\n", vehicles[i].service.service_date);
            }
            found = 1;
            break;
        }
    }

    if (!found) {
        printf("Vehicle with license plate %s not found.\n", license_plate);
    }
}

void generate_report(struct Vehicle* vehicles, int no_of_vehicles) {
    if (no_of_vehicles == 0) {
        printf("No vehicles available.\n");
        return;
    }

    float total_revenue = 0.0;
    printf("\nSummary Report:\n");
    printf("---------------------------------------------------------------\n");
    printf("| %-13s | %-15s | %-12s | %-15s | %-10s | %-12s |\n",
           "License Plate", "Owner Name", "Vehicle Type",
           "Service Type", "Cost", "Service Date");
    printf("---------------------------------------------------------------\n");
    for (int i = 0; i < no_of_vehicles; i++) {
        printf("| %-13s | %-15s | %-12s | %-15s | %-10.2f | %-12s |\n",
               vehicles[i].license_plate, vehicles[i].owner_name,
               vehicles[i].vehicle_type, vehicles[i].service.service_type,
               vehicles[i].service.cost, vehicles[i].service.service_date);
        total_revenue += vehicles[i].service.cost;
    }
    printf("---------------------------------------------------------------\n");
    printf("Total Revenue: %.2f\n", total_revenue);
    printf("----------------------------------------\n");
    printf("\n");
    return;
```

```
}
```



```
Vehicle Service Center Management System
1. Add Vehicle
2. Update Service History
3. Display Service Details
4. Generate Summary Report
5. Exit
Enter your choice: 1

Enter license plate: 3456
Enter owner name: Akash
Enter vehicle type: Bike
Vehicle added successfully.

Vehicle Service Center Management System
1. Add Vehicle
2. Update Service History
3. Display Service Details
4. Generate Summary Report
5. Exit
Enter your choice: 2

Enter license plate: 1234
Enter service type: Break Repair
Enter cost: 2000
Enter service date (DD/MM/YYYY): 12/11/24
Service history updated successfully.

Vehicle Service Center Management System
1. Add Vehicle
2. Update Service History
3. Display Service Details
4. Generate Summary Report
5. Exit
Enter your choice: 3
```



```
Enter license plate: 1234
Owner Name: Bettina
Vehicle Type: Car
Service Type: Break Repair
Service Cost: 2000.00
Service Date: 12/11/24

Vehicle Service Center Management System
1. Add Vehicle
2. Update Service History
3. Display Service Details
4. Generate Summary Report
5. Exit
Enter your choice: 4


Summary Report:
---------------------------------------------------------------
| License Plate | Owner Name       | Vehicle Type | Service Type    | Cost        | Service Date |
---------------------------------------------------------------
| 1234          | Bettina          | Car          | Break Repair    | 2000.00     | 12/11/24     |
| 3456          | Akash            | Bike         |                 | 0.00        |              |
---------------------------------------------------------------
Total Revenue: 2000.00
----------------------------------------


Vehicle Service Center Management System
1. Add Vehicle
2. Update Service History
3. Display Service Details
4. Generate Summary Report
5. Exit
Enter your choice: 5

Exiting...
```