Course Title:	Programming Fundamentals Lab (CL1002)
Assignment Title:	Lab Manual 11 tasks
Submitted to:	Sir Sandesh Kumar
Name:	Muhammad Usman Khan
Roll No:	25K-2038 BCY-1A
Date:	20 November 2025

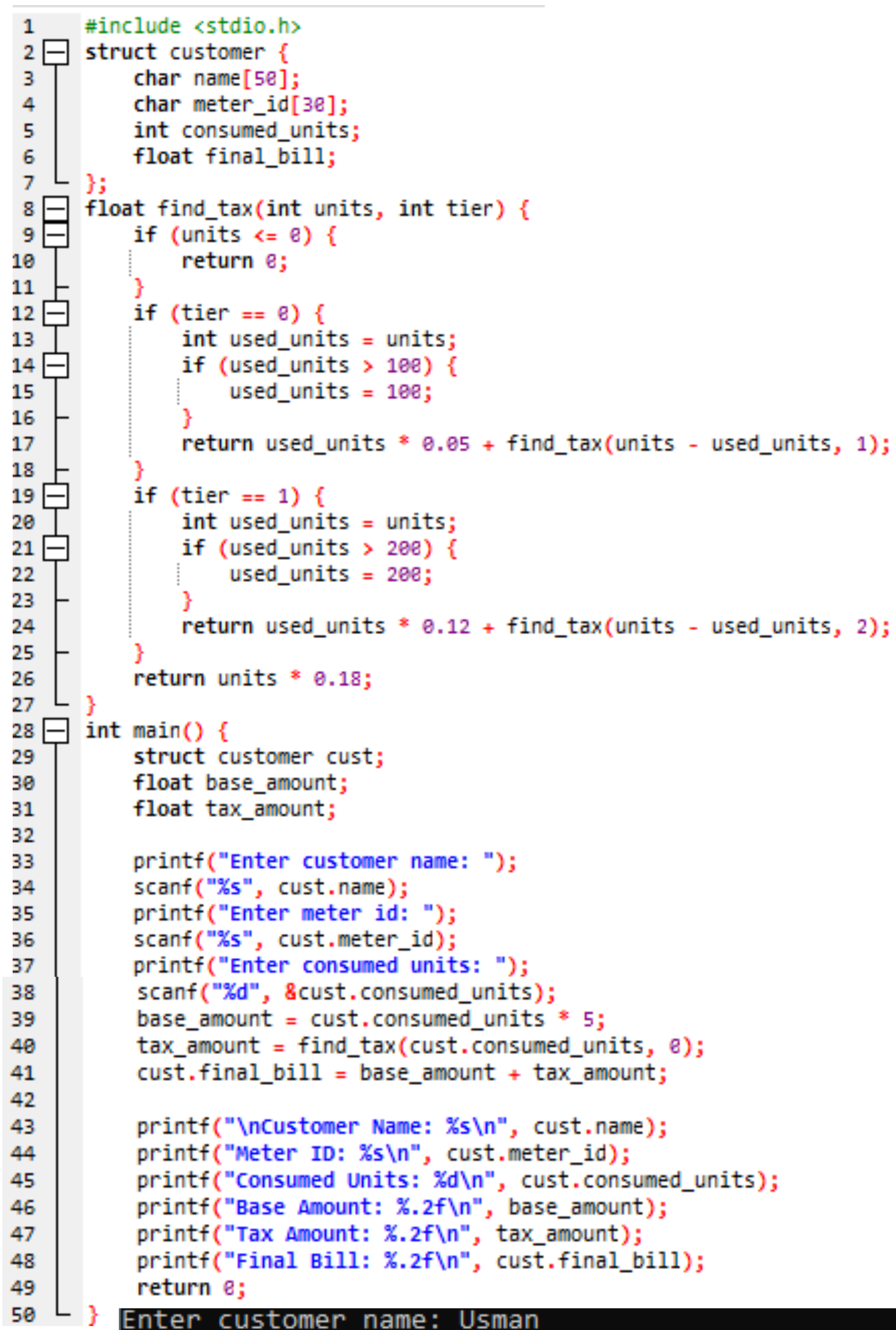
LAB EXERCISES [6 Marks]

1. Design a Smart Electricity Billing Using Tiered Recursive Tax Calculation where a power company uses tiered billing, where the tax increases as consumption increases (e.g., first 100 units → 5%, next 200 units → 12%, above that → 18%) by writing a program using recursion to calculate the total payable amount for a customer. Use a customer structure storing name, meter-ID, consumed units, and final bill. Implement a recursive function that computes tax tier-by-tier until all units are processed.

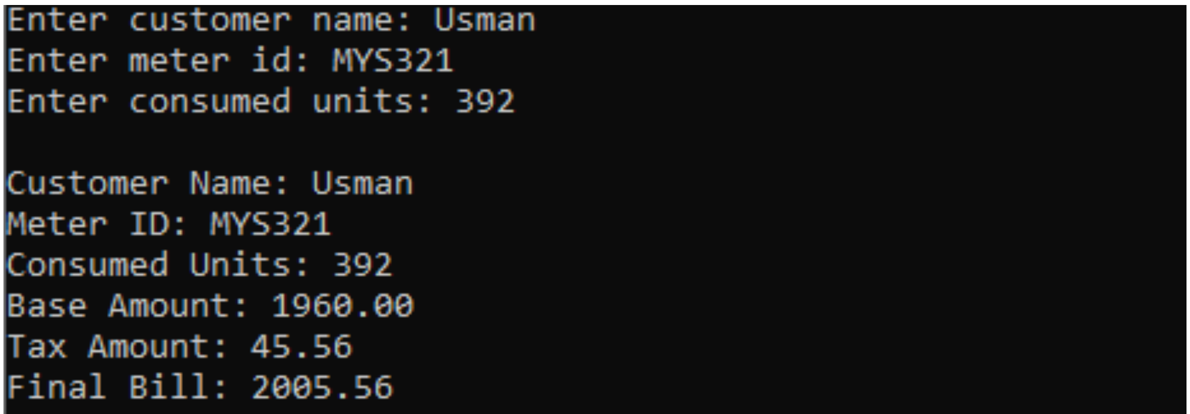
2. Design a Recursive Route Cost Estimation for a Delivery Company where a delivery service calculates the cost of a route based on distance segments (e.g., every 10 km adds a special charge based on road type). Create a recursive function that calculates total delivery cost segment-by-segment. Use a structure Route containing: routeID, distance, basePrice, and roadType. Pointers should be used to update the final cost.

3. Design an Inventory Expiry Monitoring using Nested Structures & Recursion where a store has products, each belonging to a category. Use nested structures:
   Category → categoryName, department
   Product → productName, expiryDays, category (nested)
   Write a recursive function that determines which products expire within the next N days. Use recursion to process one product at a time and print alerts.

Question 1

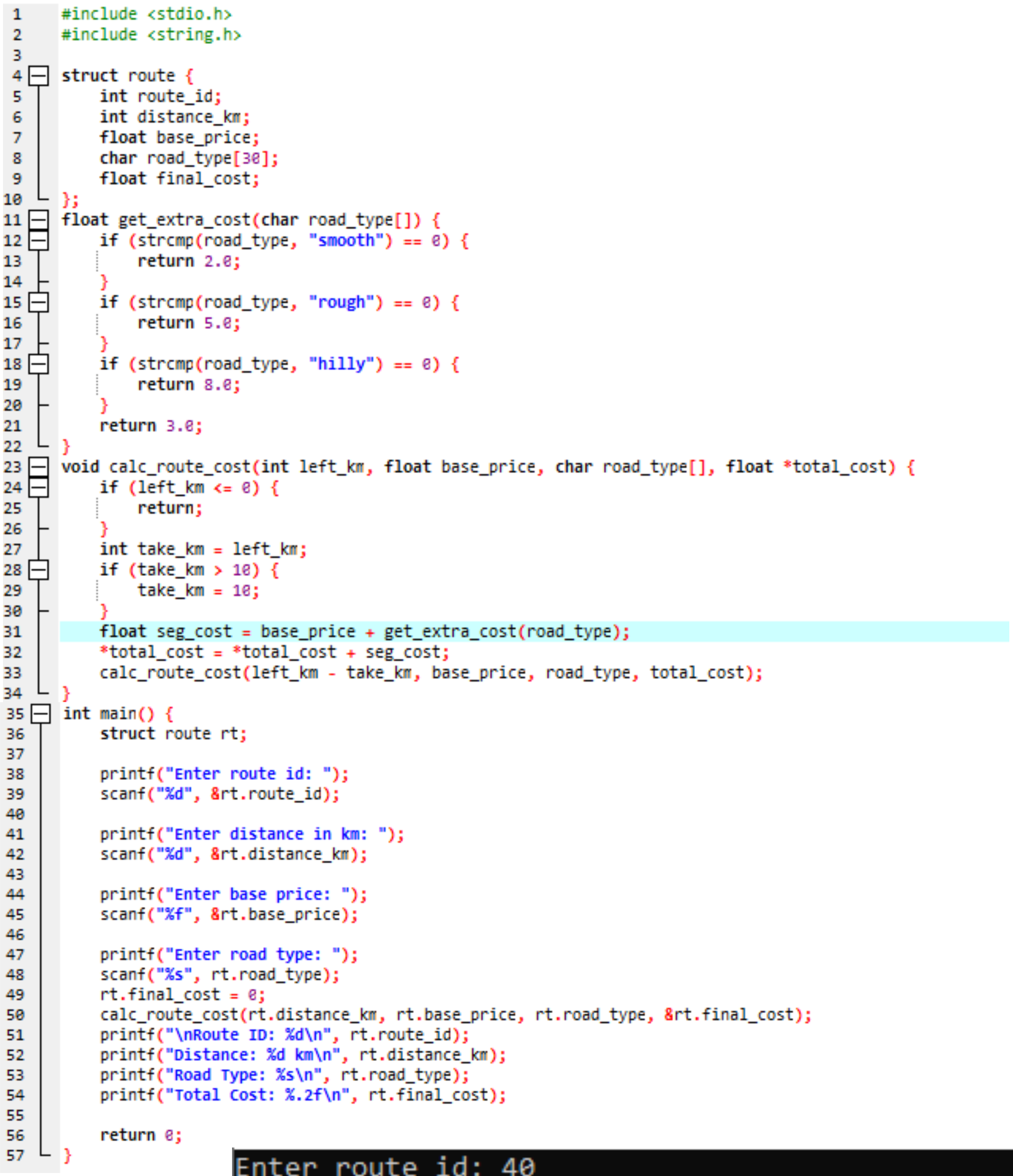


```c
#include <stdio.h>
struct customer {
    char name[50];
    char meter_id[30];
    int consumed_units;
    float final_bill;
};
float find_tax(int units, int tier) {
    if (units <= 0) {
        return 0;
    }
    if (tier == 0) {
        int used_units = units;
        if (used_units > 100) {
            used_units = 100;
        }
        return used_units * 0.05 + find_tax(units - used_units, 1);
    }
    if (tier == 1) {
        int used_units = units;
        if (used_units > 200) {
            used_units = 200;
        }
        return used_units * 0.12 + find_tax(units - used_units, 2);
    }
    return units * 0.18;
}
int main() {
    struct customer cust;
    float base_amount;
    float tax_amount;

    printf("Enter customer name: ");
    scanf("%s", cust.name);
    printf("Enter meter id: ");
    scanf("%s", cust.meter_id);
    printf("Enter consumed units: ");
    scanf("%d", &cust.consumed_units);
    base_amount = cust.consumed_units * 5;
    tax_amount = find_tax(cust.consumed_units, 0);
    cust.final_bill = base_amount + tax_amount;

    printf("\nCustomer Name: %s\n", cust.name);
    printf("Meter ID: %s\n", cust.meter_id);
    printf("Consumed Units: %d\n", cust.consumed_units);
    printf("Base Amount: %.2f\n", base_amount);
    printf("Tax Amount: %.2f\n", tax_amount);
    printf("Final Bill: %.2f\n", cust.final_bill);
    return 0;
}
```

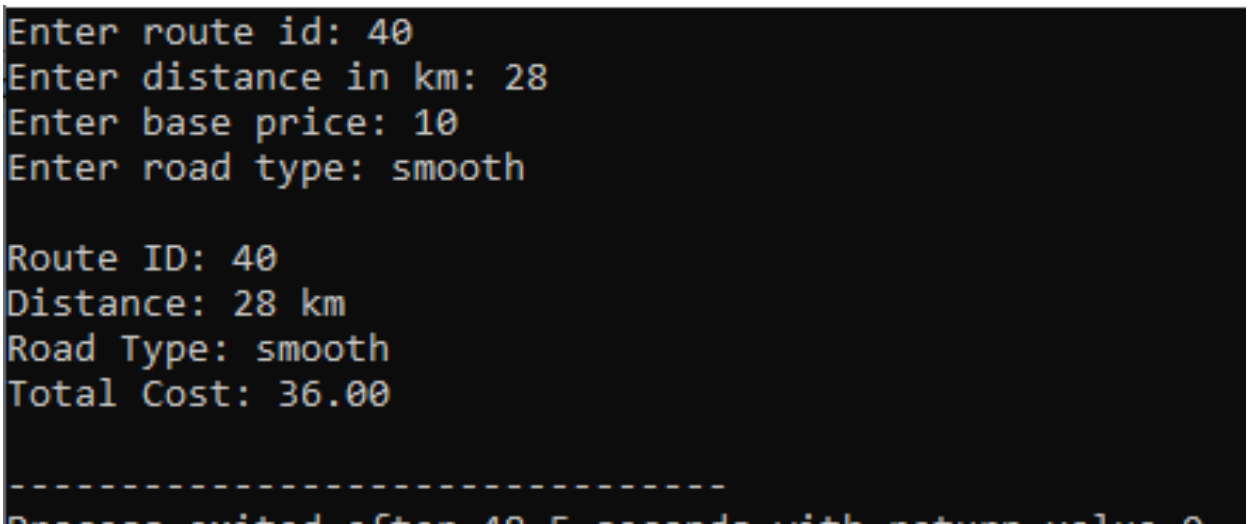


```
Enter customer name: Usman
Enter meter id: MYS321
Enter consumed units: 392

Customer Name: Usman
Meter ID: MYS321
Consumed Units: 392
Base Amount: 1960.00
Tax Amount: 45.56
Final Bill: 2005.56
```

Question 2

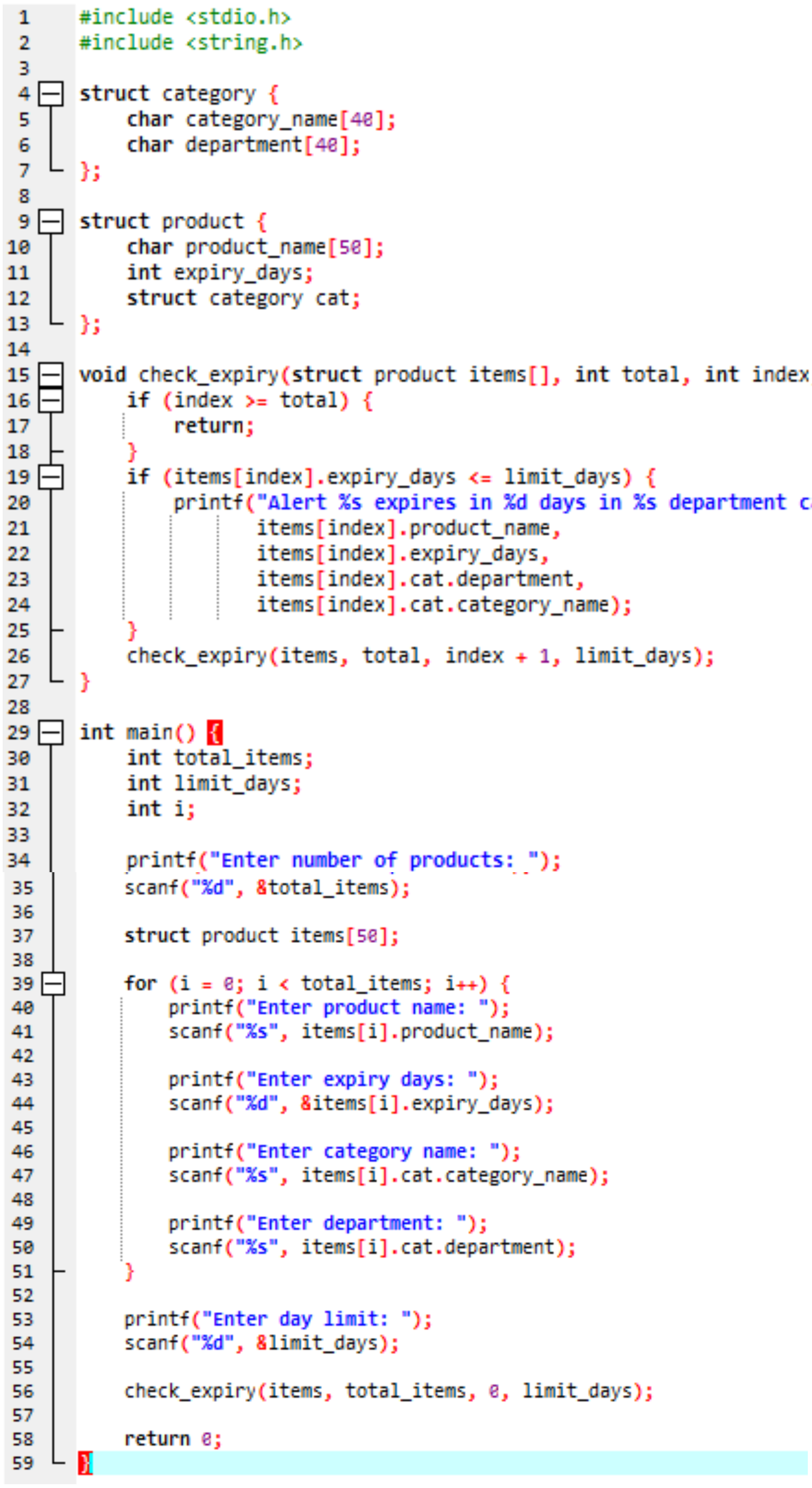


```c
#include <stdio.h>
#include <string.h>

struct route {
    int route_id;
    int distance_km;
    float base_price;
    char road_type[30];
    float final_cost;
};
float get_extra_cost(char road_type[]) {
    if (strcmp(road_type, "smooth") == 0) {
        return 2.0;
    }
    if (strcmp(road_type, "rough") == 0) {
        return 5.0;
    }
    if (strcmp(road_type, "hilly") == 0) {
        return 8.0;
    }
    return 3.0;
}
void calc_route_cost(int left_km, float base_price, char road_type[], float *total_cost) {
    if (left_km <= 0) {
        return;
    }
    int take_km = left_km;
    if (take_km > 10) {
        take_km = 10;
    }
    float seg_cost = base_price + get_extra_cost(road_type);
    *total_cost = *total_cost + seg_cost;
    calc_route_cost(left_km - take_km, base_price, road_type, total_cost);
}
int main() {
    struct route rt;

    printf("Enter route id: ");
    scanf("%d", &rt.route_id);

    printf("Enter distance in km: ");
    scanf("%d", &rt.distance_km);

    printf("Enter base price: ");
    scanf("%f", &rt.base_price);

    printf("Enter road type: ");
    scanf("%s", rt.road_type);
    rt.final_cost = 0;
    calc_route_cost(rt.distance_km, rt.base_price, rt.road_type, &rt.final_cost);
    printf("\nRoute ID: %d\n", rt.route_id);
    printf("Distance: %d km\n", rt.distance_km);
    printf("Road Type: %s\n", rt.road_type);
    printf("Total Cost: %.2f\n", rt.final_cost);

    return 0;
}
```

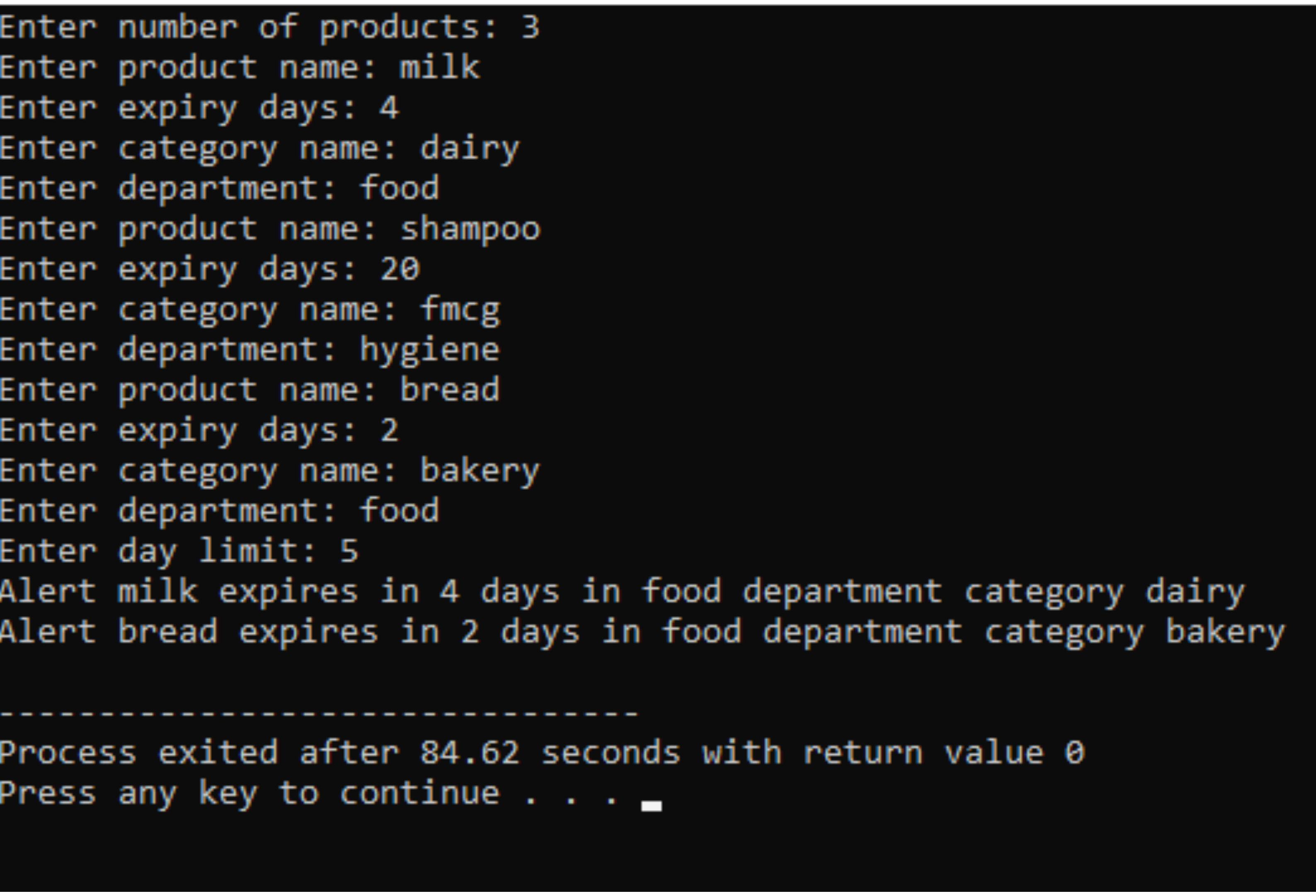


```
Enter route id: 40
Enter distance in km: 28
Enter base price: 10
Enter road type: smooth

Route ID: 40
Distance: 28 km
Road Type: smooth
Total Cost: 36.00

--------------------------------
Process exited after 40.5 seconds with return value 0
```

Question 3



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

struct category {
    char category_name[40];
    char department[40];
};

struct product {
    char product_name[50];
    int expiry_days;
    struct category cat;
};

void check_expiry(struct product items[], int total, int index, int limit_days) {
    if (index >= total) {
        return;
    }
    if (items[index].expiry_days <= limit_days) {
        printf("Alert %s expires in %d days in %s department category %s\n",
               items[index].product_name,
               items[index].expiry_days,
               items[index].cat.department,
               items[index].cat.category_name);
    }
    check_expiry(items, total, index + 1, limit_days);
}

int main() {
    int total_items;
    int limit_days;
    int i;

    printf("Enter number of products: ");
    scanf("%d", &total_items);

    struct product items[50];

    for (i = 0; i < total_items; i++) {
        printf("Enter product name: ");
        scanf("%s", items[i].product_name);

        printf("Enter expiry days: ");
        scanf("%d", &items[i].expiry_days);

        printf("Enter category name: ");
        scanf("%s", items[i].cat.category_name);

        printf("Enter department: ");
        scanf("%s", items[i].cat.department);
    }

    printf("Enter day limit: ");
    scanf("%d", &limit_days);

    check_expiry(items, total_items, 0, limit_days);

    return 0;
}
```



```
Enter number of products: 3
Enter product name: milk
Enter expiry days: 4
Enter category name: dairy
Enter department: food
Enter product name: shampoo
Enter expiry days: 20
Enter category name: fmcg
Enter department: hygiene
Enter product name: bread
Enter expiry days: 2
Enter category name: bakery
Enter department: food
Enter day limit: 5
Alert milk expires in 4 days in food department category dairy
Alert bread expires in 2 days in food department category bakery

--------------------------------
Process exited after 84.62 seconds with return value 0
Press any key to continue . . .
```