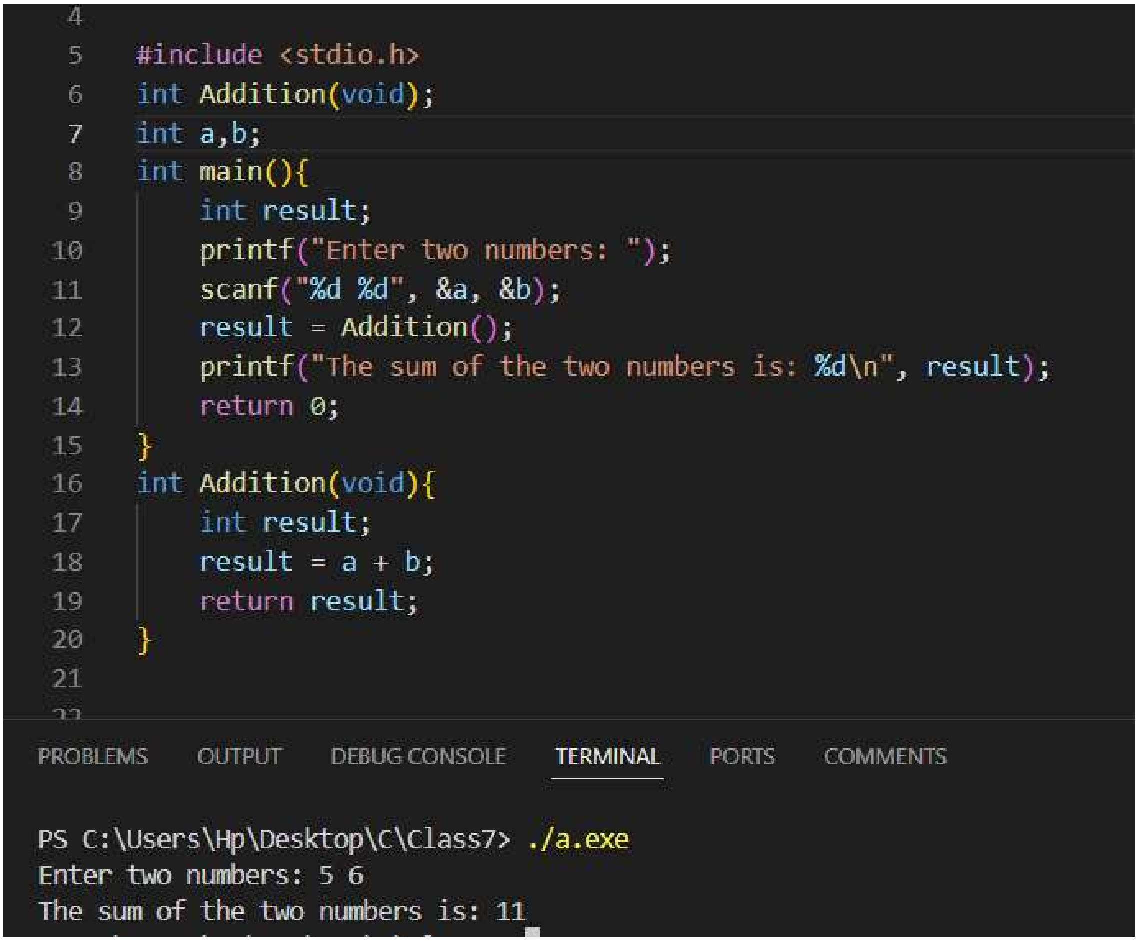
1)Sum of Two Numbers

Write a program that takes two integers as input and calculates their sum using a function.

Pass the integers to the function using call by value.

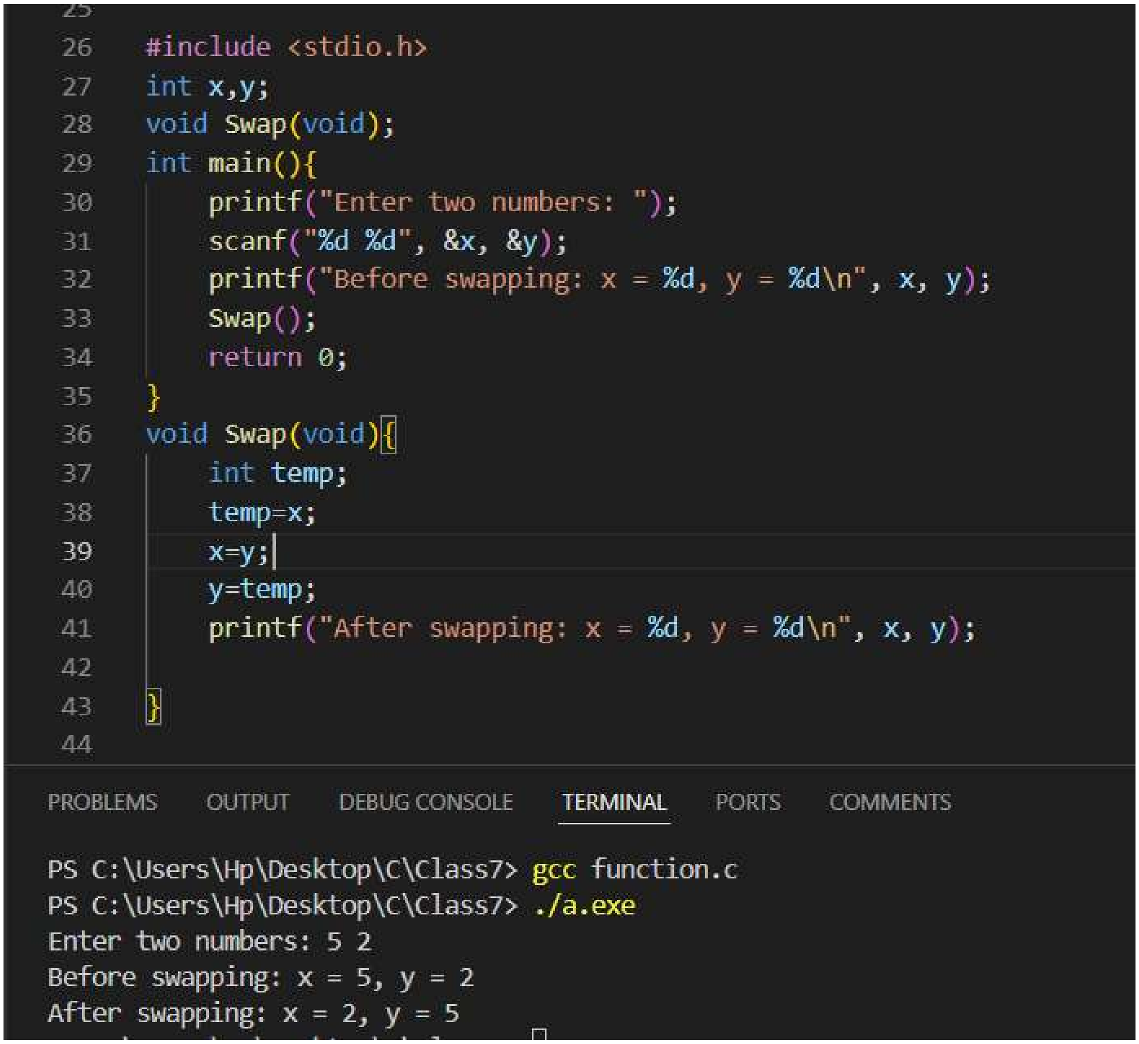


```c
#include <stdio.h>
int Addition(void);
int a,b;
int main(){
    int result;
    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);
    result = Addition();
    printf("The sum of the two numbers is: %d\n", result);
    return 0;
}
int Addition(void){
    int result;
    result = a + b;
    return result;
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter two numbers: 5 6
The sum of the two numbers is: 11
```

2)Swap Two Numbers

Write a program to swap two numbers using a function. Observe and explain why the original numbers remain unchanged due to call by value.

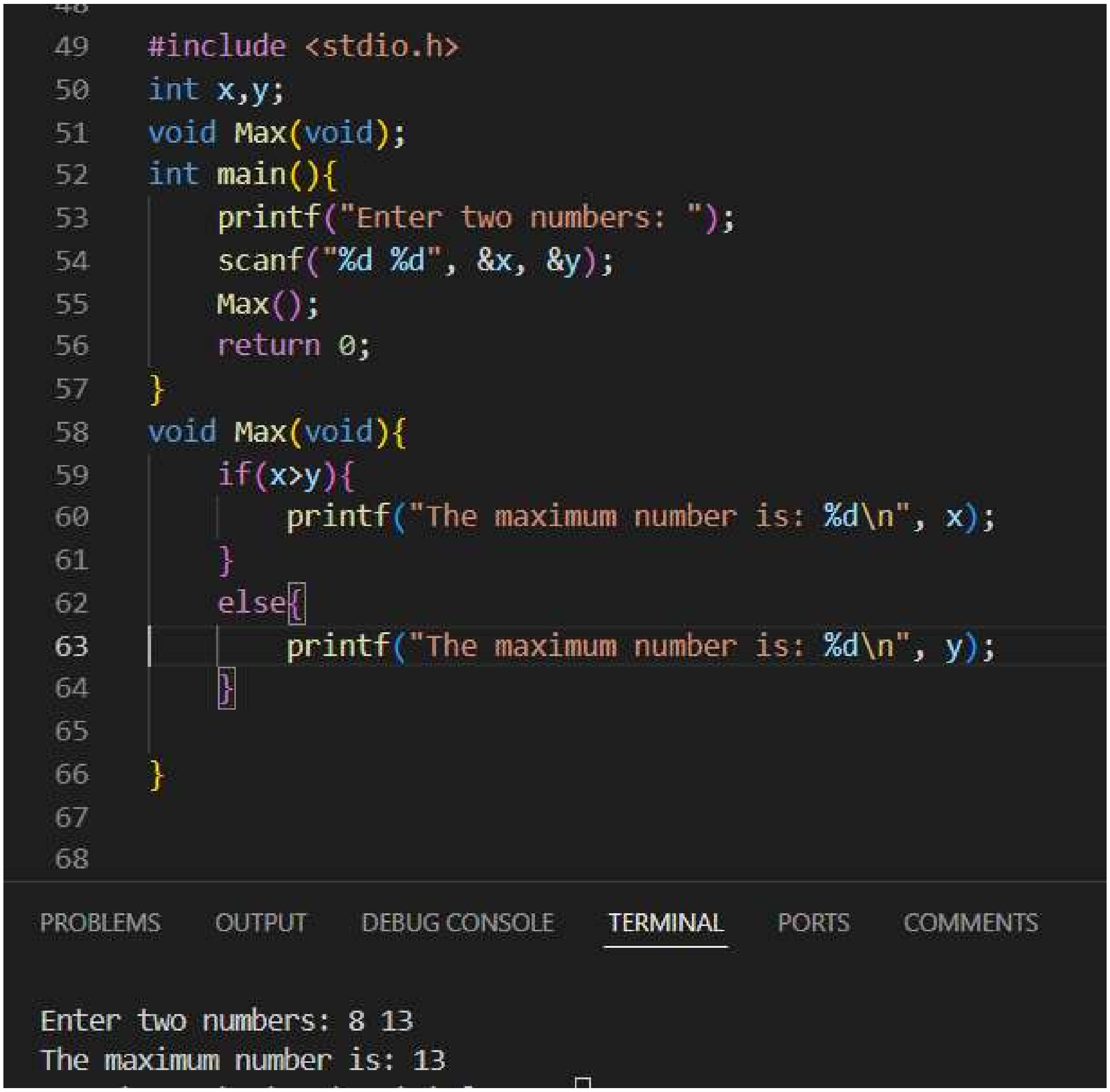


```c
#include <stdio.h>
int x,y;
void Swap(void);
int main(){
    printf("Enter two numbers: ");
    scanf("%d %d", &x, &y);
    printf("Before swapping: x = %d, y = %d\n", x, y);
    Swap();
    return 0;
}
void Swap(void){
    int temp;
    temp=x;
    x=y;
    y=temp;
    printf("After swapping: x = %d, y = %d\n", x, y);

}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter two numbers: 5 2
Before swapping: x = 5, y = 2
After swapping: x = 2, y = 5
```

3)Find Maximum of Two Numbers

Implement a function that takes two integers as arguments and returns the larger of the two. Demonstrate how the original values are not altered.

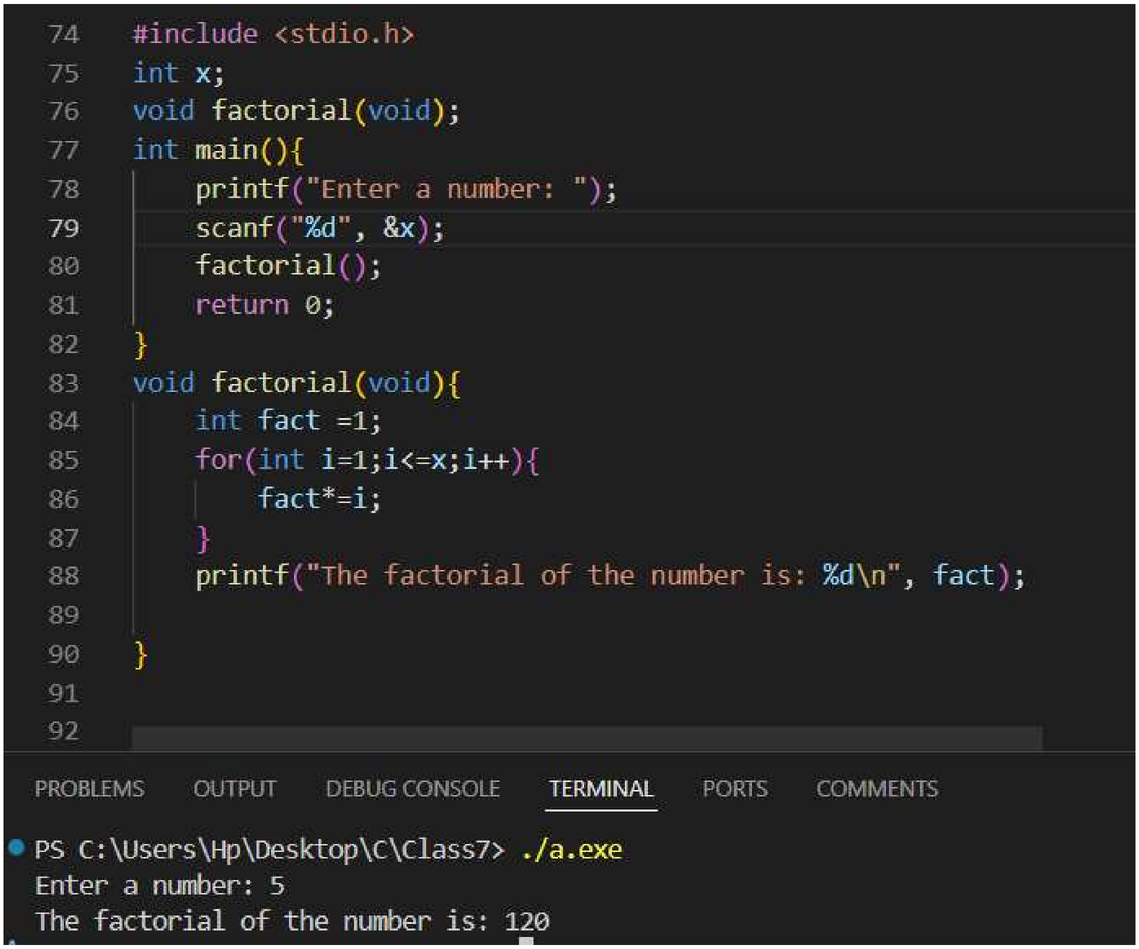


```c
#include <stdio.h>
int x,y;
void Max(void);
int main(){
    printf("Enter two numbers: ");
    scanf("%d %d", &x, &y);
    Max();
    return 0;
}
void Max(void){
    if(x>y){
        printf("The maximum number is: %d\n", x);
    }
    else{
        printf("The maximum number is: %d\n", y);
    }

}
```

```
Enter two numbers: 8 13
The maximum number is: 13
```

4)Factorial Calculation

Create a function to compute the factorial of a given number passed to it. Ensure the original number remains unaltered.

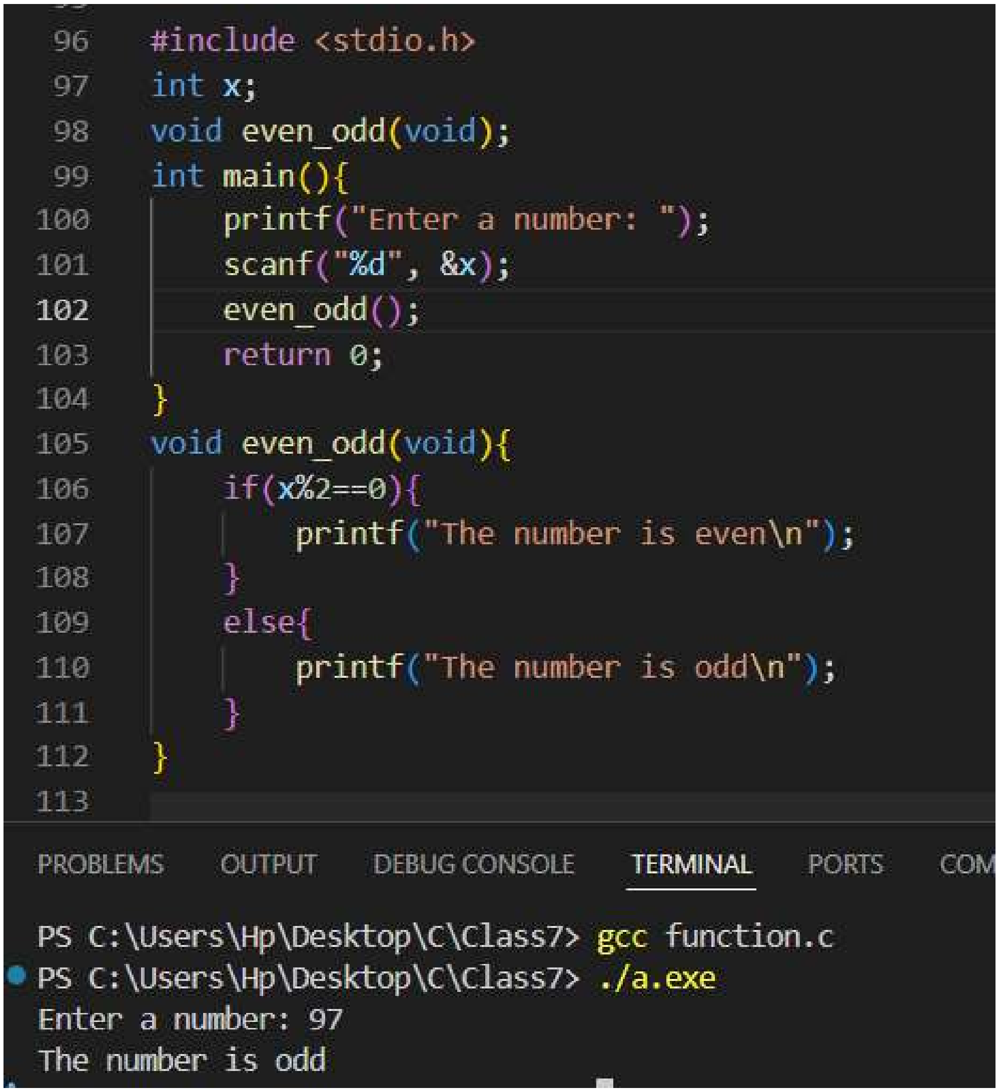


```c
#include <stdio.h>
int x;
void factorial(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    factorial();
    return 0;
}
void factorial(void){
    int fact =1;
    for(int i=1;i<=x;i++){
        fact*=i;
    }
    printf("The factorial of the number is: %d\n", fact);

}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 5
The factorial of the number is: 120
```

5)Check Even or Odd

Write a program where a function determines whether a given integer is even or odd. The function should use call by value.

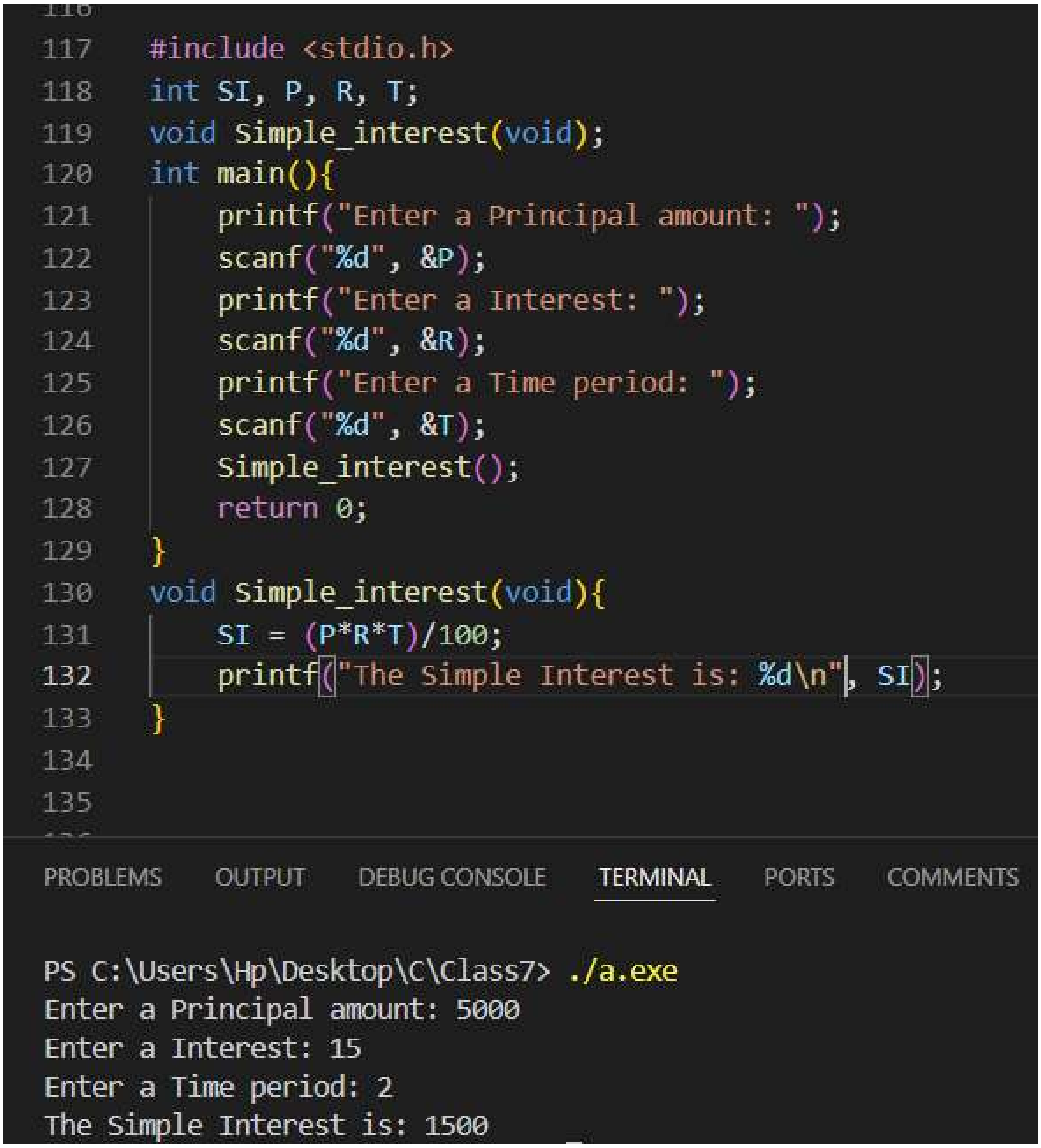


```c
#include <stdio.h>
int x;
void even_odd(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    even_odd();
    return 0;
}
void even_odd(void){
    if(x%2==0){
        printf("The number is even\n");
    }
    else{
        printf("The number is odd\n");
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 97
The number is odd
```

6)Calculate Simple Interest

Write a program that calculates simple interest using a function. Pass principal, rate, and time as arguments and return the computed interest.

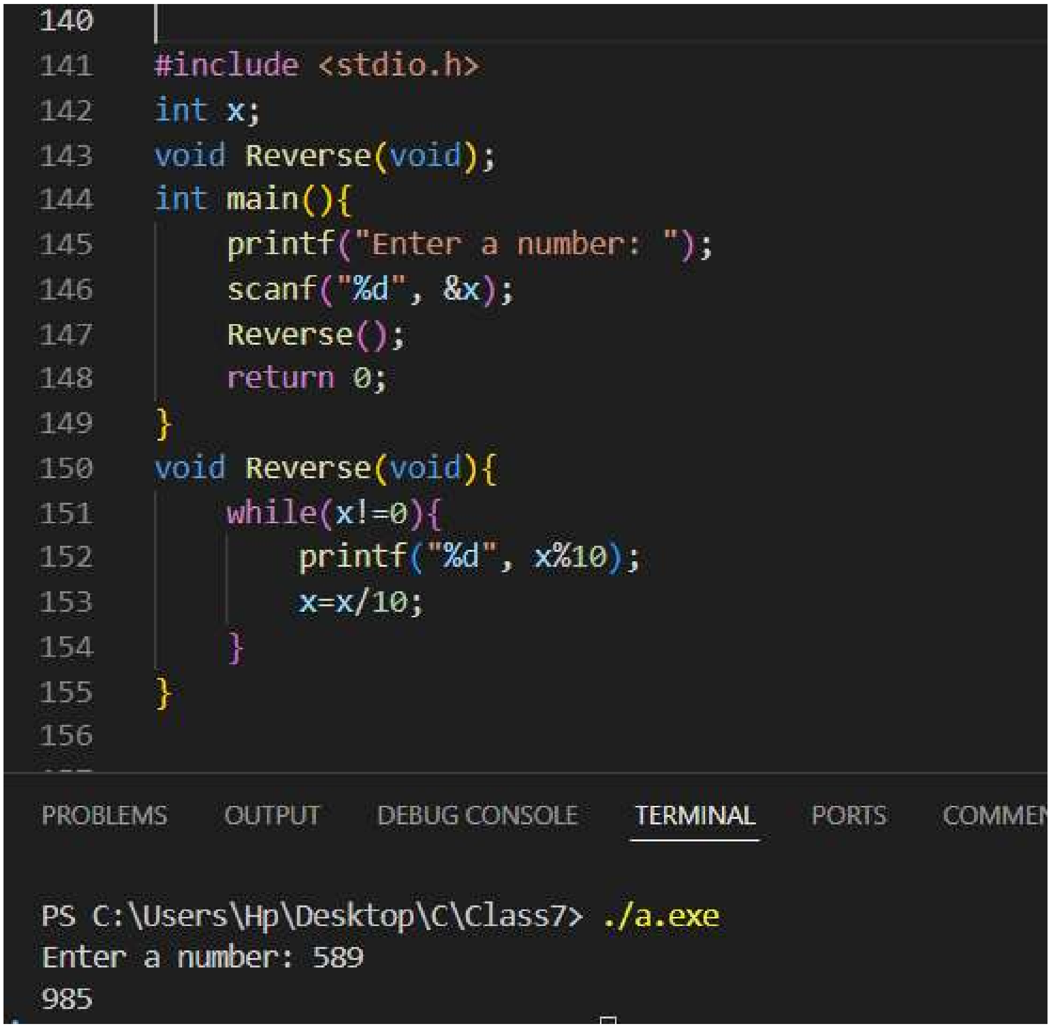


```c
#include <stdio.h>
int SI, P, R, T;
void Simple_interest(void);
int main(){
    printf("Enter a Principal amount: ");
    scanf("%d", &P);
    printf("Enter a Interest: ");
    scanf("%d", &R);
    printf("Enter a Time period: ");
    scanf("%d", &T);
    Simple_interest();
    return 0;
}
void Simple_interest(void){
    SI = (P*R*T)/100;
    printf("The Simple Interest is: %d\n", SI);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a Principal amount: 5000
Enter a Interest: 15
Enter a Time period: 2
The Simple Interest is: 1500
```

7)Reverse a Number

Create a function that takes an integer and returns its reverse. Demonstrate how call by value affects the original number.

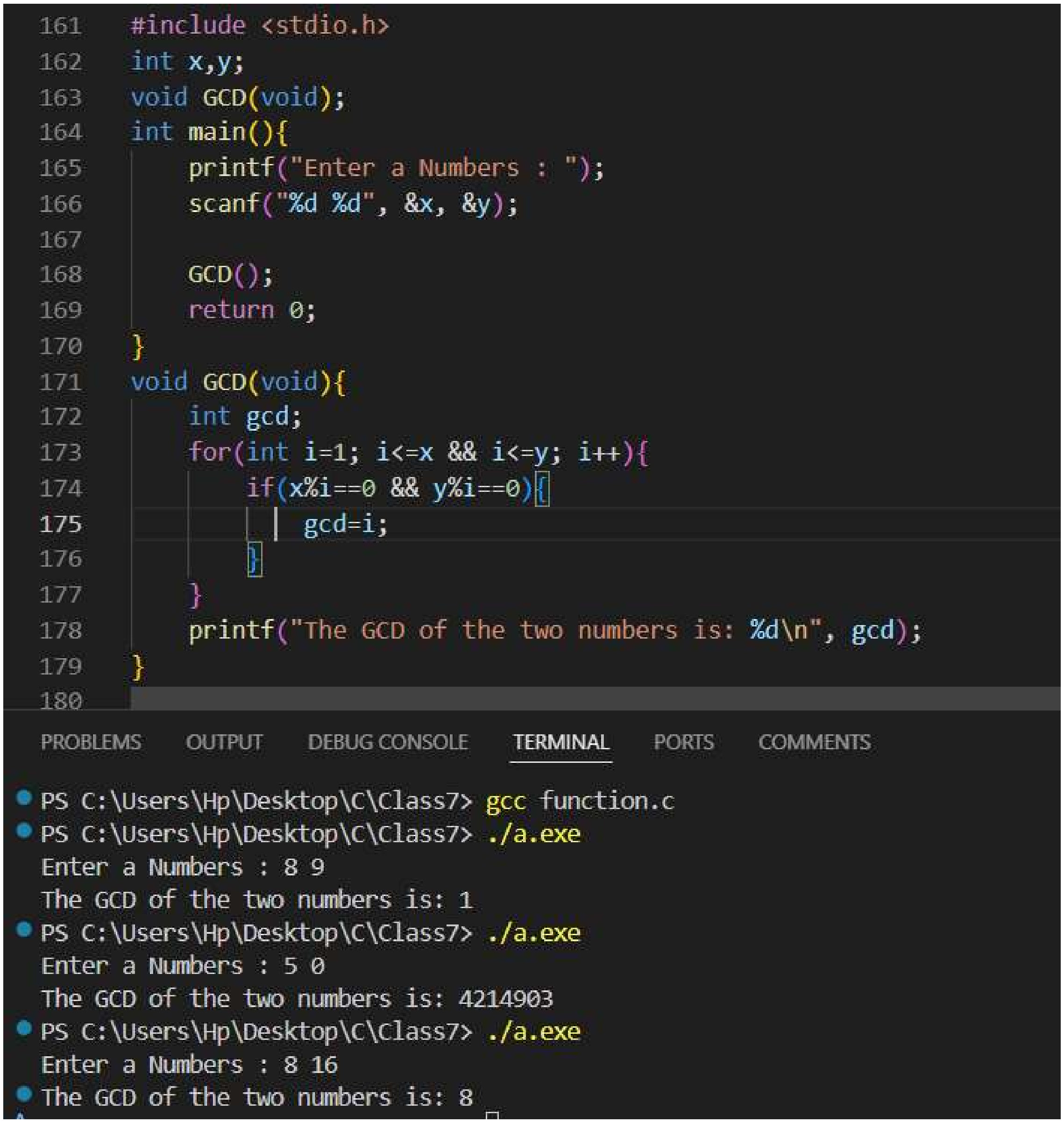


```c

#include <stdio.h>
int x;
void Reverse(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    Reverse();
    return 0;
}
void Reverse(void){
    while(x!=0){
        printf("%d", x%10);
        x=x/10;
    }
}

```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 589
985
```

8)GCD of Two Numbers

Write a function to calculate the greatest common divisor (GCD) of two numbers passed by value.

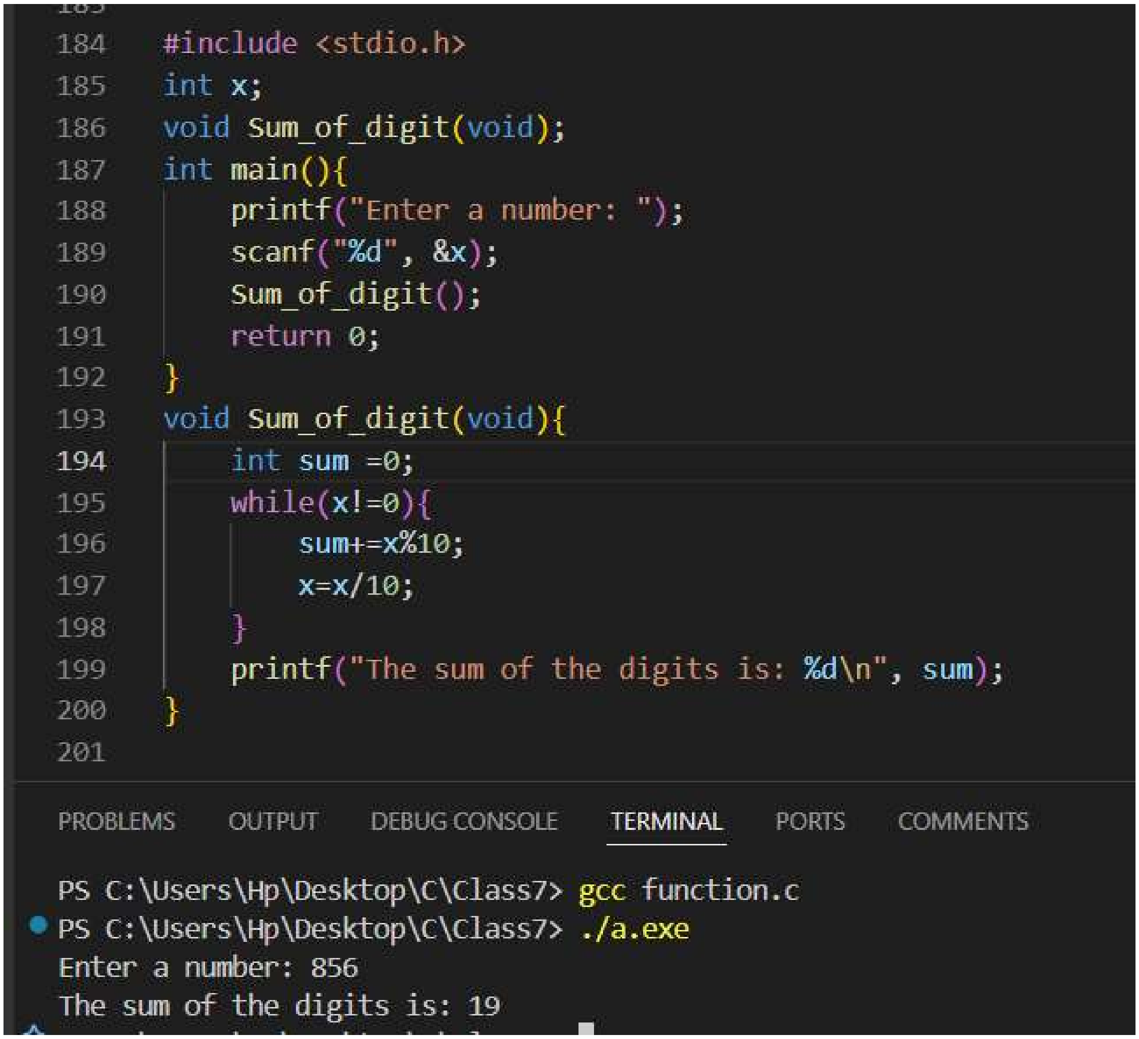


```c
#include <stdio.h>
int x,y;
void GCD(void);
int main(){
    printf("Enter a Numbers : ");
    scanf("%d %d", &x, &y);

    GCD();
    return 0;
}
void GCD(void){
    int gcd;
    for(int i=1; i<=x && i<=y; i++){
        if(x%i==0 && y%i==0){
            gcd=i;
        }
    }
    printf("The GCD of the two numbers is: %d\n", gcd);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a Numbers : 8 9
The GCD of the two numbers is: 1
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a Numbers : 5 0
The GCD of the two numbers is: 4214903
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a Numbers : 8 16
The GCD of the two numbers is: 8
```

9)Sum of Digits

Implement a function that computes the sum of the digits of a number passed as an argument.

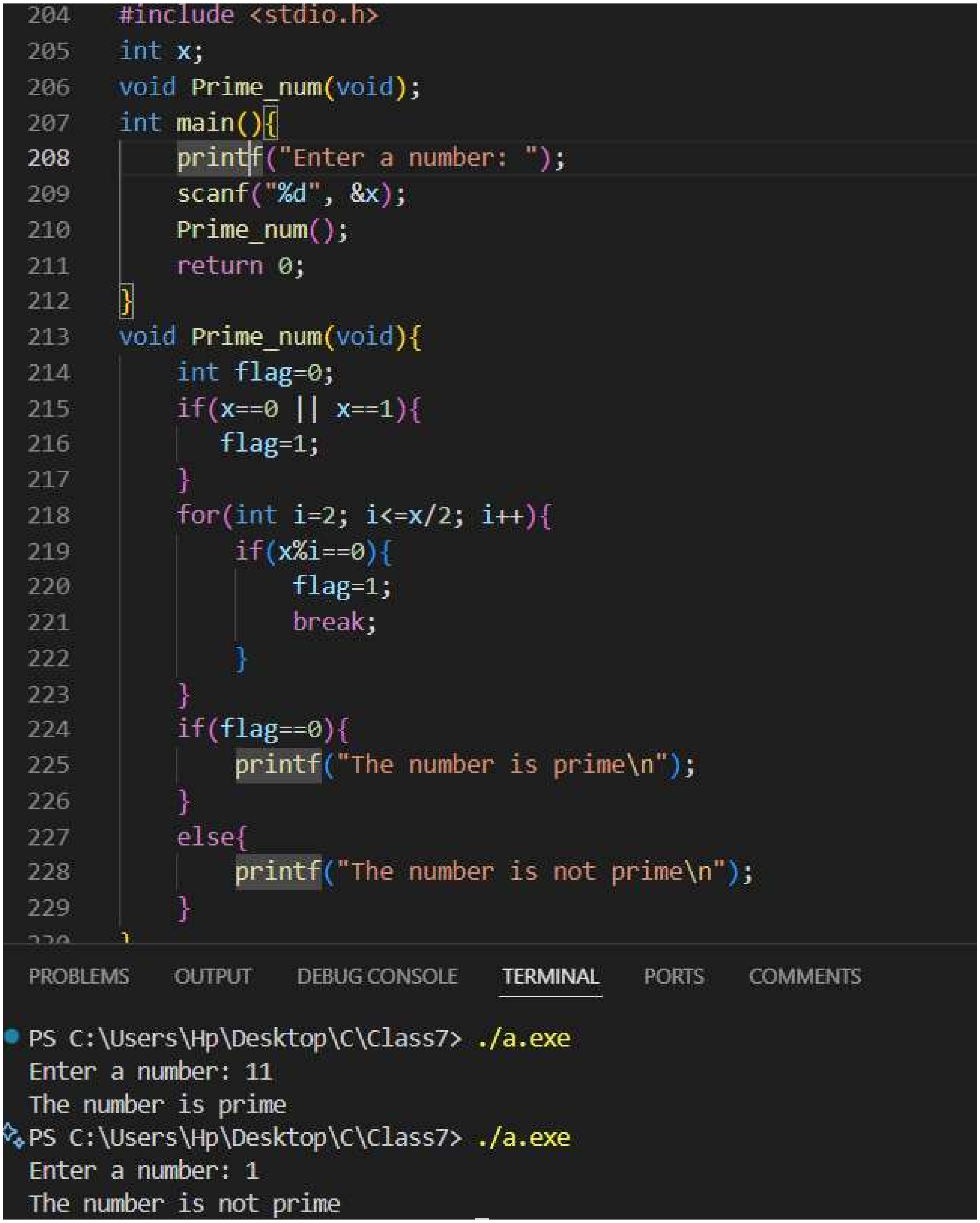


```c
#include <stdio.h>
int x;
void Sum_of_digit(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    Sum_of_digit();
    return 0;
}
void Sum_of_digit(void){
    int sum =0;
    while(x!=0){
        sum+=x%10;
        x=x/10;
    }
    printf("The sum of the digits is: %d\n", sum);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 856
The sum of the digits is: 19
```

10)Prime Number Check

Write a program where a function checks if a given number is prime. Pass the number as an argument by value.

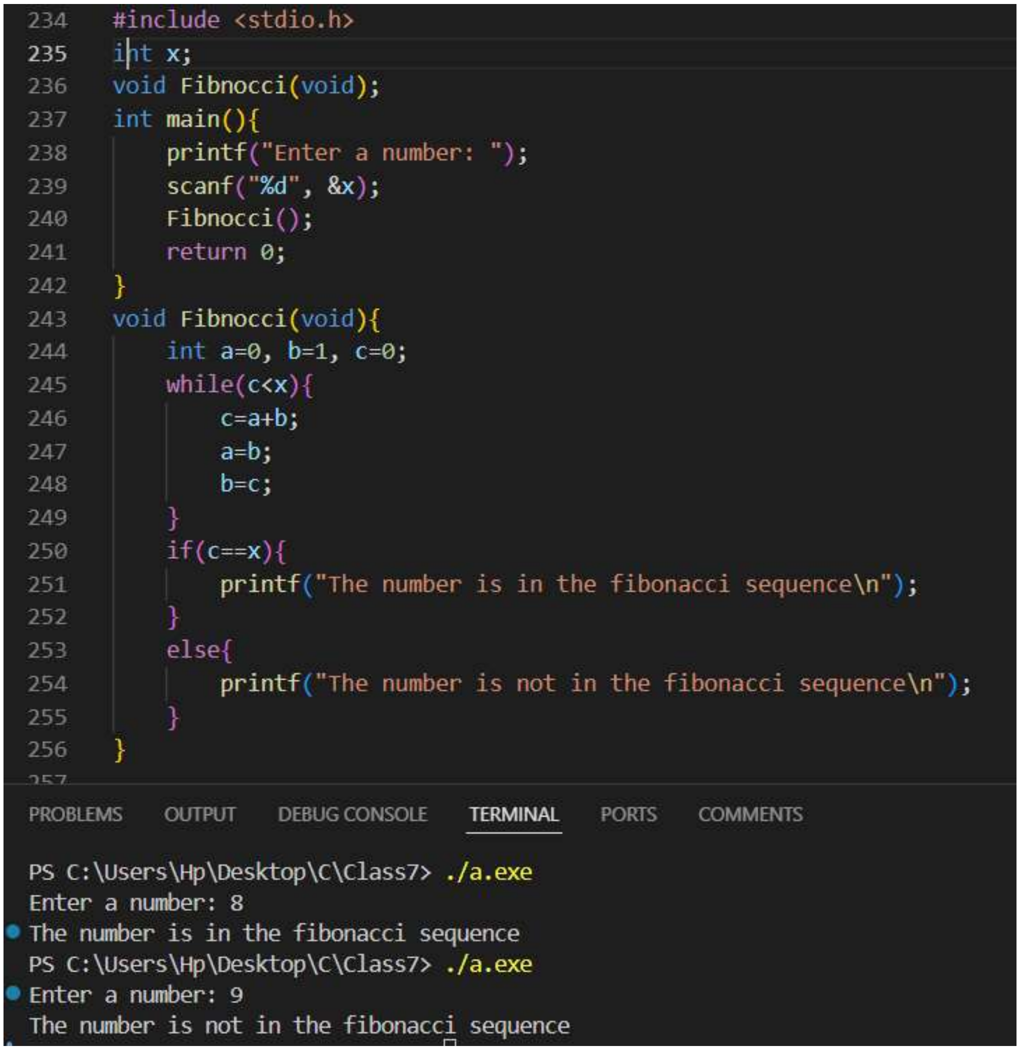


```c
#include <stdio.h>
int x;
void Prime_num(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    Prime_num();
    return 0;
}
void Prime_num(void){
    int flag=0;
    if(x==0 || x==1){
       flag=1;
    }
    for(int i=2; i<=x/2; i++){
        if(x%i==0){
            flag=1;
            break;
        }
    }
    if(flag==0){
        printf("The number is prime\n");
    }
    else{
        printf("The number is not prime\n");
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 11
The number is prime
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 1
The number is not prime
```

11)Fibonacci Sequence Check

Create a function that checks whether a given number belongs to the Fibonacci sequence. Pass the number by value.



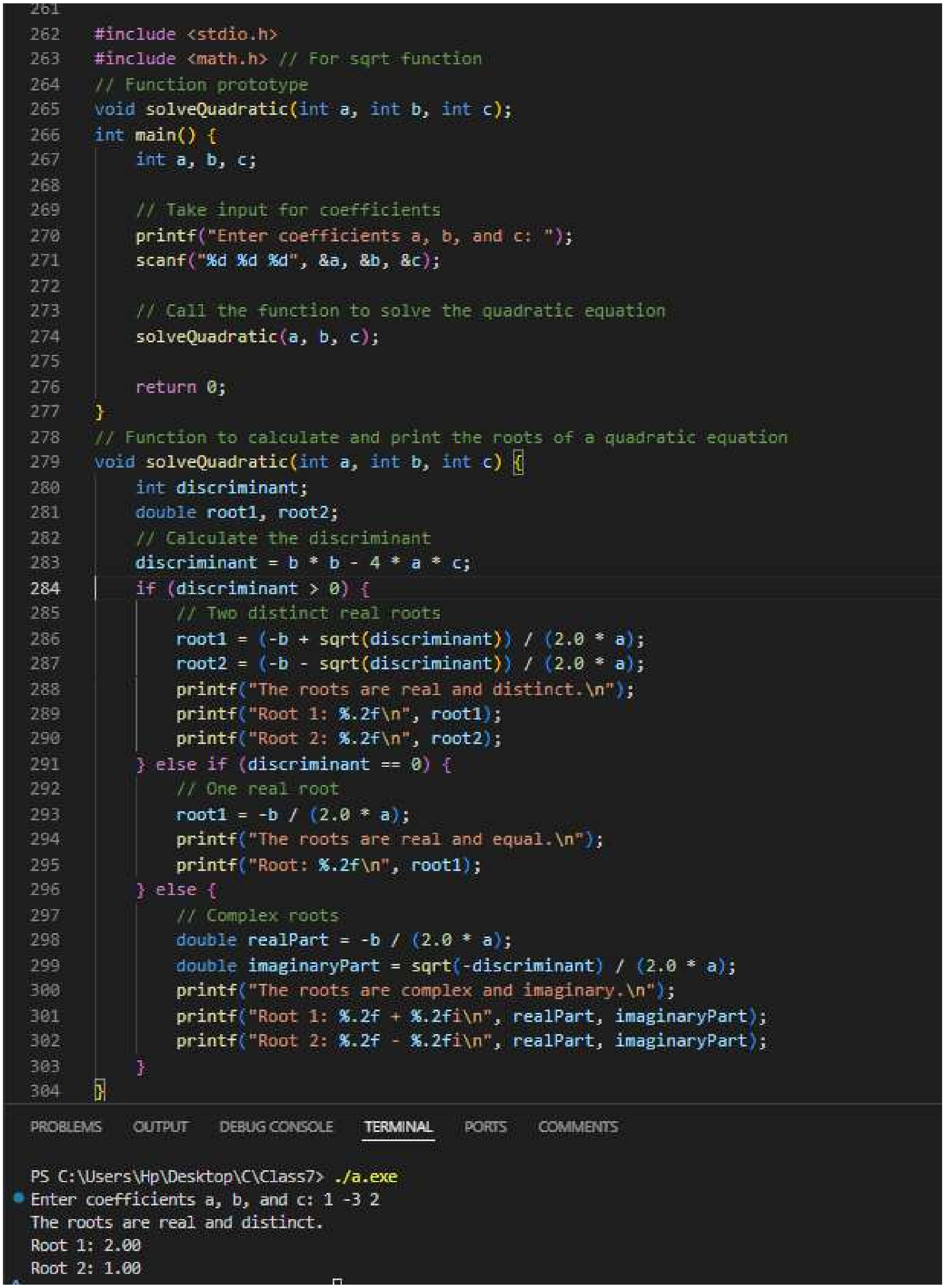
```c
#include <stdio.h>
int x;
void Fibnocci(void);
int main(){
    printf("Enter a number: ");
    scanf("%d", &x);
    Fibnocci();
    return 0;
}
void Fibnocci(void){
    int a=0, b=1, c=0;
    while(c<x){
        c=a+b;
        a=b;
        b=c;
    }
    if(c==x){
        printf("The number is in the fibonacci sequence\n");
    }
    else{
        printf("The number is not in the fibonacci sequence\n");
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 8
The number is in the fibonacci sequence
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 9
The number is not in the fibonacci sequence
```

12)Quadratic Equation Solver

Write a function to calculate the roots of a quadratic equation ax2+bx+c=0ax^2 + bx + c = 0ax2+bx+c=0.

Pass the coefficients a,b,a, b,a,b, and ccc as arguments.

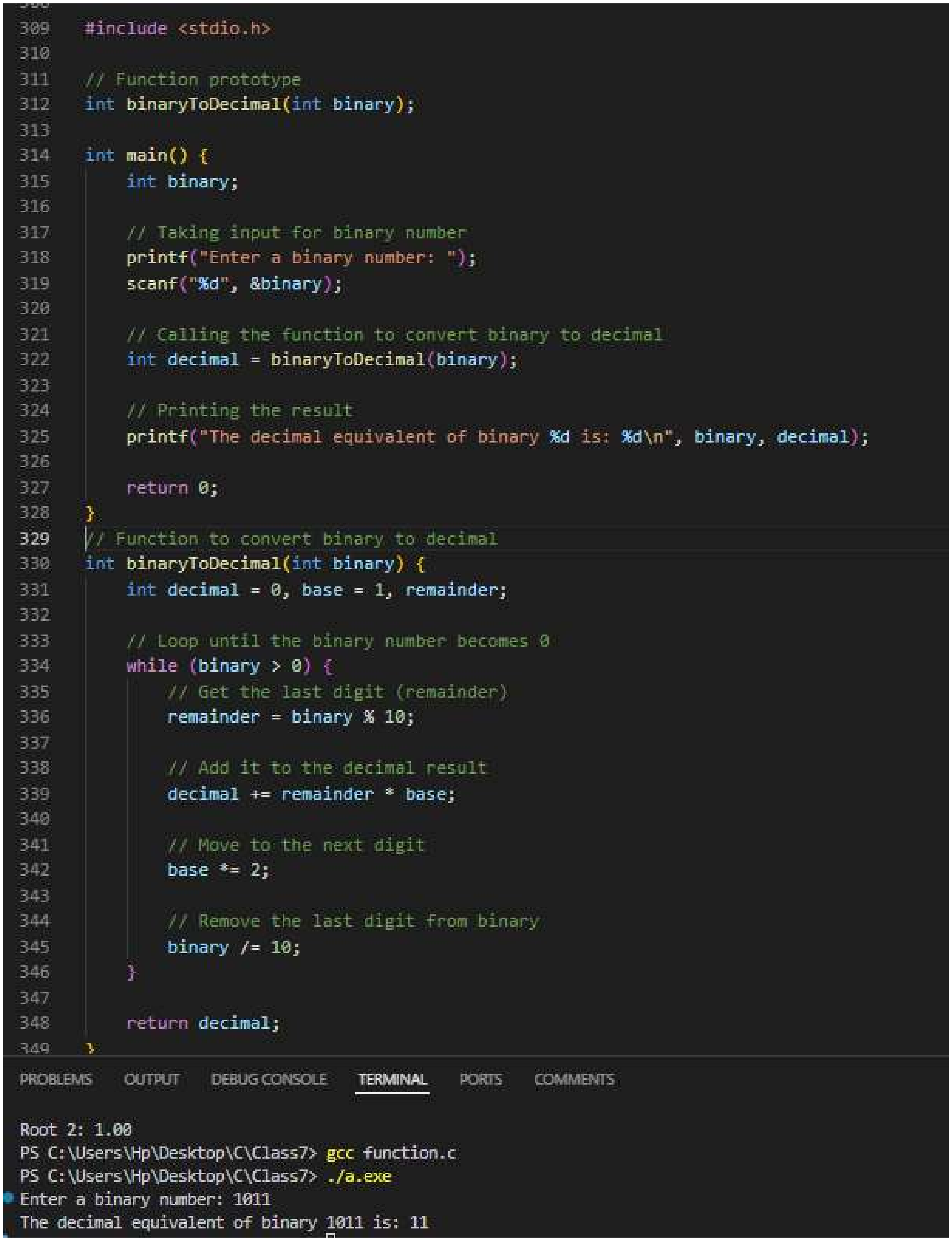


```c
#include <stdio.h>
#include <math.h> // For sqrt function
// Function prototype
void solveQuadratic(int a, int b, int c);
int main() {
    int a, b, c;

    // Take input for coefficients
    printf("Enter coefficients a, b, and c: ");
    scanf("%d %d %d", &a, &b, &c);

    // Call the function to solve the quadratic equation
    solveQuadratic(a, b, c);

    return 0;
}
// Function to calculate and print the roots of a quadratic equation
void solveQuadratic(int a, int b, int c) {
    int discriminant;
    double root1, root2;
    // Calculate the discriminant
    discriminant = b * b - 4 * a * c;
    if (discriminant > 0) {
        // Two distinct real roots
        root1 = (-b + sqrt(discriminant)) / (2.0 * a);
        root2 = (-b - sqrt(discriminant)) / (2.0 * a);
        printf("The roots are real and distinct.\n");
        printf("Root 1: %.2f\n", root1);
        printf("Root 2: %.2f\n", root2);
    } else if (discriminant == 0) {
        // One real root
        root1 = -b / (2.0 * a);
        printf("The roots are real and equal.\n");
        printf("Root: %.2f\n", root1);
    } else {
        // Complex roots
        double realPart = -b / (2.0 * a);
        double imaginaryPart = sqrt(-discriminant) / (2.0 * a);
        printf("The roots are complex and imaginary.\n");
        printf("Root 1: %.2f + %.2fi\n", realPart, imaginaryPart);
        printf("Root 2: %.2f - %.2fi\n", realPart, imaginaryPart);
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter coefficients a, b, and c: 1 -3 2
The roots are real and distinct.
Root 1: 2.00
Root 2: 1.00
```

13)Binary to Decimal Conversion

Implement a function to convert a binary number (passed as an integer) into its decimal equivalent.

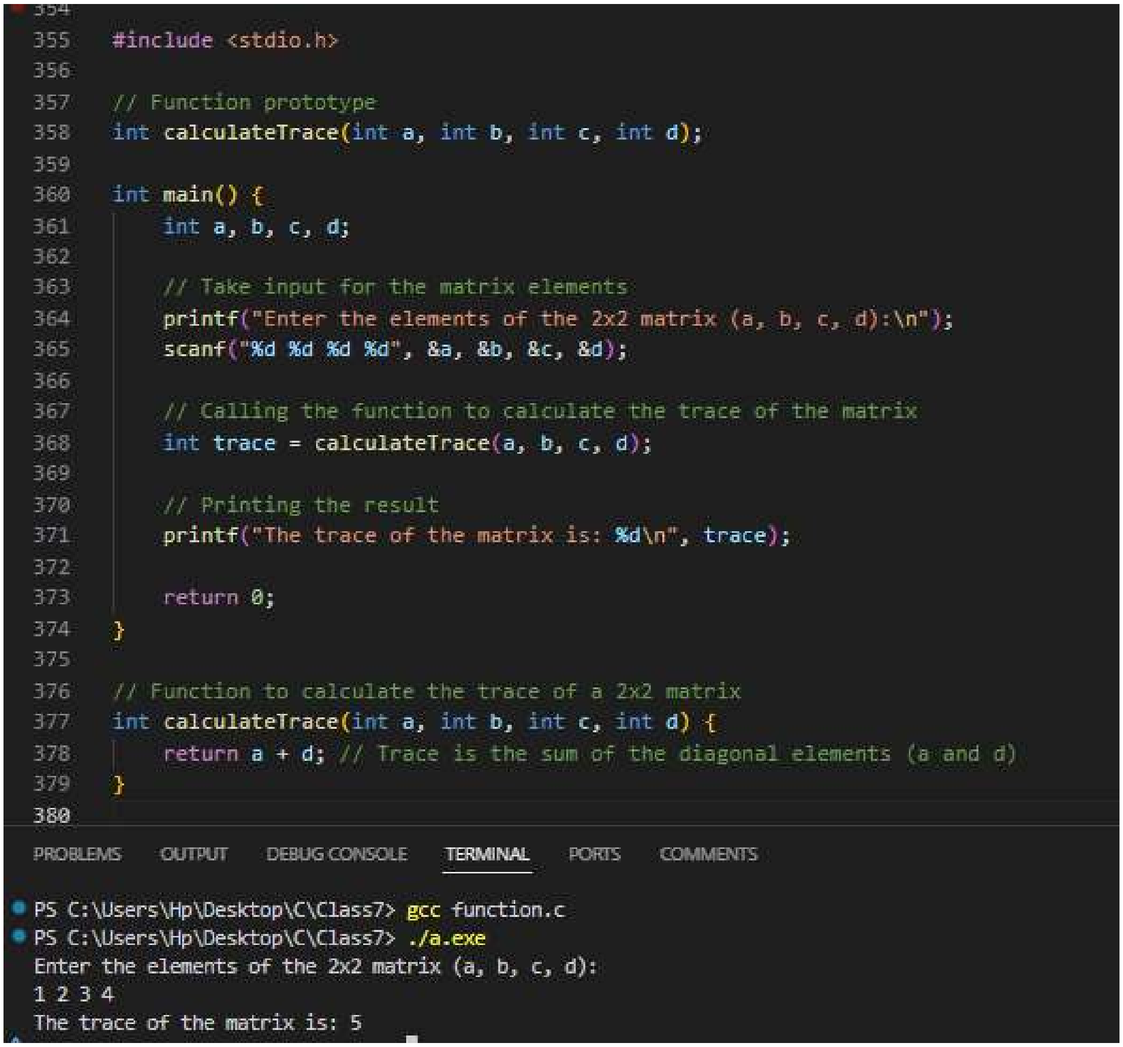


```c
#include <stdio.h>

// Function prototype
int binaryToDecimal(int binary);

int main() {
    int binary;

    // Taking input for binary number
    printf("Enter a binary number: ");
    scanf("%d", &binary);

    // Calling the function to convert binary to decimal
    int decimal = binaryToDecimal(binary);

    // Printing the result
    printf("The decimal equivalent of binary %d is: %d\n", binary, decimal);

    return 0;
}
// Function to convert binary to decimal
int binaryToDecimal(int binary) {
    int decimal = 0, base = 1, remainder;

    // Loop until the binary number becomes 0
    while (binary > 0) {
        // Get the last digit (remainder)
        remainder = binary % 10;

        // Add it to the decimal result
        decimal += remainder * base;

        // Move to the next digit
        base *= 2;

        // Remove the last digit from binary
        binary /= 10;
    }

    return decimal;
}
```

```
Root 2: 1.00
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a binary number: 1011
The decimal equivalent of binary 1011 is: 11
```

14)Matrix Trace Calculation

Write a program where a function computes the trace of a 2x2 matrix (sum of its diagonal elements). Pass the matrix elements individually as arguments.

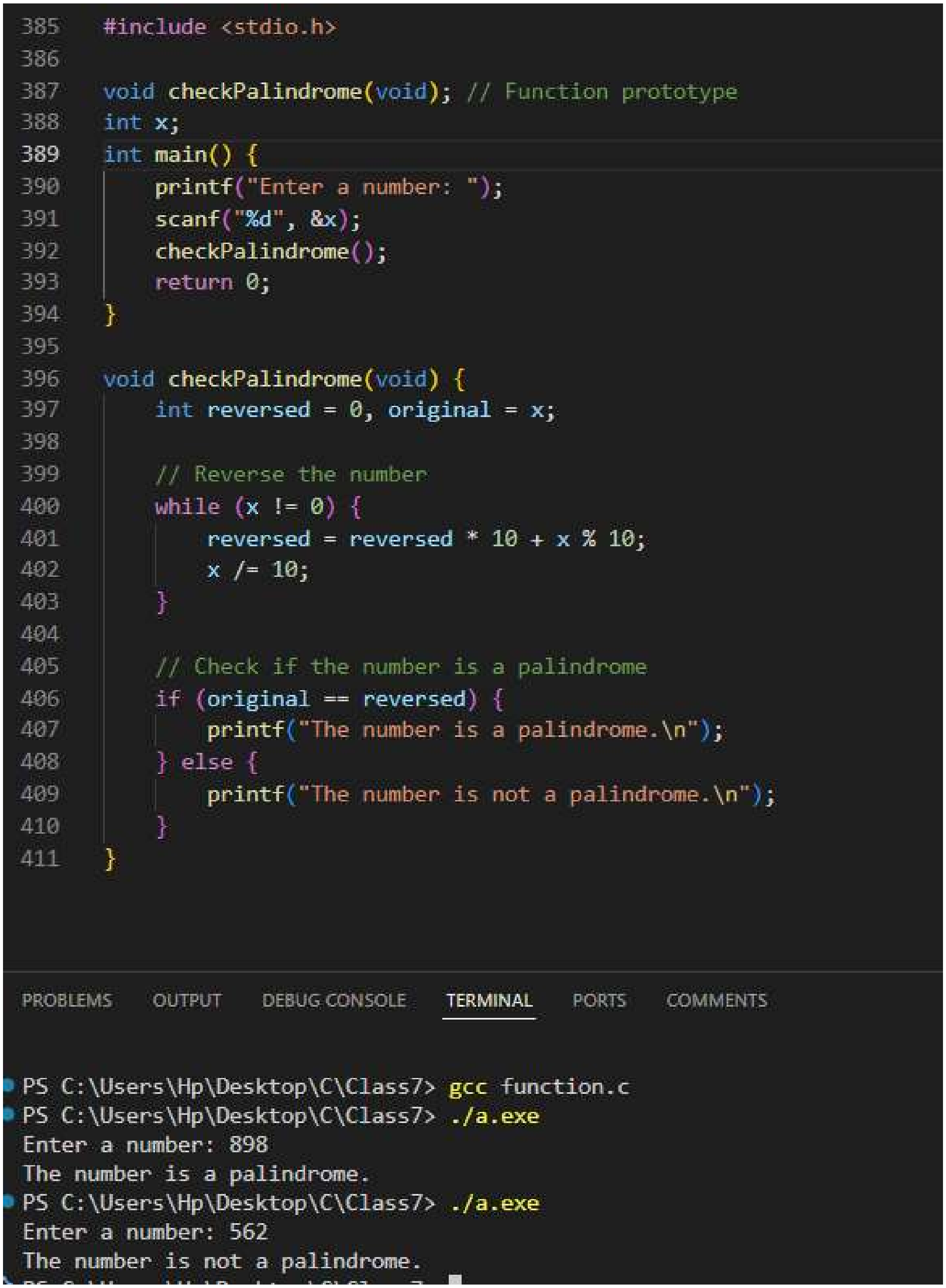


```c
#include <stdio.h>

// Function prototype
int calculateTrace(int a, int b, int c, int d);

int main() {
    int a, b, c, d;

    // Take input for the matrix elements
    printf("Enter the elements of the 2x2 matrix (a, b, c, d):\n");
    scanf("%d %d %d %d", &a, &b, &c, &d);

    // Calling the function to calculate the trace of the matrix
    int trace = calculateTrace(a, b, c, d);

    // Printing the result
    printf("The trace of the matrix is: %d\n", trace);

    return 0;
}

// Function to calculate the trace of a 2x2 matrix
int calculateTrace(int a, int b, int c, int d) {
    return a + d; // Trace is the sum of the diagonal elements (a and d)
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the elements of the 2x2 matrix (a, b, c, d):
1 2 3 4
The trace of the matrix is: 5
```

15)Palindrome Number Check

Create a function that checks whether a given number is a palindrome. Pass the number by value and return the result.



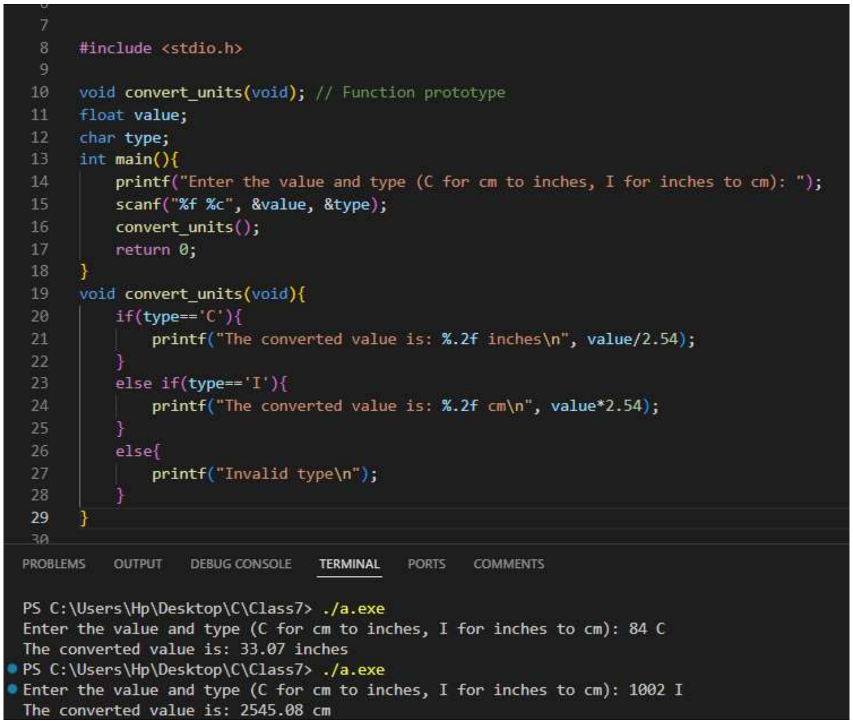
```c
#include <stdio.h>

void checkPalindrome(void); // Function prototype
int x;
int main() {
    printf("Enter a number: ");
    scanf("%d", &x);
    checkPalindrome();
    return 0;
}

void checkPalindrome(void) {
    int reversed = 0, original = x;

    // Reverse the number
    while (x != 0) {
        reversed = reversed * 10 + x % 10;
        x /= 10;
    }

    // Check if the number is a palindrome
    if (original == reversed) {
        printf("The number is a palindrome.\n");
    } else {
        printf("The number is not a palindrome.\n");
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 898
The number is a palindrome.
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter a number: 562
The number is not a palindrome.
```

1. Unit Conversion for Manufacturing Processes

- **Input:** A floating-point value representing the measurement and a character indicating the conversion type (e.g., 'C' for cm-to-inches or 'I' for inches-to-cm).
- **Output:** The converted value.
- **Function:**

float convert_units(float value, char type);



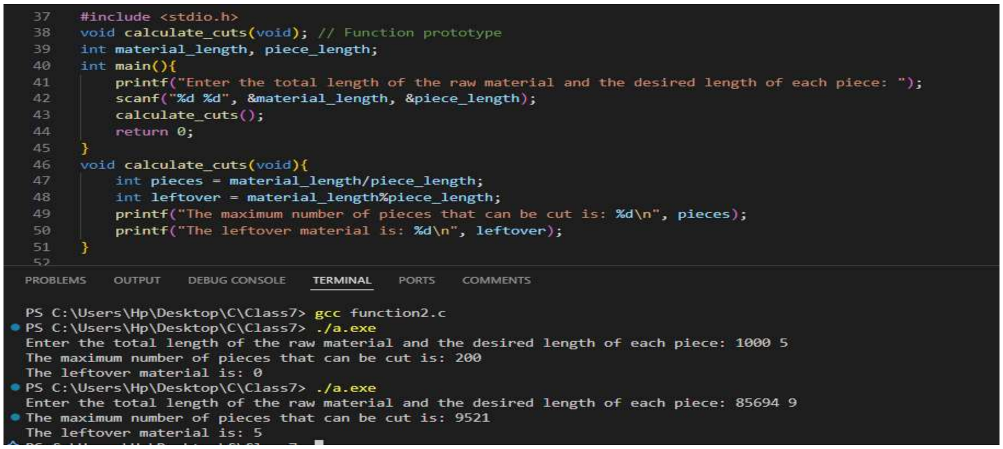
```
#include <stdio.h>

void convert_units(void); // Function prototype
float value;
char type;
int main(){
    printf("Enter the value and type (C for cm to inches, I for inches to cm): ");
    scanf("%f %c", &value, &type);
    convert_units();
    return 0;
}
void convert_units(void){
    if(type=='C'){
        printf("The converted value is: %.2f inches\n", value/2.54);
    }
    else if(type=='I'){
        printf("The converted value is: %.2f cm\n", value*2.54);
    }
    else{
        printf("Invalid type\n");
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the value and type (C for cm to inches, I for inches to cm): 84 C
The converted value is: 33.07 inches
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the value and type (C for cm to inches, I for inches to cm): 1002 I
The converted value is: 2545.08 cm
```

2. Cutting Material Optimization

- **Input:** Two integers: the total length of the raw material and the desired length of each piece.
- **Output:** The maximum number of pieces that can be cut and the leftover material.
- **Function:**

int calculate_cuts(int material_length, int piece_length);



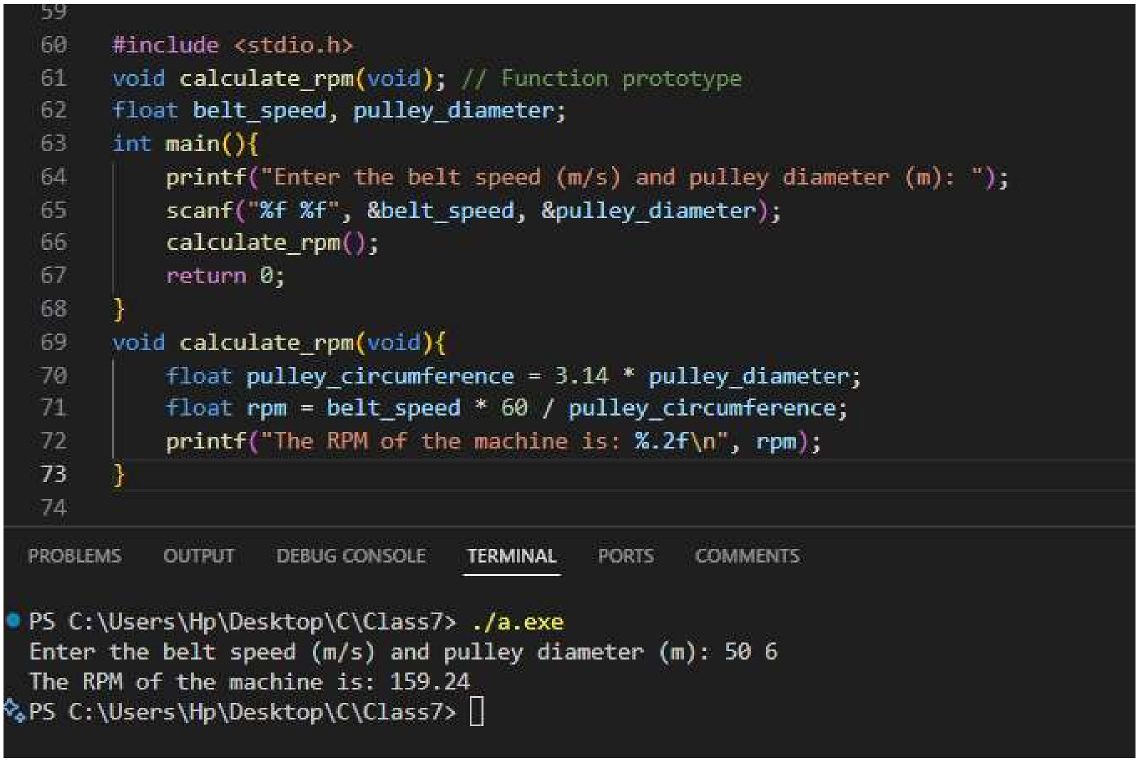
```c
#include <stdio.h>
void calculate_cuts(void); // Function prototype
int material_length, piece_length;
int main(){
    printf("Enter the total length of the raw material and the desired length of each piece: ");
    scanf("%d %d", &material_length, &piece_length);
    calculate_cuts();
    return 0;
}
void calculate_cuts(void){
    int pieces = material_length/piece_length;
    int leftover = material_length%piece_length;
    printf("The maximum number of pieces that can be cut is: %d\n", pieces);
    printf("The leftover material is: %d\n", leftover);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the total length of the raw material and the desired length of each piece: 1000 5
The maximum number of pieces that can be cut is: 200
The leftover material is: 0
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the total length of the raw material and the desired length of each piece: 85694 9
The maximum number of pieces that can be cut is: 9521
The leftover material is: 5
```

3. Machine Speed Calculation

- **Input:** Two floating-point numbers: belt speed (m/s) and pulley diameter (m).
- **Output:** The RPM of the machine.
- **Function:**

float calculate_rpm(float belt_speed, float pulley_diameter);

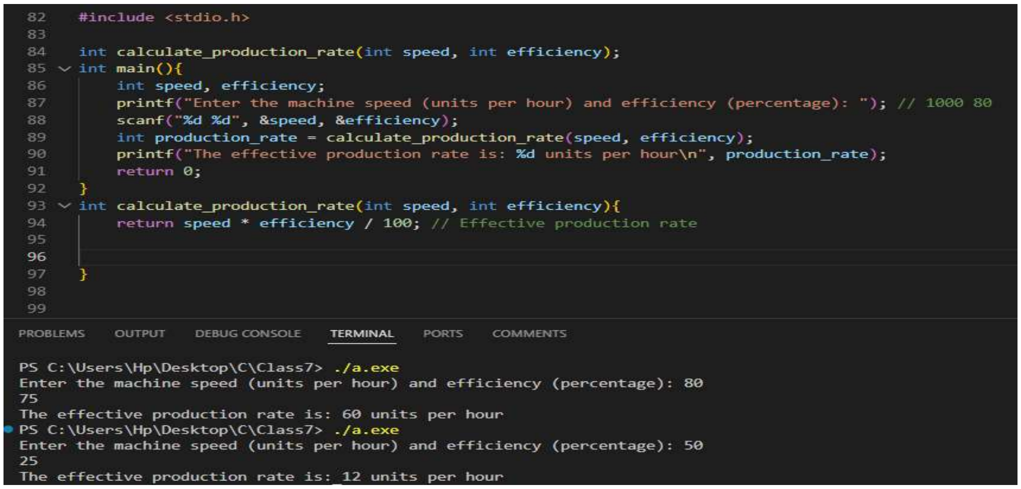


```c
#include <stdio.h>
void calculate_rpm(void); // Function prototype
float belt_speed, pulley_diameter;
int main(){
    printf("Enter the belt speed (m/s) and pulley diameter (m): ");
    scanf("%f %f", &belt_speed, &pulley_diameter);
    calculate_rpm();
    return 0;
}
void calculate_rpm(void){
    float pulley_circumference = 3.14 * pulley_diameter;
    float rpm = belt_speed * 60 / pulley_circumference;
    printf("The RPM of the machine is: %.2f\n", rpm);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the belt speed (m/s) and pulley diameter (m): 50 6
The RPM of the machine is: 159.24
PS C:\Users\Hp\Desktop\C\Class7> 
```

4. Production Rate Estimation

- **Input:** Two integers: machine speed (units per hour) and efficiency (percentage).
- **Output:** The effective production rate.
- **Function:**int calculate_production_rate(int speed, int efficiency);



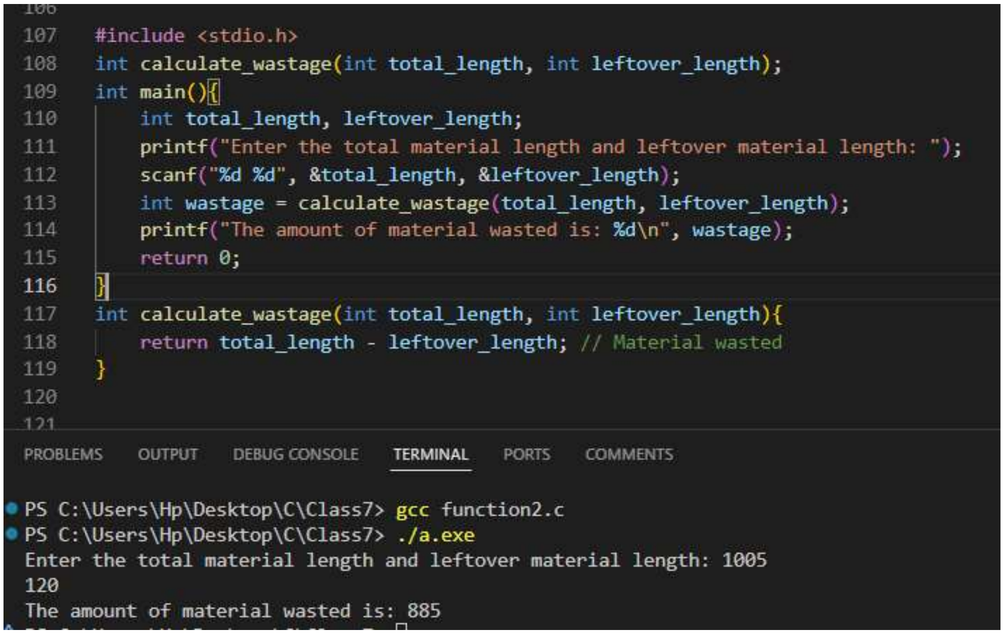
```c
#include <stdio.h>

int calculate_production_rate(int speed, int efficiency);
int main(){
    int speed, efficiency;
    printf("Enter the machine speed (units per hour) and efficiency (percentage): "); // 1000 80
    scanf("%d %d", &speed, &efficiency);
    int production_rate = calculate_production_rate(speed, efficiency);
    printf("The effective production rate is: %d units per hour\n", production_rate);
    return 0;
}
int calculate_production_rate(int speed, int efficiency){
    return speed * efficiency / 100; // Effective production rate


}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the machine speed (units per hour) and efficiency (percentage): 80
75
The effective production rate is: 60 units per hour
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the machine speed (units per hour) and efficiency (percentage): 50
25
The effective production rate is: 12 units per hour
```

5. Material Wastage Calculation

- **Input:** Two integers: total material length and leftover material length.
- **Output:** The amount of material wasted.
- **Function:**

int calculate_wastage(int total_length, int leftover_length);



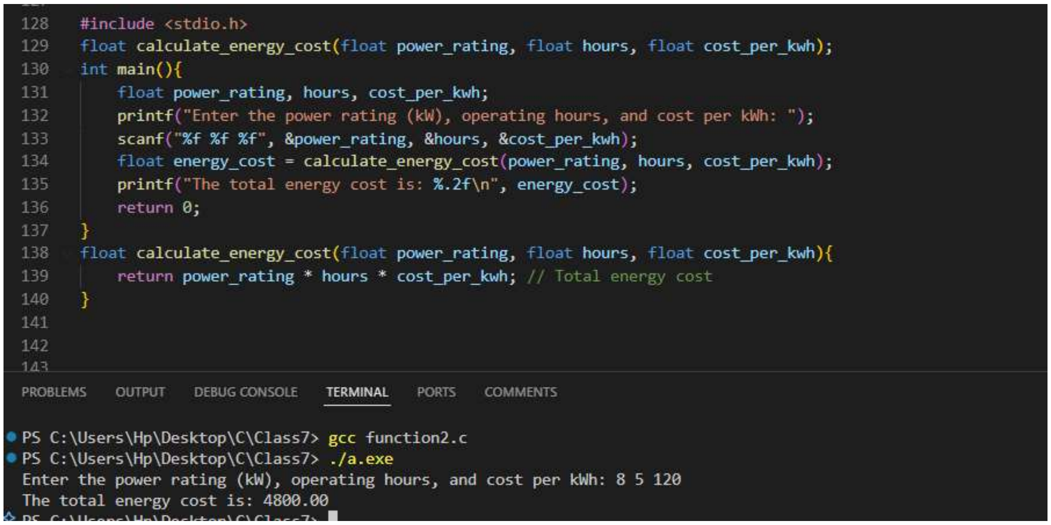
```c
#include <stdio.h>
int calculate_wastage(int total_length, int leftover_length);
int main(){
    int total_length, leftover_length;
    printf("Enter the total material length and leftover material length: ");
    scanf("%d %d", &total_length, &leftover_length);
    int wastage = calculate_wastage(total_length, leftover_length);
    printf("The amount of material wasted is: %d\n", wastage);
    return 0;
}
int calculate_wastage(int total_length, int leftover_length){
    return total_length - leftover_length; // Material wasted
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the total material length and leftover material length: 1005
120
The amount of material wasted is: 885
```

6. Energy Cost Estimation

- **Input:** Three floating-point numbers: power rating (kW), operating hours, and cost per kWh.
- **Output:** The total energy cost.
- **Function:**

float calculate_energy_cost(float power_rating, float hours, float cost_per_kwh);



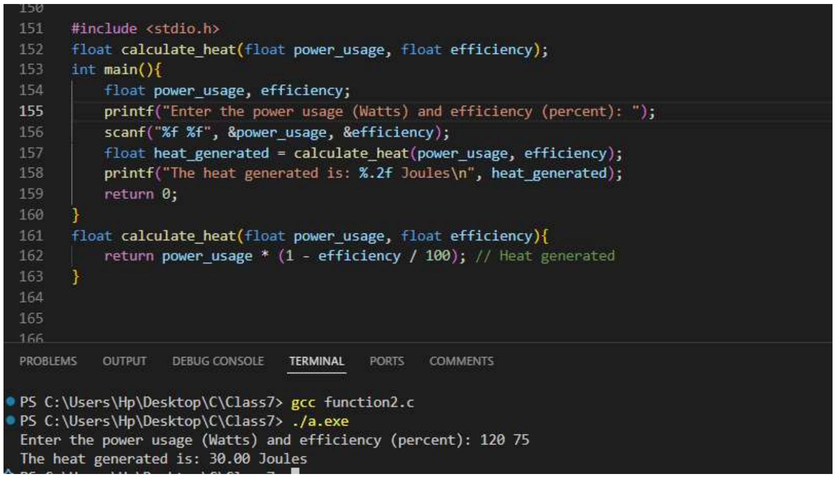
```c
#include <stdio.h>
float calculate_energy_cost(float power_rating, float hours, float cost_per_kwh);
int main(){
    float power_rating, hours, cost_per_kwh;
    printf("Enter the power rating (kW), operating hours, and cost per kWh: ");
    scanf("%f %f %f", &power_rating, &hours, &cost_per_kwh);
    float energy_cost = calculate_energy_cost(power_rating, hours, cost_per_kwh);
    printf("The total energy cost is: %.2f\n", energy_cost);
    return 0;
}
float calculate_energy_cost(float power_rating, float hours, float cost_per_kwh){
    return power_rating * hours * cost_per_kwh; // Total energy cost
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the power rating (kW), operating hours, and cost per kWh: 8 5 120
The total energy cost is: 4800.00
```

7. Heat Generation in Machines

- **Input:** Two floating-point numbers: power usage (Watts) and efficiency (%).
- **Output:** Heat generated (Joules).
- **Function:**

float calculate_heat(float power_usage, float efficiency);

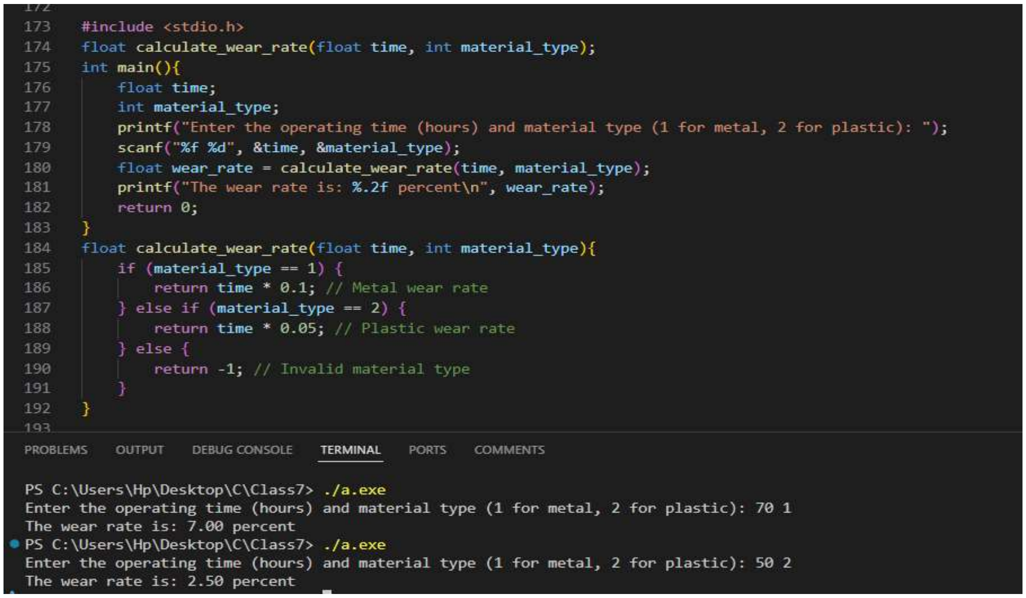


```c
#include <stdio.h>
float calculate_heat(float power_usage, float efficiency);
int main(){
    float power_usage, efficiency;
    printf("Enter the power usage (Watts) and efficiency (percent): ");
    scanf("%f %f", &power_usage, &efficiency);
    float heat_generated = calculate_heat(power_usage, efficiency);
    printf("The heat generated is: %.2f Joules\n", heat_generated);
    return 0;
}
float calculate_heat(float power_usage, float efficiency){
    return power_usage * (1 - efficiency / 100); // Heat generated
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the power usage (Watts) and efficiency (percent): 120 75
The heat generated is: 30.00 Joules
```

8. Tool Wear Rate Calculation

- **Input:** A floating-point number for operating time (hours) and an integer for material type (e.g., 1 for metal, 2 for plastic).
- **Output:** Wear rate (percentage).
- **Function:** float calculate_wear_rate(float time, int material_type);

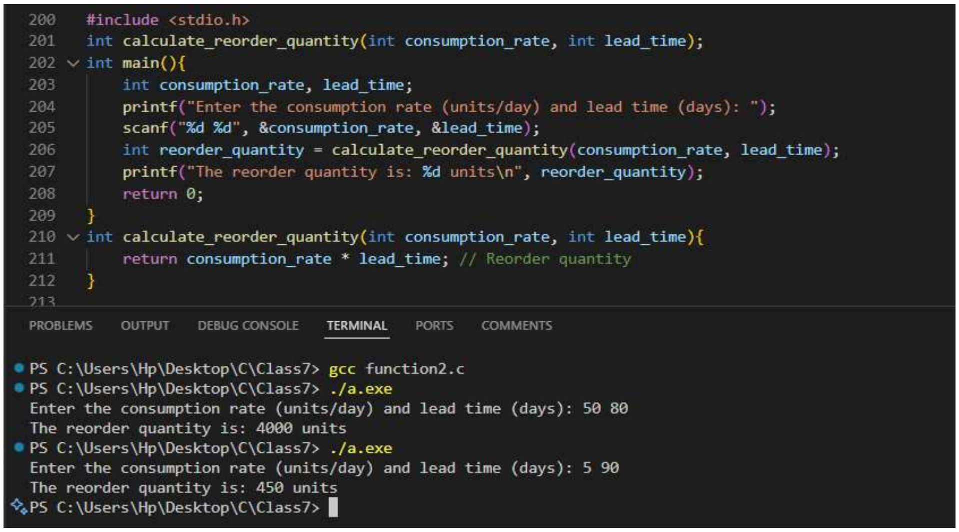


```c
#include <stdio.h>
float calculate_wear_rate(float time, int material_type);
int main(){
    float time;
    int material_type;
    printf("Enter the operating time (hours) and material type (1 for metal, 2 for plastic): ");
    scanf("%f %d", &time, &material_type);
    float wear_rate = calculate_wear_rate(time, material_type);
    printf("The wear rate is: %.2f percent\n", wear_rate);
    return 0;
}
float calculate_wear_rate(float time, int material_type){
    if (material_type == 1) {
        return time * 0.1; // Metal wear rate
    } else if (material_type == 2) {
        return time * 0.05; // Plastic wear rate
    } else {
        return -1; // Invalid material type
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the operating time (hours) and material type (1 for metal, 2 for plastic): 70 1
The wear rate is: 7.00 percent
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the operating time (hours) and material type (1 for metal, 2 for plastic): 50 2
The wear rate is: 2.50 percent
```

9. Inventory Management

- **Input:** Two integers: consumption rate (units/day) and lead time (days).
- **Output:** Reorder quantity (units).
- **Function:** int calculate_reorder_quantity(int consumption_rate, int lead_time);

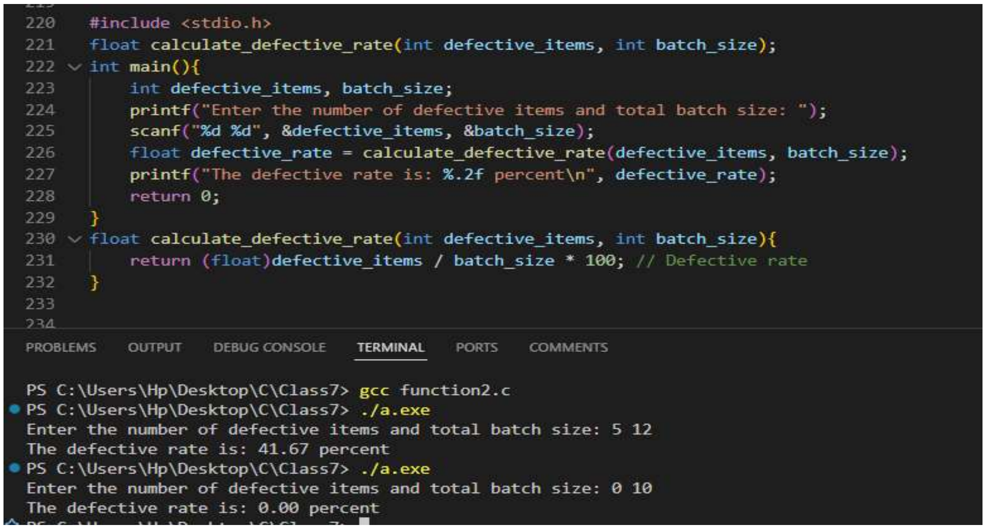


```c
#include <stdio.h>
int calculate_reorder_quantity(int consumption_rate, int lead_time);
int main(){
    int consumption_rate, lead_time;
    printf("Enter the consumption rate (units/day) and lead time (days): ");
    scanf("%d %d", &consumption_rate, &lead_time);
    int reorder_quantity = calculate_reorder_quantity(consumption_rate, lead_time);
    printf("The reorder quantity is: %d units\n", reorder_quantity);
    return 0;
}
int calculate_reorder_quantity(int consumption_rate, int lead_time){
    return consumption_rate * lead_time; // Reorder quantity
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the consumption rate (units/day) and lead time (days): 50 80
The reorder quantity is: 4000 units
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the consumption rate (units/day) and lead time (days): 5 90
The reorder quantity is: 450 units
PS C:\Users\Hp\Desktop\C\Class7>
```

10. Quality Control: Defective Rate Analysis

- **Input:** Two integers: number of defective items and total batch size.
- **Output:** Defective rate (percentage).
- **Function:** float calculate_defective_rate(int defective_items, int batch_size);

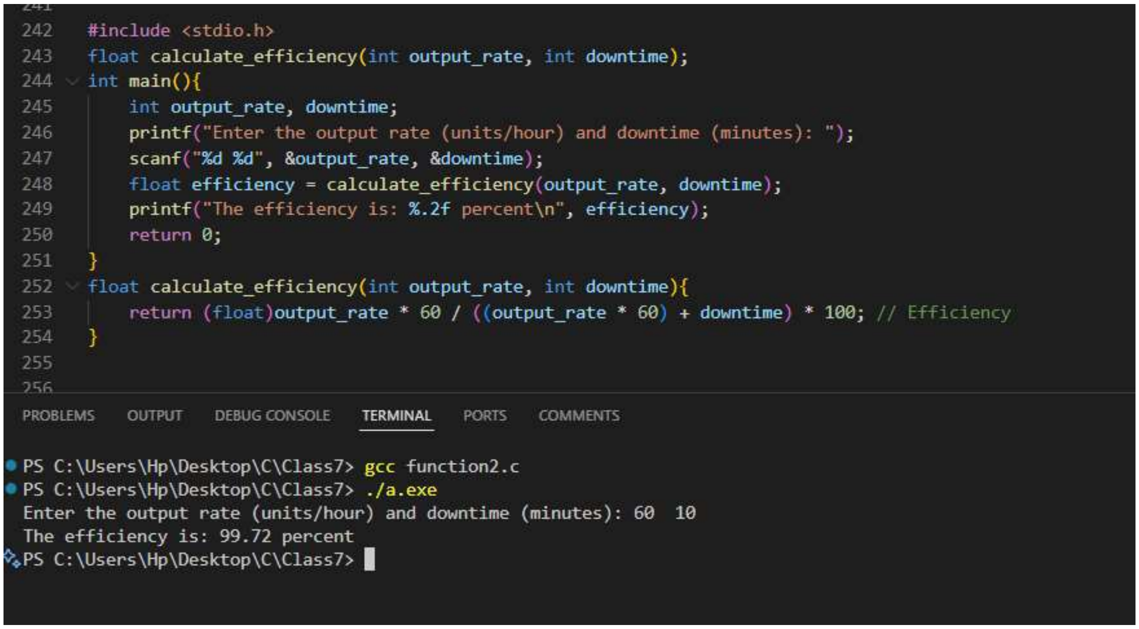


```c
#include <stdio.h>
float calculate_defective_rate(int defective_items, int batch_size);
int main(){
    int defective_items, batch_size;
    printf("Enter the number of defective items and total batch size: ");
    scanf("%d %d", &defective_items, &batch_size);
    float defective_rate = calculate_defective_rate(defective_items, batch_size);
    printf("The defective rate is: %.2f percent\n", defective_rate);
    return 0;
}
float calculate_defective_rate(int defective_items, int batch_size){
    return (float)defective_items / batch_size * 100; // Defective rate
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the number of defective items and total batch size: 5 12
The defective rate is: 41.67 percent
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the number of defective items and total batch size: 0 10
The defective rate is: 0.00 percent
```

11. Assembly Line Efficiency

- **Input:** Two integers: output rate (units/hour) and downtime (minutes).
- **Output:** Efficiency (percentage).
- **Function:** float calculate_efficiency(int output_rate, int downtime);

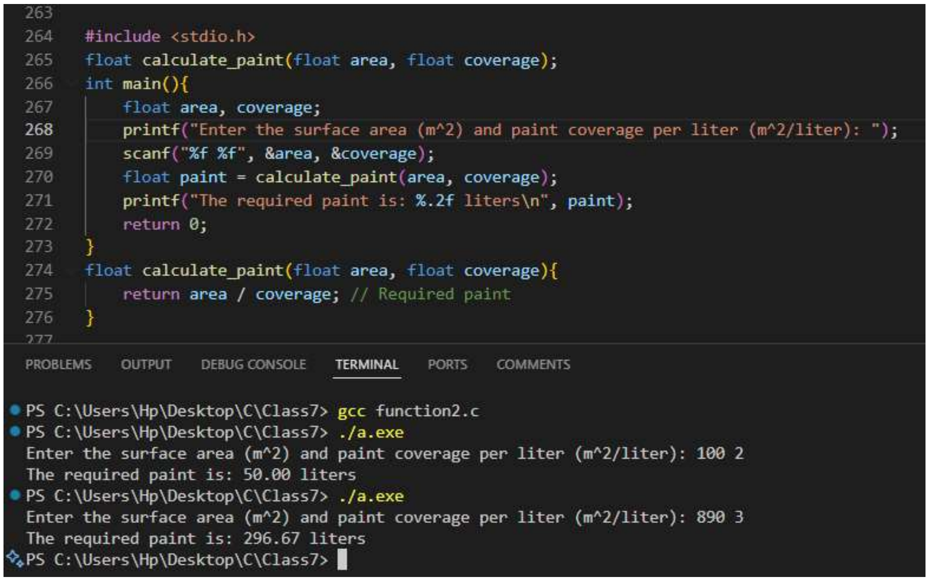


```c
#include <stdio.h>
float calculate_efficiency(int output_rate, int downtime);
int main(){
    int output_rate, downtime;
    printf("Enter the output rate (units/hour) and downtime (minutes): ");
    scanf("%d %d", &output_rate, &downtime);
    float efficiency = calculate_efficiency(output_rate, downtime);
    printf("The efficiency is: %.2f percent\n", efficiency);
    return 0;
}
float calculate_efficiency(int output_rate, int downtime){
    return (float)output_rate * 60 / ((output_rate * 60) + downtime) * 100; // Efficiency
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the output rate (units/hour) and downtime (minutes): 60  10
The efficiency is: 99.72 percent
PS C:\Users\Hp\Desktop\C\Class7>
```

12. Paint Coverage Estimation

- **Input:** Two floating-point numbers: surface area (m^2) and paint coverage per liter (m^2/liter).
- **Output:** Required paint (liters).
- **Function:** float calculate_paint(float area, float coverage);

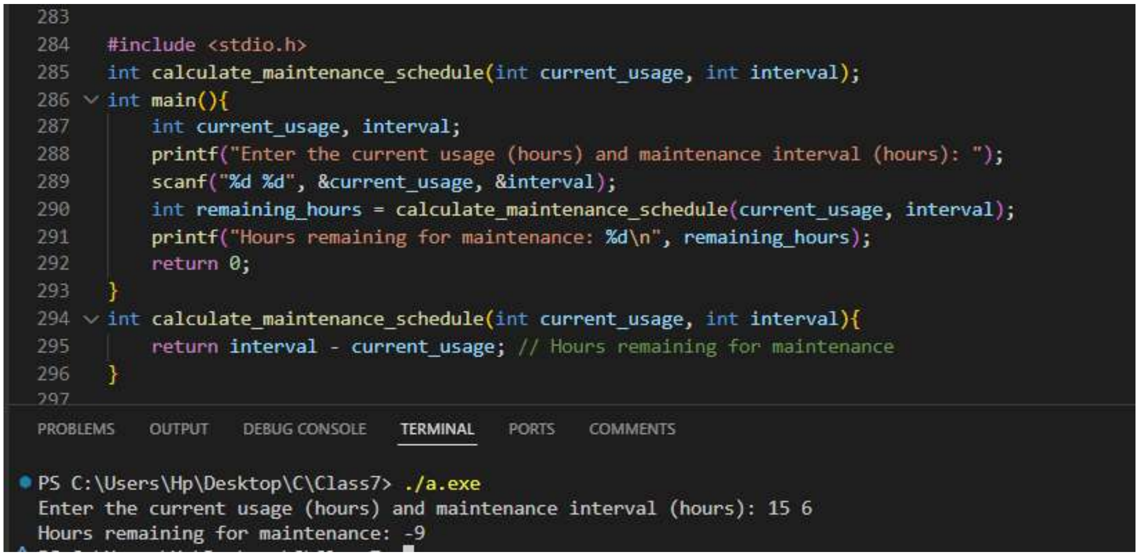


```c
#include <stdio.h>
float calculate_paint(float area, float coverage);
int main(){
    float area, coverage;
    printf("Enter the surface area (m^2) and paint coverage per liter (m^2/liter): ");
    scanf("%f %f", &area, &coverage);
    float paint = calculate_paint(area, coverage);
    printf("The required paint is: %.2f liters\n", paint);
    return 0;
}
float calculate_paint(float area, float coverage){
    return area / coverage; // Required paint
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the surface area (m^2) and paint coverage per liter (m^2/liter): 100 2
The required paint is: 50.00 liters
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the surface area (m^2) and paint coverage per liter (m^2/liter): 890 3
The required paint is: 296.67 liters
PS C:\Users\Hp\Desktop\C\Class7>
```

13. Machine Maintenance Schedule

- **Input:** Two integers: current usage (hours) and maintenance interval (hours).
- **Output:** Hours remaining for maintenance.
- **Function:** int calculate_maintenance_schedule(int current_usage, int interval);

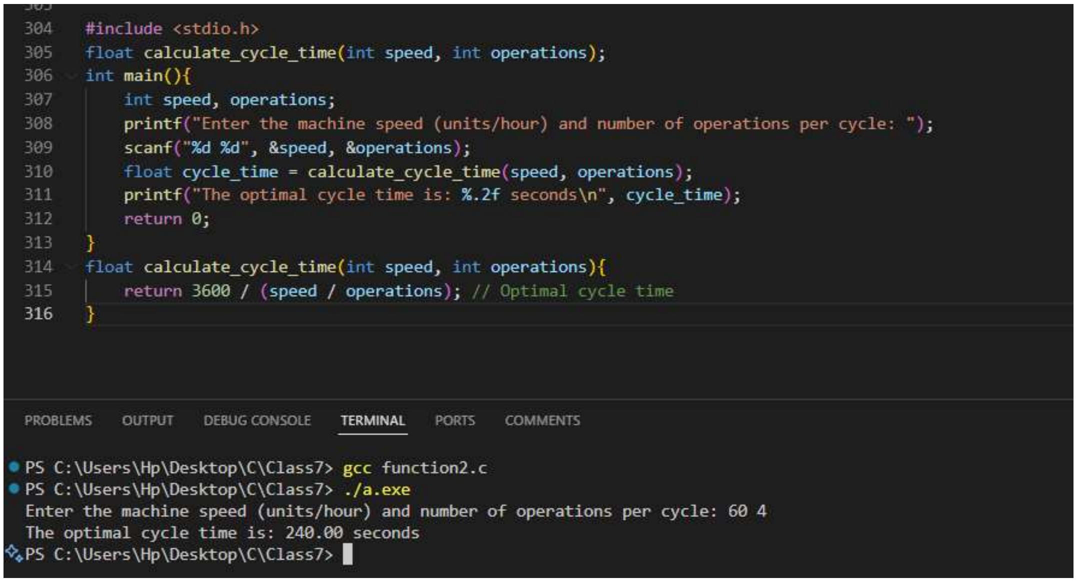


```c
#include <stdio.h>
int calculate_maintenance_schedule(int current_usage, int interval);
int main(){
    int current_usage, interval;
    printf("Enter the current usage (hours) and maintenance interval (hours): ");
    scanf("%d %d", &current_usage, &interval);
    int remaining_hours = calculate_maintenance_schedule(current_usage, interval);
    printf("Hours remaining for maintenance: %d\n", remaining_hours);
    return 0;
}
int calculate_maintenance_schedule(int current_usage, int interval){
    return interval - current_usage; // Hours remaining for maintenance
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the current usage (hours) and maintenance interval (hours): 15 6
Hours remaining for maintenance: -9
```

14. Cycle Time Optimization

- **Input:** Two integers: machine speed (units/hour) and number of operations per cycle.
- **Output:** Optimal cycle time (seconds).
- **Function:** float calculate_cycle_time(int speed, int operations);



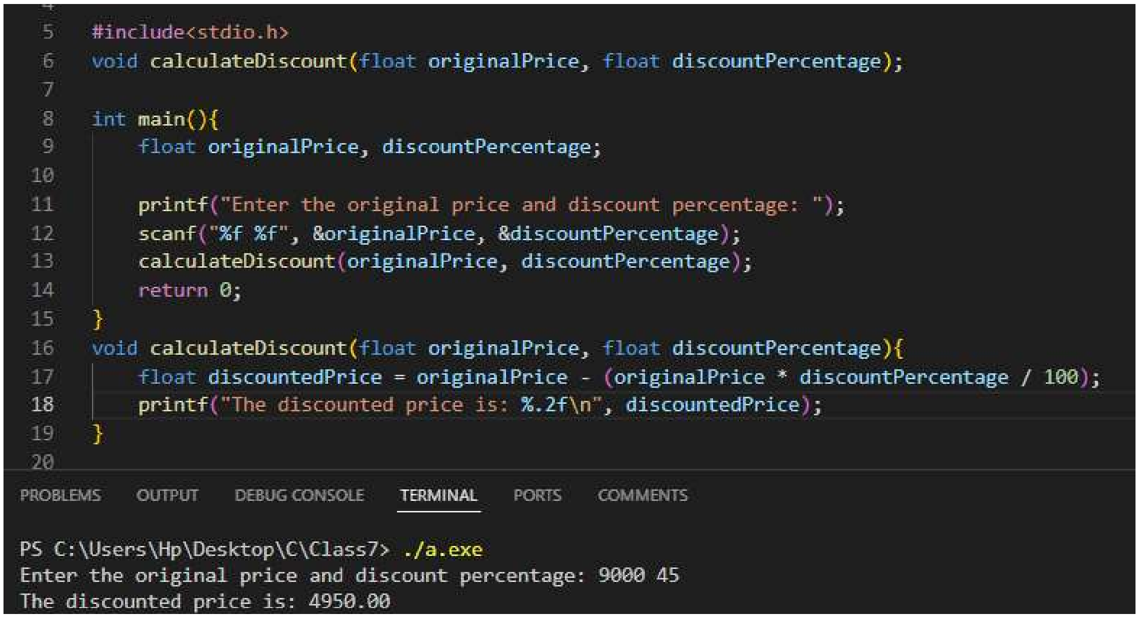
```c
#include <stdio.h>
float calculate_cycle_time(int speed, int operations);
int main(){
    int speed, operations;
    printf("Enter the machine speed (units/hour) and number of operations per cycle: ");
    scanf("%d %d", &speed, &operations);
    float cycle_time = calculate_cycle_time(speed, operations);
    printf("The optimal cycle time is: %.2f seconds\n", cycle_time);
    return 0;
}
float calculate_cycle_time(int speed, int operations){
    return 3600 / (speed / operations); // Optimal cycle time
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function2.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the machine speed (units/hour) and number of operations per cycle: 60 4
The optimal cycle time is: 240.00 seconds
PS C:\Users\Hp\Desktop\C\Class7>
```

1. Write a function that takes the original price of an item and a discount percentage as parameters. The function should return the discounted price without modifying the original price.

Function Prototype:

void calculateDiscount(**float** originalPrice, **float** discountPercentage);



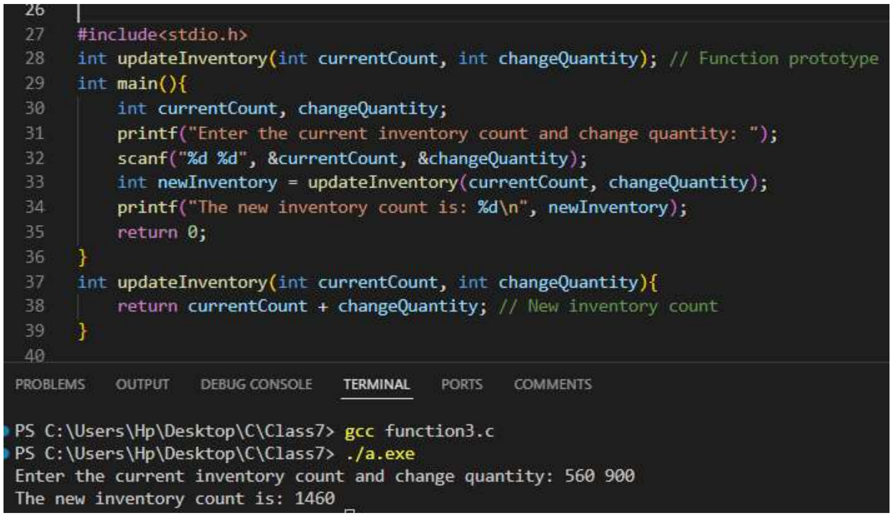
```c
#include<stdio.h>
void calculateDiscount(float originalPrice, float discountPercentage);

int main(){
    float originalPrice, discountPercentage;

    printf("Enter the original price and discount percentage: ");
    scanf("%f %f", &originalPrice, &discountPercentage);
    calculateDiscount(originalPrice, discountPercentage);
    return 0;
}
void calculateDiscount(float originalPrice, float discountPercentage){
    float discountedPrice = originalPrice - (originalPrice * discountPercentage / 100);
    printf("The discounted price is: %.2f\n", discountedPrice);
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the original price and discount percentage: 9000 45
The discounted price is: 4950.00
```

2. Create a function that takes the current inventory count of a product and a quantity to add or remove. The function should return the new inventory count without changing the original count.

Function Prototype:

int updateInventory(**int** currentCount, **int** changeQuantity);



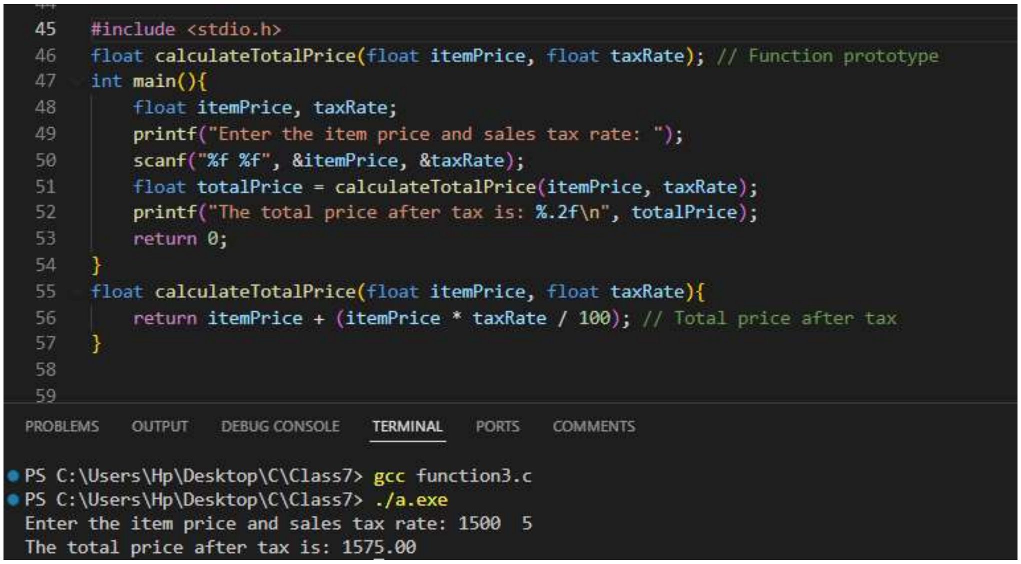
```c
#include<stdio.h>
int updateInventory(int currentCount, int changeQuantity); // Function prototype
int main(){
    int currentCount, changeQuantity;
    printf("Enter the current inventory count and change quantity: ");
    scanf("%d %d", &currentCount, &changeQuantity);
    int newInventory = updateInventory(currentCount, changeQuantity);
    printf("The new inventory count is: %d\n", newInventory);
    return 0;
}
int updateInventory(int currentCount, int changeQuantity){
    return currentCount + changeQuantity; // New inventory count
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the current inventory count and change quantity: 560 900
The new inventory count is: 1460
```

3. Implement a function that accepts the price of an item and a sales tax rate. The function should return the total price after tax without altering the original price.

Function Prototype:

float calculateTotalPrice(**float** itemPrice, **float** taxRate);

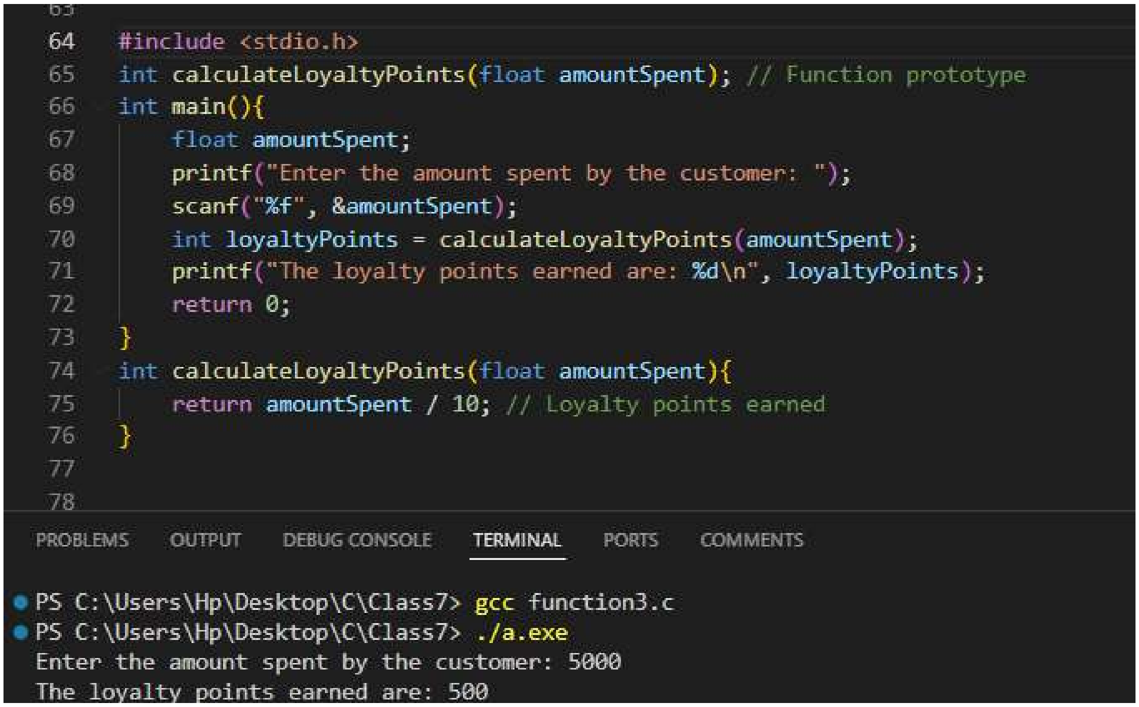


```c
#include <stdio.h>
float calculateTotalPrice(float itemPrice, float taxRate); // Function prototype
int main(){
    float itemPrice, taxRate;
    printf("Enter the item price and sales tax rate: ");
    scanf("%f %f", &itemPrice, &taxRate);
    float totalPrice = calculateTotalPrice(itemPrice, taxRate);
    printf("The total price after tax is: %.2f\n", totalPrice);
    return 0;
}
float calculateTotalPrice(float itemPrice, float taxRate){
    return itemPrice + (itemPrice * taxRate / 100); // Total price after tax
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the item price and sales tax rate: 1500  5
The total price after tax is: 1575.00
```

4. Design a function that takes the amount spent by a customer and returns the loyalty points earned based on a specific conversion rate (e.g., 1 point for every $10 spent). The original amount spent should remain unchanged.

Function Prototype: int calculateLoyaltyPoints(**float** amountSpent);



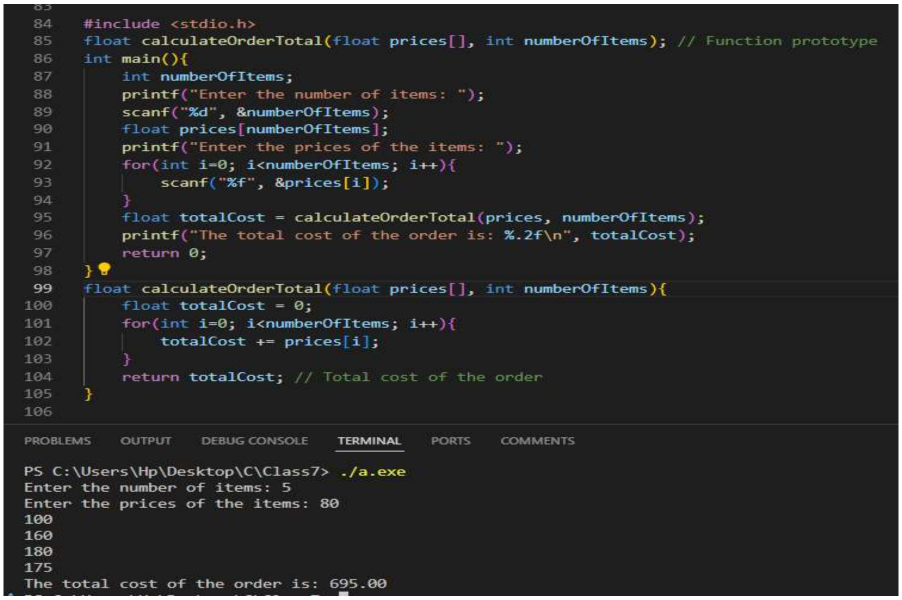
```c
#include <stdio.h>
int calculateLoyaltyPoints(float amountSpent); // Function prototype
int main(){
    float amountSpent;
    printf("Enter the amount spent by the customer: ");
    scanf("%f", &amountSpent);
    int loyaltyPoints = calculateLoyaltyPoints(amountSpent);
    printf("The loyalty points earned are: %d\n", loyaltyPoints);
    return 0;
}
int calculateLoyaltyPoints(float amountSpent){
    return amountSpent / 10; // Loyalty points earned
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the amount spent by the customer: 5000
The loyalty points earned are: 500
```

5. Write a function that receives an array of item prices and the number of items. The function should return the total cost of the order without modifying the individual item prices.

Function Prototype:

float calculateOrderTotal(**float** prices[], **int** numberOfItems);

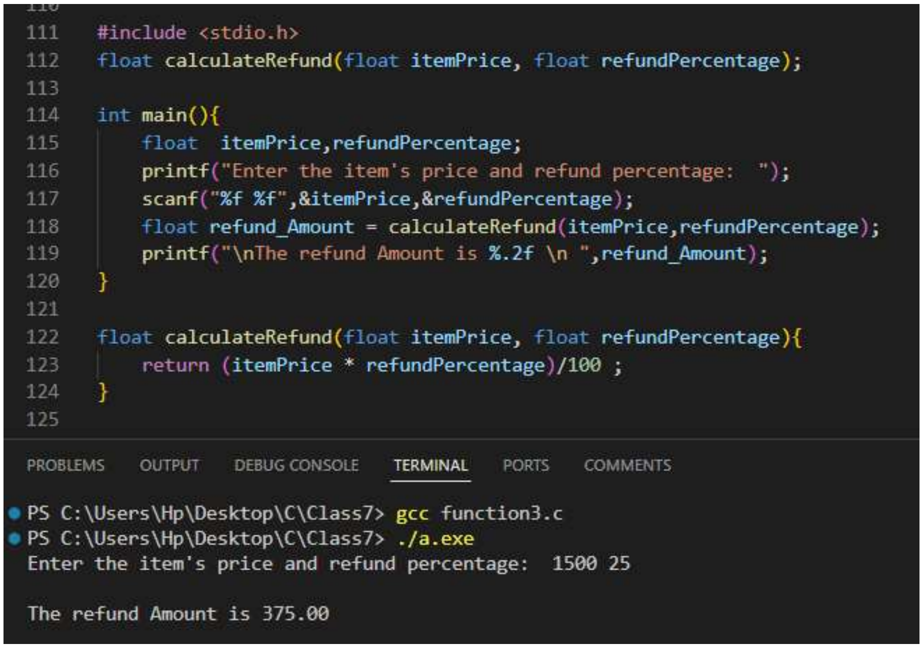


```
#include <stdio.h>
float calculateOrderTotal(float prices[], int numberOfItems); // Function prototype
int main(){
    int numberOfItems;
    printf("Enter the number of items: ");
    scanf("%d", &numberOfItems);
    float prices[numberOfItems];
    printf("Enter the prices of the items: ");
    for(int i=0; i<numberOfItems; i++){
        scanf("%f", &prices[i]);
    }
    float totalCost = calculateOrderTotal(prices, numberOfItems);
    printf("The total cost of the order is: %.2f\n", totalCost);
    return 0;
}
float calculateOrderTotal(float prices[], int numberOfItems){
    float totalCost = 0;
    for(int i=0; i<numberOfItems; i++){
        totalCost += prices[i];
    }
    return totalCost; // Total cost of the order
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the number of items: 5
Enter the prices of the items: 80
100
160
180
175
The total cost of the order is: 695.00
```

6. Create a function that takes an item's price and a refund percentage as input. The function should return the refund amount without changing the original item's price.

Function Prototype: float calculateRefund(**float** itemPrice, **float** refundPercentage);

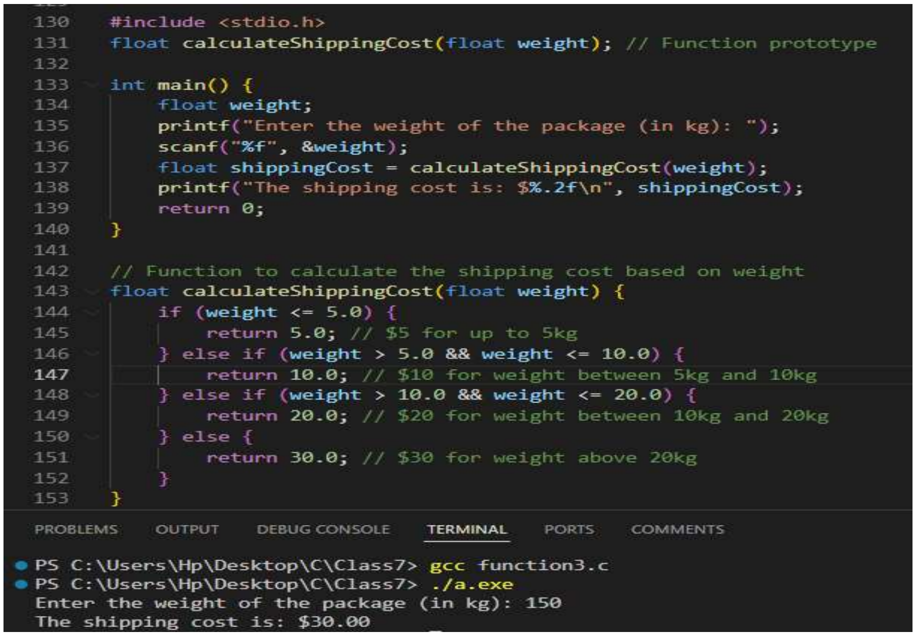


```
#include <stdio.h>
float calculateRefund(float itemPrice, float refundPercentage);

int main(){
    float  itemPrice,refundPercentage;
    printf("Enter the item's price and refund percentage:  ");
    scanf("%f %f",&itemPrice,&refundPercentage);
    float refund_Amount = calculateRefund(itemPrice,refundPercentage);
    printf("\nThe refund Amount is %.2f \n ",refund_Amount);
}

float calculateRefund(float itemPrice, float refundPercentage){
    return (itemPrice * refundPercentage)/100 ;
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the item's price and refund percentage:  1500 25

The refund Amount is 375.00
```

7. Implement a function that takes the weight of a package and calculates shipping costs based on weight brackets (e.g., $5 for up to 5kg, $10 for 5-10kg). The original weight should remain unchanged.

Function Prototype: float calculateShippingCost(**float** weight);

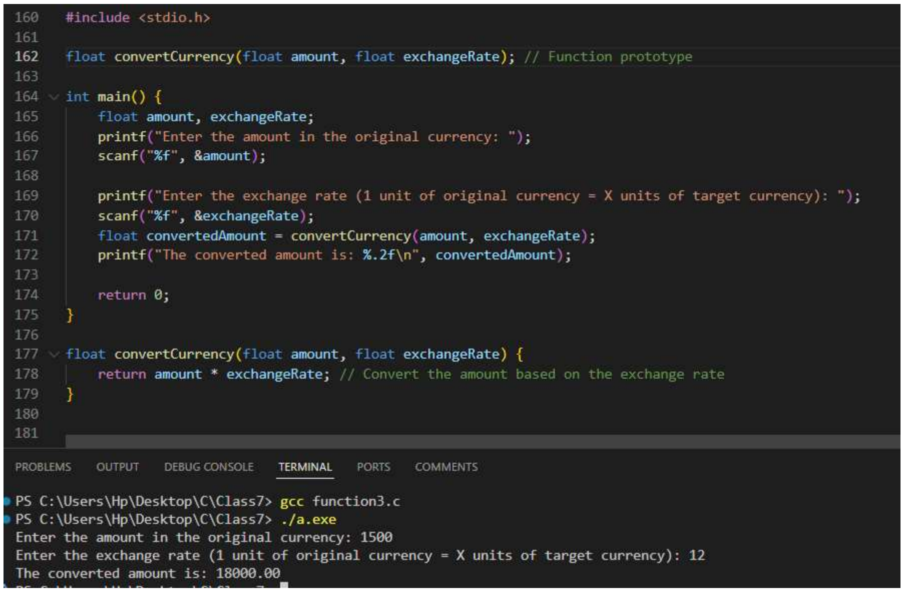


```c
#include <stdio.h>
float calculateShippingCost(float weight); // Function prototype

int main() {
    float weight;
    printf("Enter the weight of the package (in kg): ");
    scanf("%f", &weight);
    float shippingCost = calculateShippingCost(weight);
    printf("The shipping cost is: $%.2f\n", shippingCost);
    return 0;
}

// Function to calculate the shipping cost based on weight
float calculateShippingCost(float weight) {
    if (weight <= 5.0) {
        return 5.0; // $5 for up to 5kg
    } else if (weight > 5.0 && weight <= 10.0) {
        return 10.0; // $10 for weight between 5kg and 10kg
    } else if (weight > 10.0 && weight <= 20.0) {
        return 20.0; // $20 for weight between 10kg and 20kg
    } else {
        return 30.0; // $30 for weight above 20kg
    }
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the weight of the package (in kg): 150
The shipping cost is: $30.00
```

8. Design a function that converts an amount from one currency to another based on an exchange rate provided as input. The original amount should not be altered.

Function Prototype: float convertCurrency(**float** amount, **float** exchangeRate);



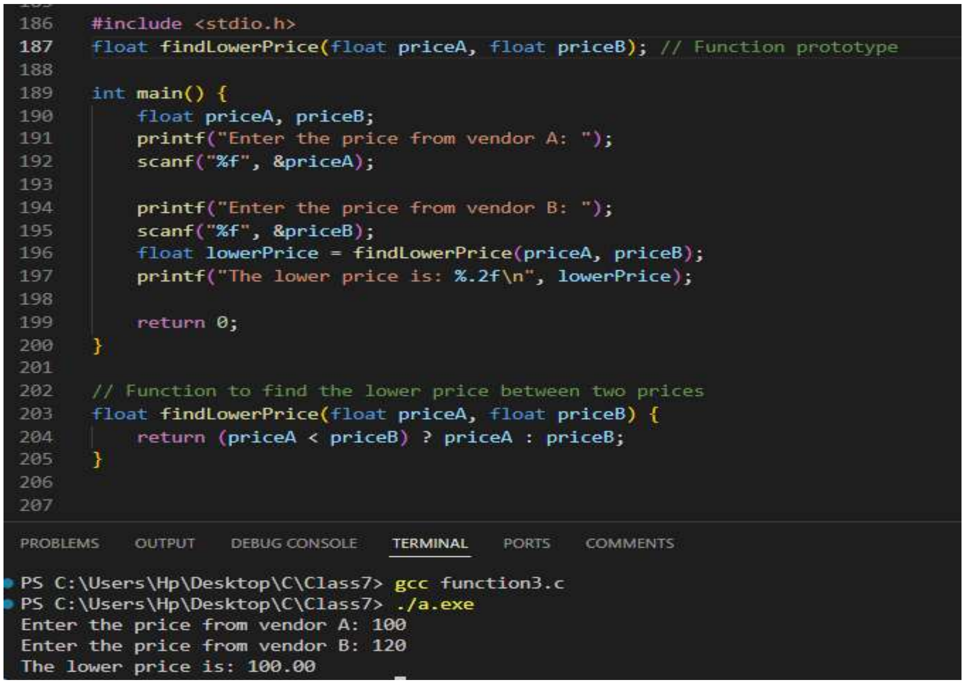
```c
#include <stdio.h>

float convertCurrency(float amount, float exchangeRate); // Function prototype

int main() {
    float amount, exchangeRate;
    printf("Enter the amount in the original currency: ");
    scanf("%f", &amount);

    printf("Enter the exchange rate (1 unit of original currency = X units of target currency): ");
    scanf("%f", &exchangeRate);
    float convertedAmount = convertCurrency(amount, exchangeRate);
    printf("The converted amount is: %.2f\n", convertedAmount);

    return 0;
}

float convertCurrency(float amount, float exchangeRate) {
    return amount * exchangeRate; // Convert the amount based on the exchange rate
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the amount in the original currency: 1500
Enter the exchange rate (1 unit of original currency = X units of target currency): 12
The converted amount is: 18000.00
```

9. Write a function that takes two prices from different vendors and returns the lower price without modifying either input price.

Function Prototype:

float findLowerPrice(**float** priceA, **float** priceB);

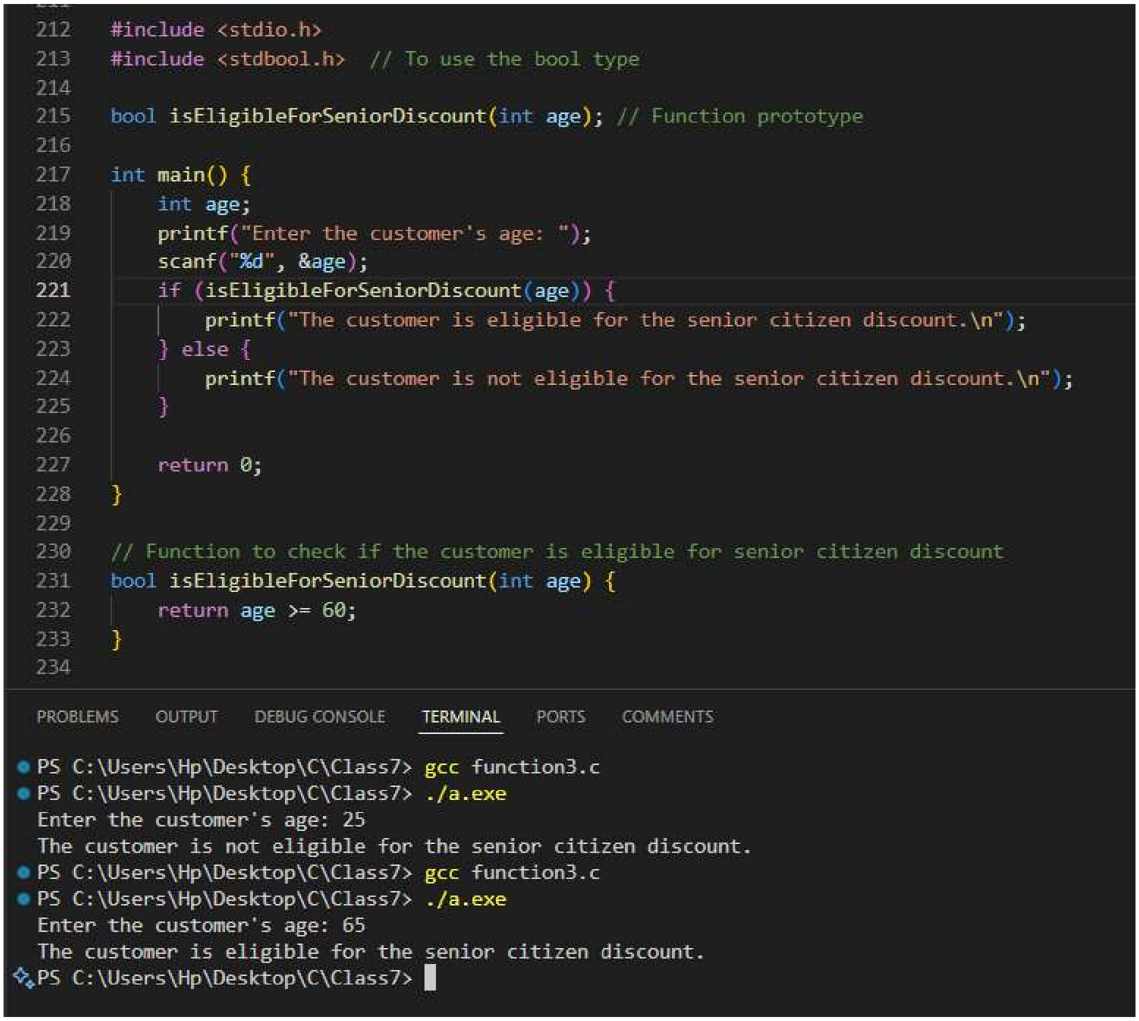


```c
#include <stdio.h>
float findLowerPrice(float priceA, float priceB); // Function prototype

int main() {
    float priceA, priceB;
    printf("Enter the price from vendor A: ");
    scanf("%f", &priceA);

    printf("Enter the price from vendor B: ");
    scanf("%f", &priceB);
    float lowerPrice = findLowerPrice(priceA, priceB);
    printf("The lower price is: %.2f\n", lowerPrice);

    return 0;
}

// Function to find the lower price between two prices
float findLowerPrice(float priceA, float priceB) {
    return (priceA < priceB) ? priceA : priceB;
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the price from vendor A: 100
Enter the price from vendor B: 120
The lower price is: 100.00
```

10. Create a function that checks if a customer is eligible for a senior citizen discount based on their age. The function should take age as input and return whether they qualify without changing the age value.

Function Prototype: bool isEligibleForSeniorDiscount(**int** age);



```c
#include <stdio.h>
#include <stdbool.h>  // To use the bool type

bool isEligibleForSeniorDiscount(int age); // Function prototype

int main() {
    int age;
    printf("Enter the customer's age: ");
    scanf("%d", &age);
    if (isEligibleForSeniorDiscount(age)) {
        printf("The customer is eligible for the senior citizen discount.\n");
    } else {
        printf("The customer is not eligible for the senior citizen discount.\n");
    }

    return 0;
}

// Function to check if the customer is eligible for senior citizen discount
bool isEligibleForSeniorDiscount(int age) {
    return age >= 60;
}
```

```
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the customer's age: 25
The customer is not eligible for the senior citizen discount.
PS C:\Users\Hp\Desktop\C\Class7> gcc function3.c
PS C:\Users\Hp\Desktop\C\Class7> ./a.exe
Enter the customer's age: 65
The customer is eligible for the senior citizen discount.
PS C:\Users\Hp\Desktop\C\Class7> 
```