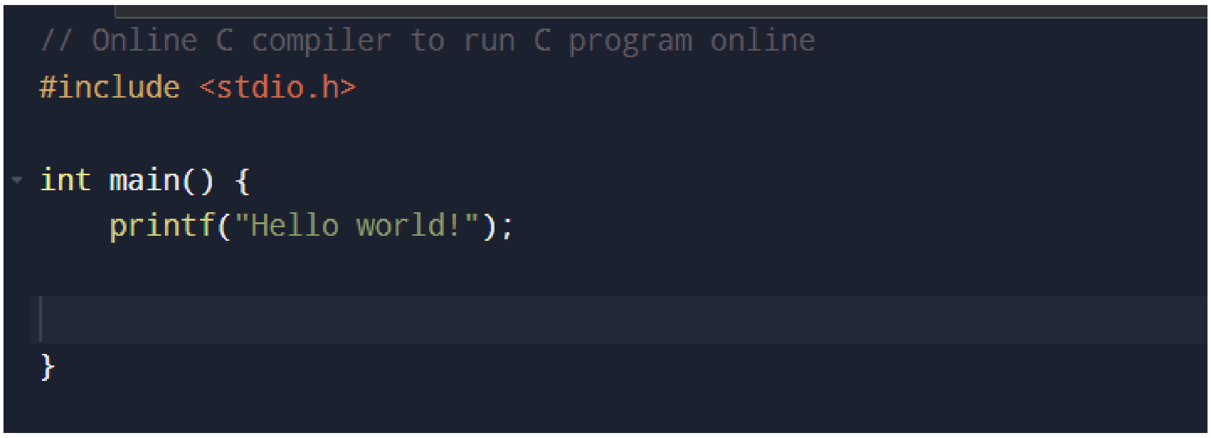
Lab -1

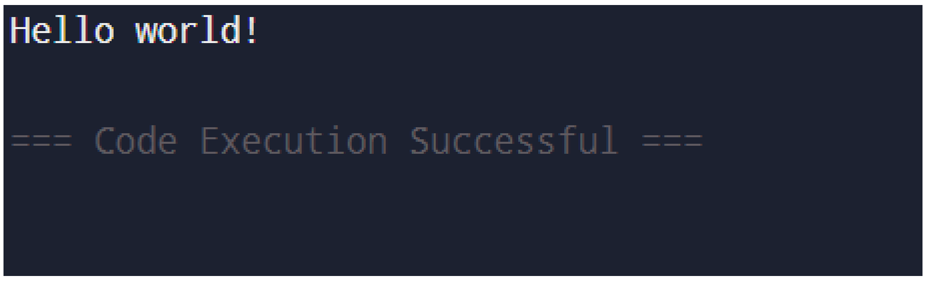
Basic C Programs

Q1. Print "Hello, World!"



```c
// Online C compiler to run C program online
#include <stdio.h>

int main() {
    printf("Hello world!");


}
```

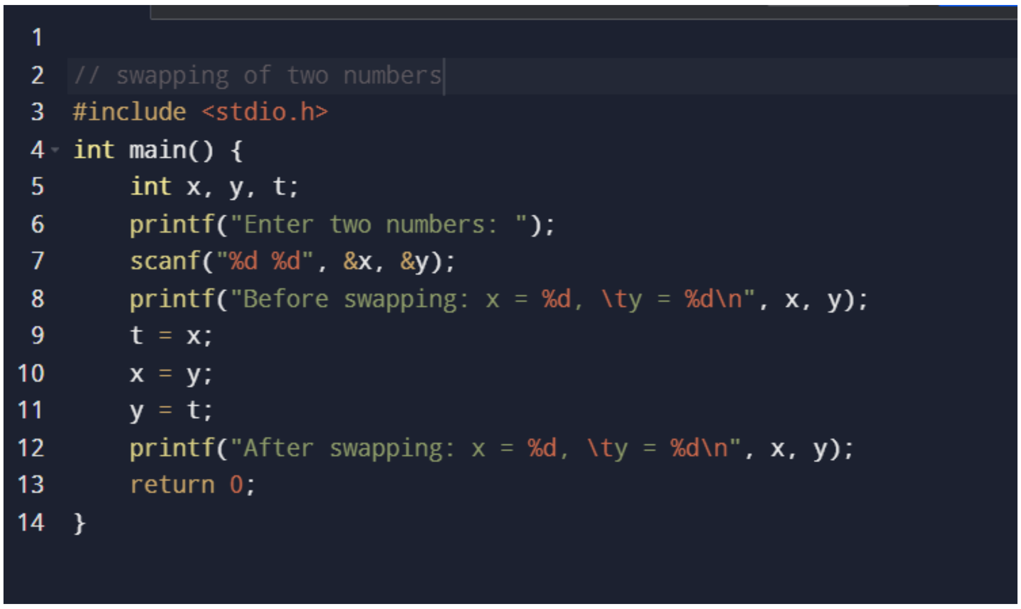
Output :



```
Hello world!


=== Code Execution Successful ===
```

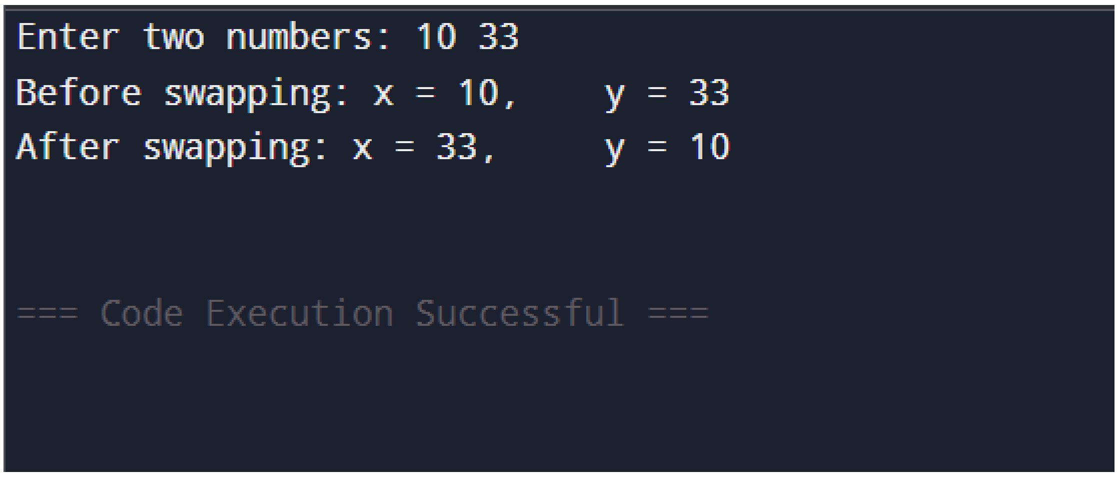
Q2. Swap Two Numbers

Scenario: A cashier mistakenly enters two values in reverse. Write a program to swap them.



```c

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

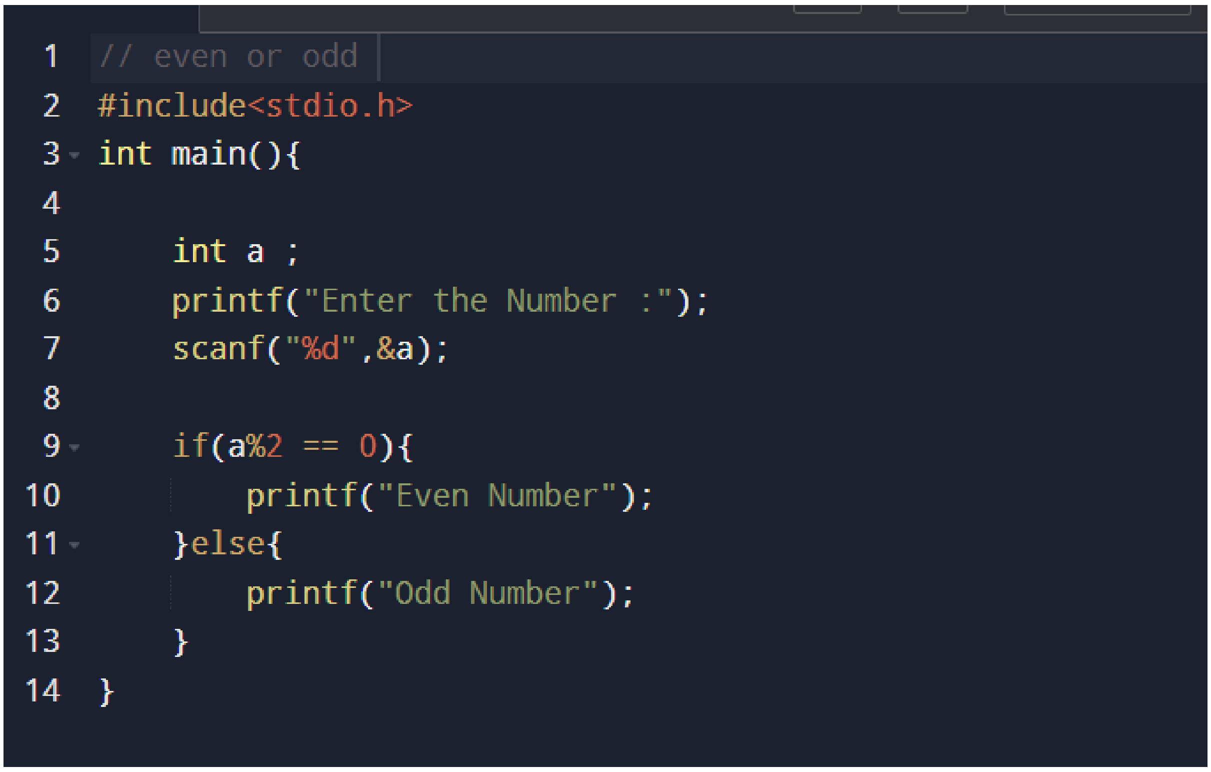
Output :



```
Enter two numbers: 10 33
Before swapping: x = 10,     y = 33
After swapping: x = 33,      y = 10


=== Code Execution Successful ===
```

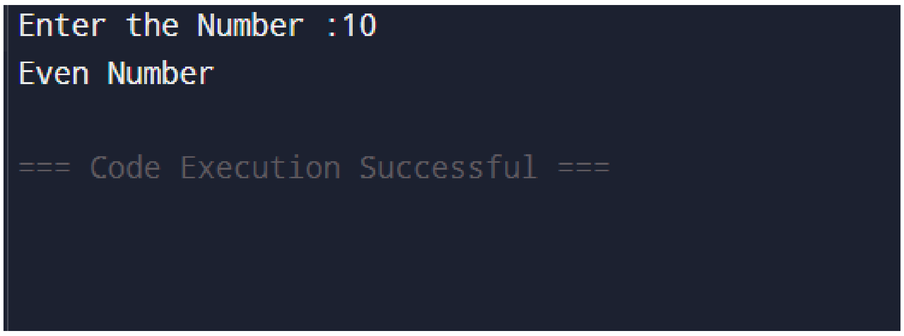
Q3. Check Even or Odd

Scenario: A billing machine checks if a customer’s token number is even or odd.



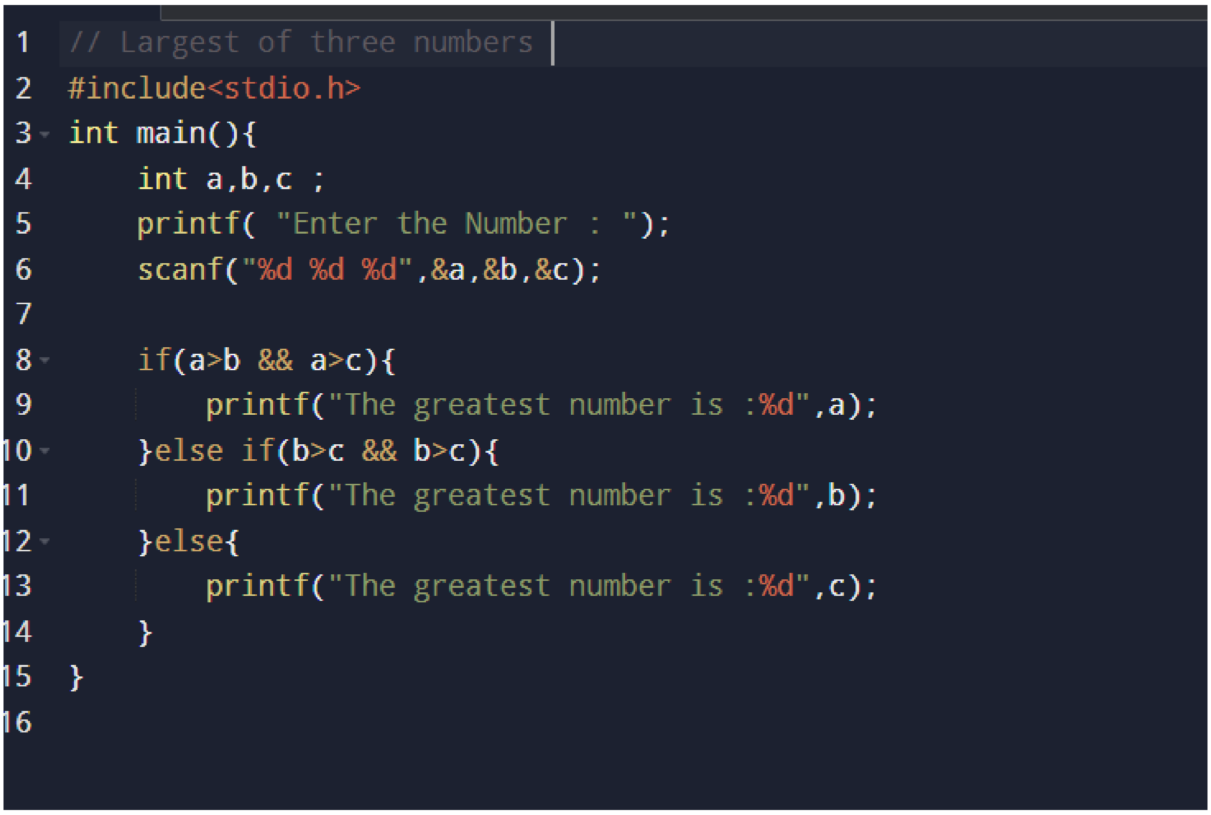
```c
// even or odd
#include<stdio.h>
int main(){

    int a ;
    printf("Enter the Number :");
    scanf("%d",&a);

    if(a%2 == 0){
        printf("Even Number");
    }else{
        printf("Odd Number");
    }
}
```

Output :



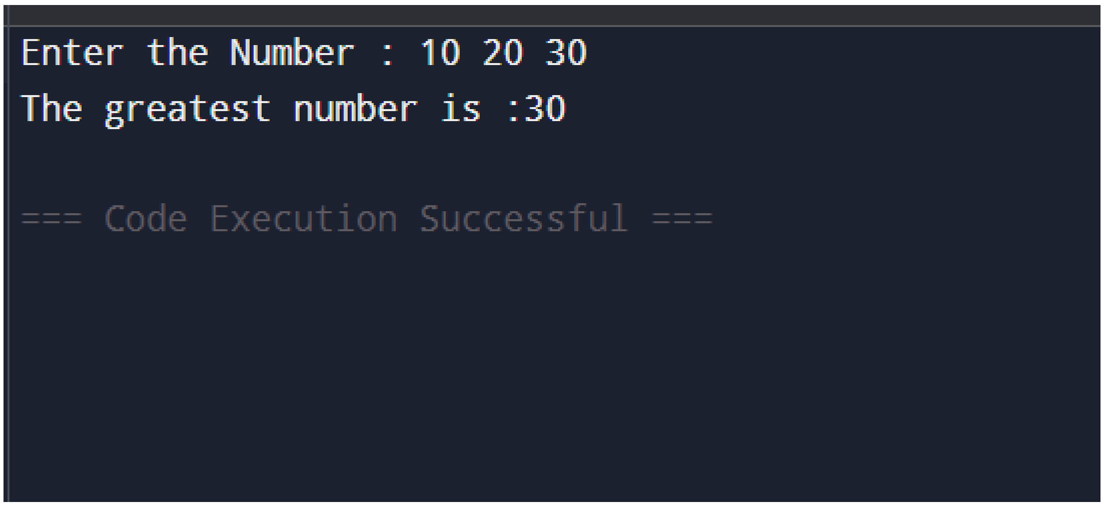
```
Enter the Number :10
Even Number

=== Code Execution Successful ===
```

Q4. Find Largest of Three Numbers



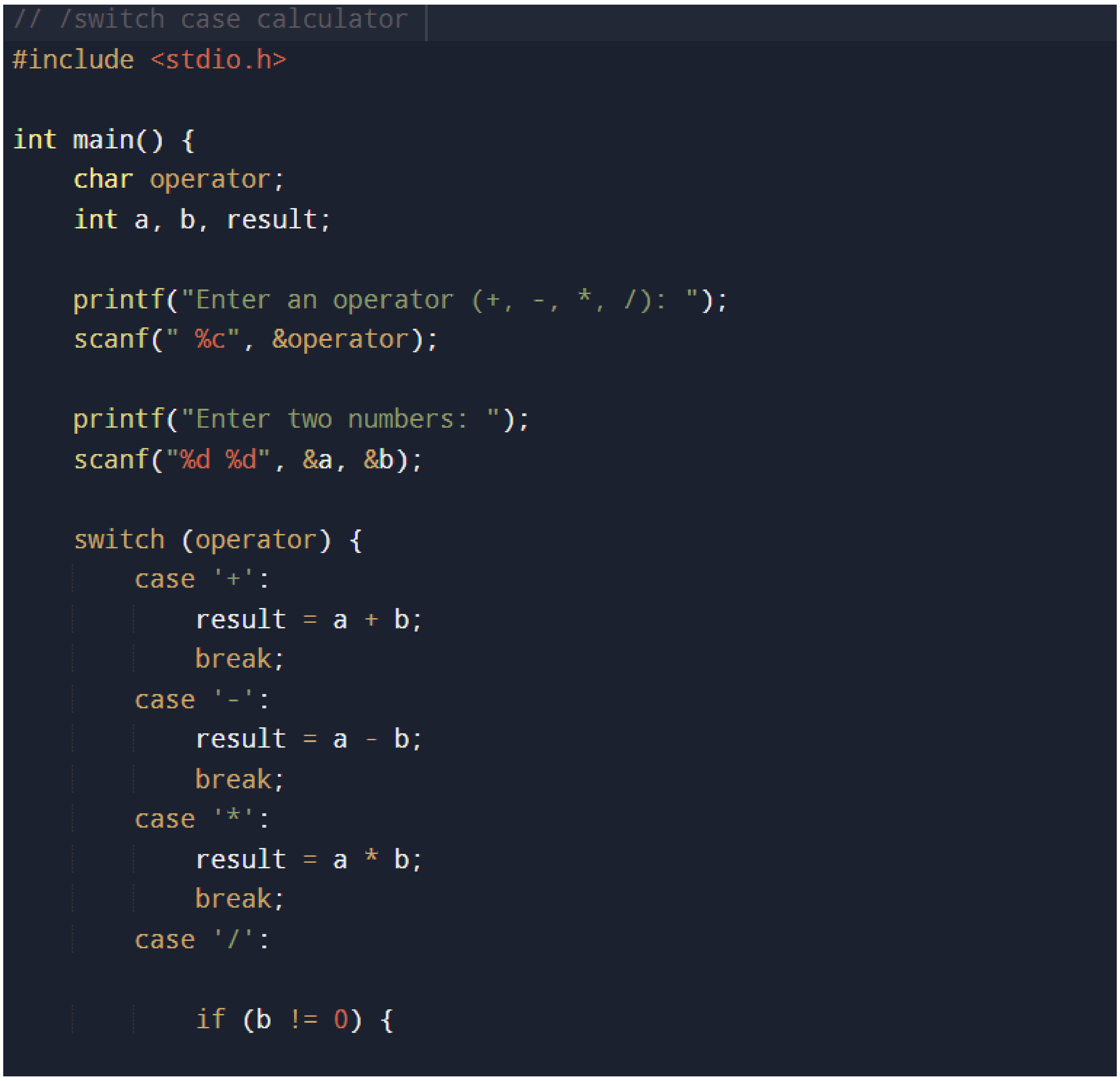
```c
// Largest of three numbers
#include<stdio.h>
int main(){
    int a,b,c ;
    printf( "Enter the Number : ");
    scanf("%d %d %d",&a,&b,&c);

    if(a>b && a>c){
        printf("The greatest number is :%d",a);
    }else if(b>c && b>c){
        printf("The greatest number is :%d",b);
    }else{
        printf("The greatest number is :%d",c);
    }
}

```

Output :



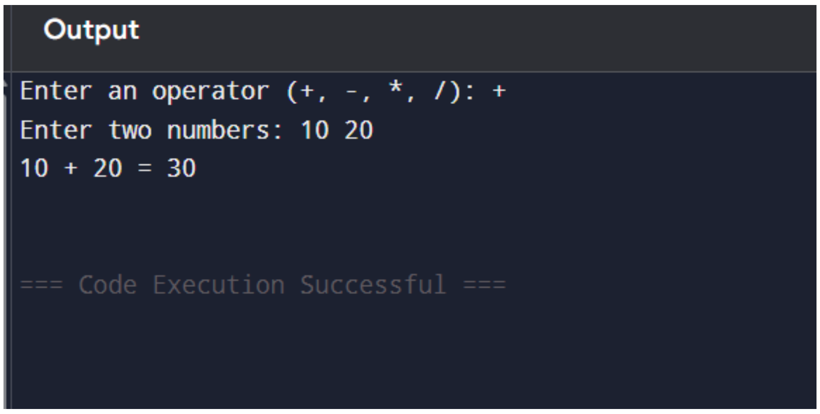
```
Enter the Number : 10 20 30
The greatest number is :30


=== Code Execution Successful ===
```

Q5. Simple Calculator (switch case)



```
// /switch case calculator
#include <stdio.h>

int main() {
    char operator;
    int a, b, result;

    printf("Enter an operator (+, -, *, /): ");
    scanf(" %c", &operator);

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    switch (operator) {
        case '+':
            result = a + b;
            break;
        case '-':
            result = a - b;
            break;
        case '*':
            result = a * b;
            break;
        case '/':

            if (b != 0) {
```

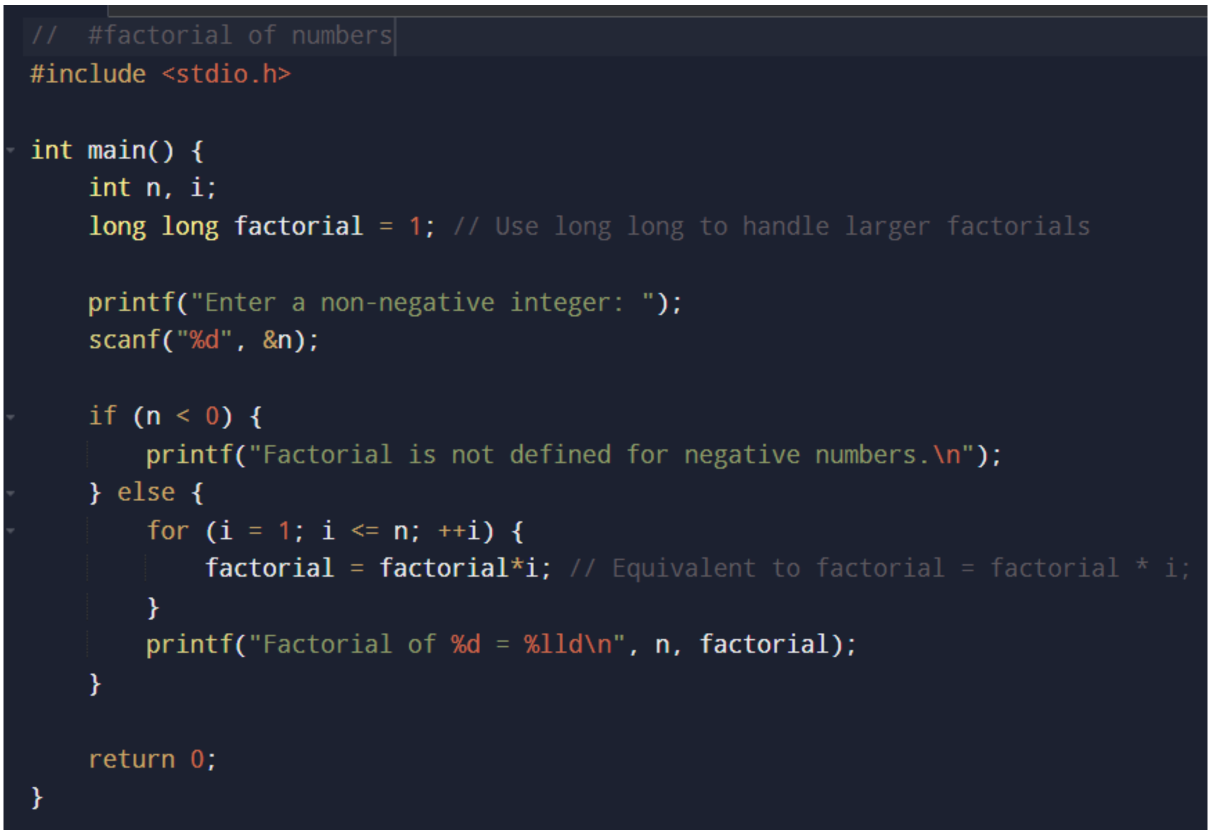
Output:



Output

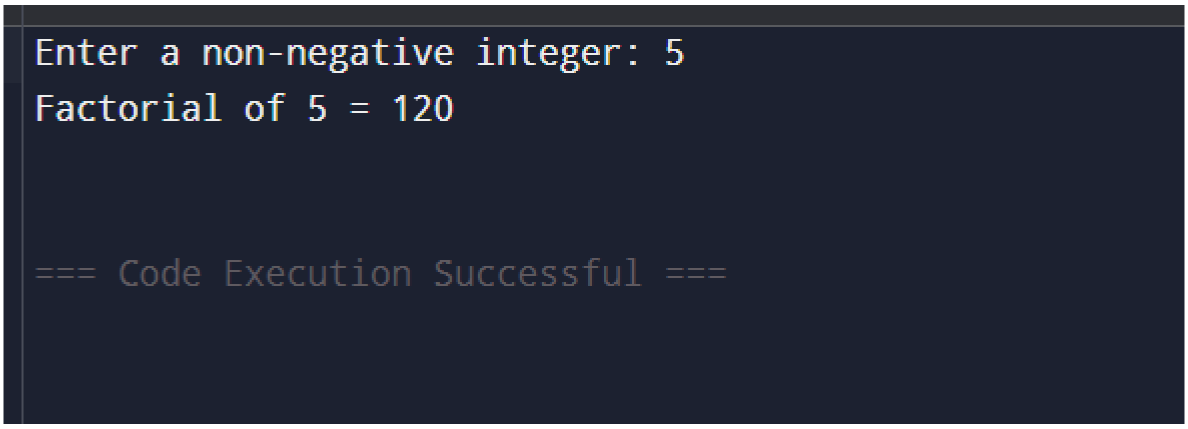
```
Enter an operator (+, -, *, /): +
Enter two numbers: 10 20
10 + 20 = 30


=== Code Execution Successful ===
```

Q6. Factorial of a Number



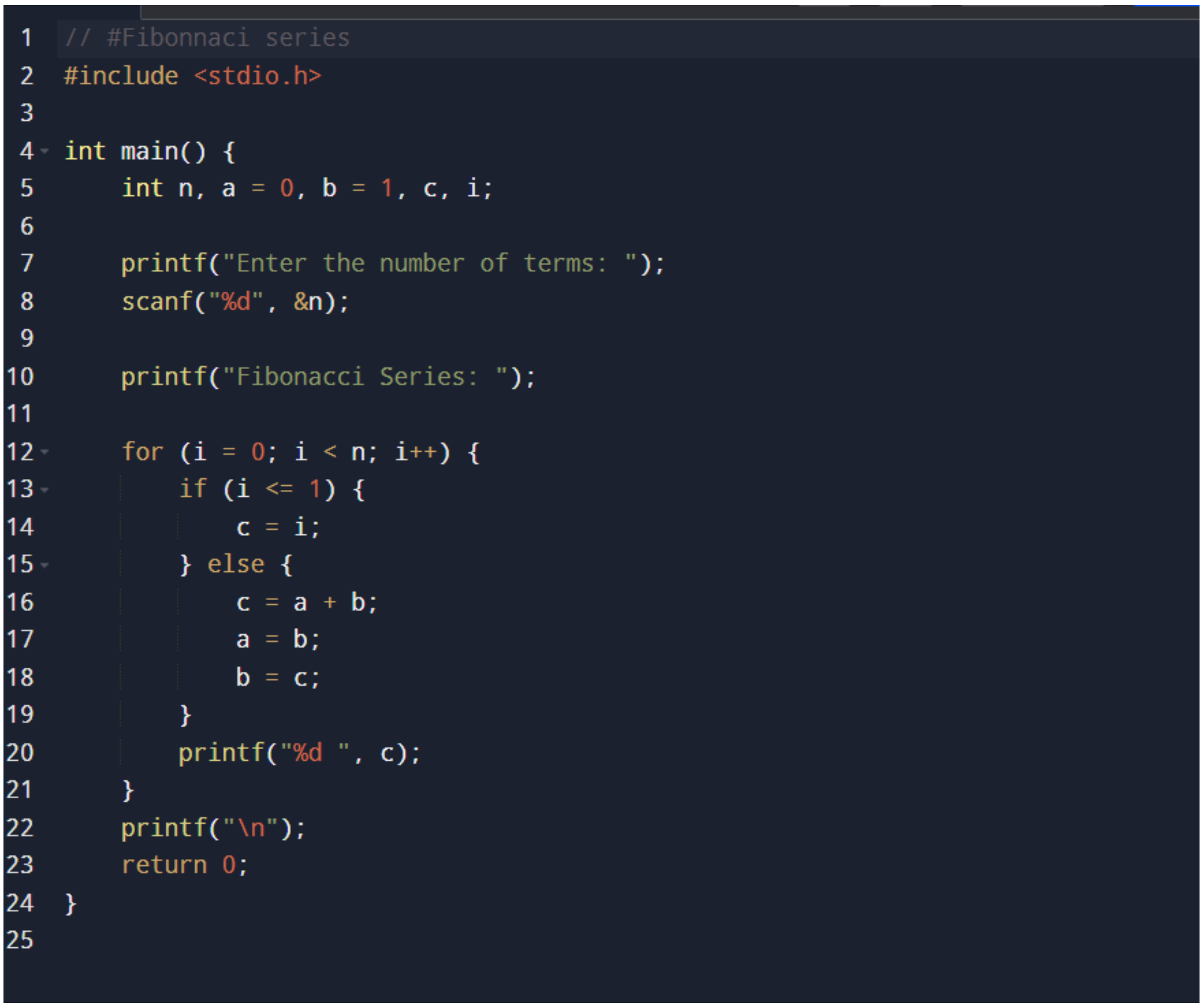
```c
//  #factorial of numbers
#include <stdio.h>

int main() {
    int n, i;
    long long factorial = 1; // Use long long to handle larger factorials

    printf("Enter a non-negative integer: ");
    scanf("%d", &n);

    if (n < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    } else {
        for (i = 1; i <= n; ++i) {
            factorial = factorial*i; // Equivalent to factorial = factorial * i;
        }
        printf("Factorial of %d = %lld\n", n, factorial);
    }

    return 0;
}
```

Output:



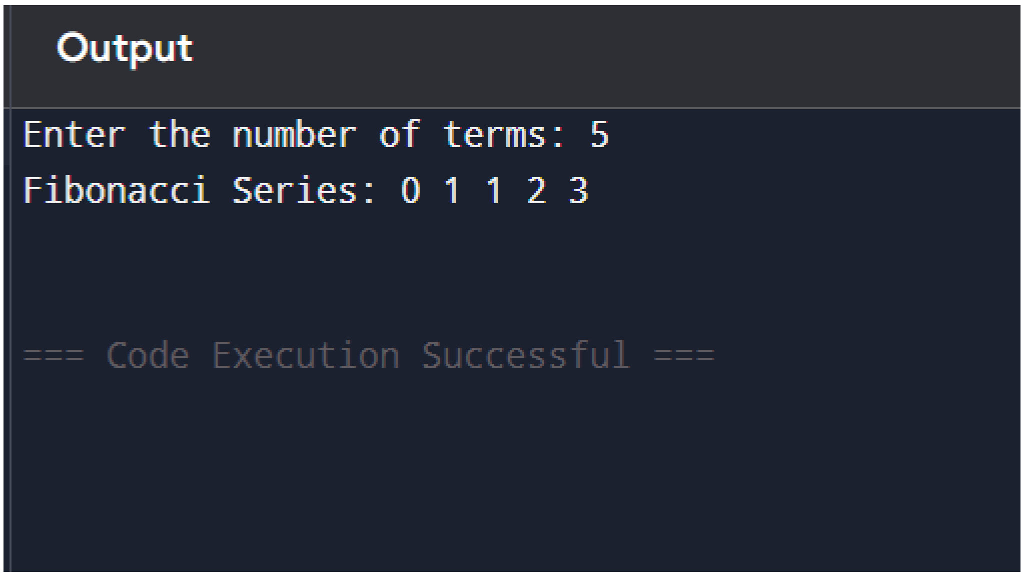
```
Enter a non-negative integer: 5
Factorial of 5 = 120


=== Code Execution Successful ===
```

Q7. Fibonacci Series (first n terms)



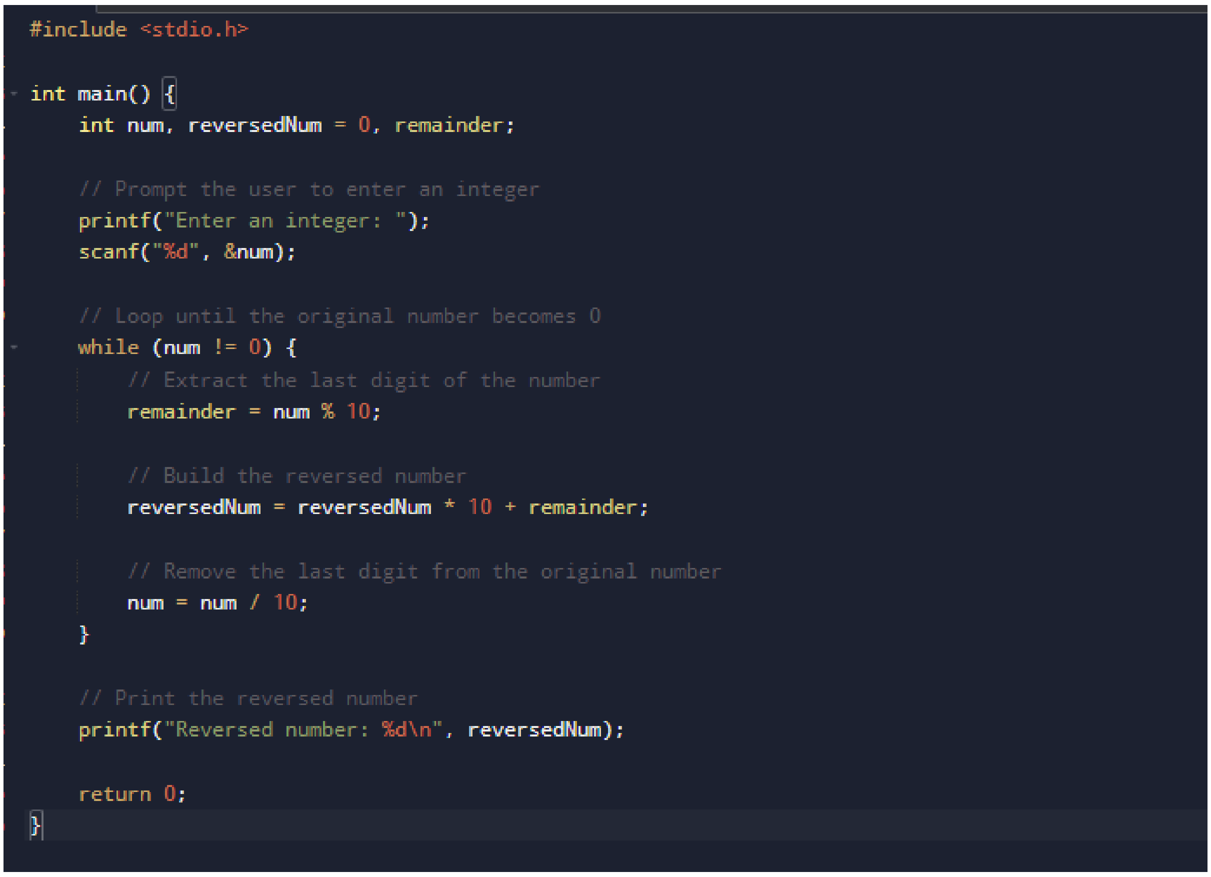
```c
// #Fibonnaci series
#include <stdio.h>

int main() {
    int n, a = 0, b = 1, c, i;

    printf("Enter the number of terms: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");

    for (i = 0; i < n; i++) {
        if (i <= 1) {
            c = i;
        } else {
            c = a + b;
            a = b;
            b = c;
        }
        printf("%d ", c);
    }
    printf("\n");
    return 0;
}

```

Output :



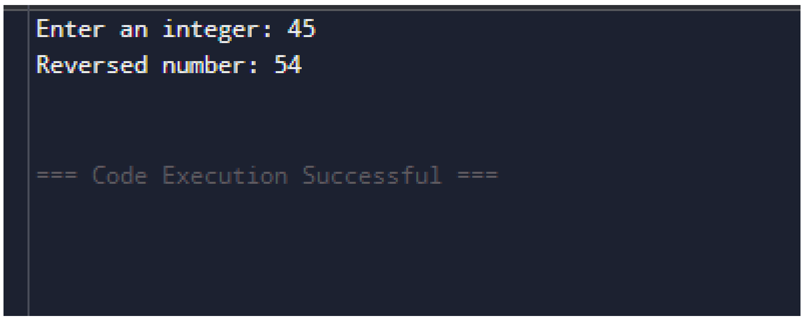
```
Enter the number of terms: 5
Fibonacci Series: 0 1 1 2 3


=== Code Execution Successful ===
```

Q8. Reverse a Number



```c
#include <stdio.h>

int main() {
    int num, reversedNum = 0, remainder;

    // Prompt the user to enter an integer
    printf("Enter an integer: ");
    scanf("%d", &num);

    // Loop until the original number becomes 0
    while (num != 0) {
        // Extract the last digit of the number
        remainder = num % 10;

        // Build the reversed number
        reversedNum = reversedNum * 10 + remainder;

        // Remove the last digit from the original number
        num = num / 10;
    }

    // Print the reversed number
    printf("Reversed number: %d\n", reversedNum);

    return 0;
}
```

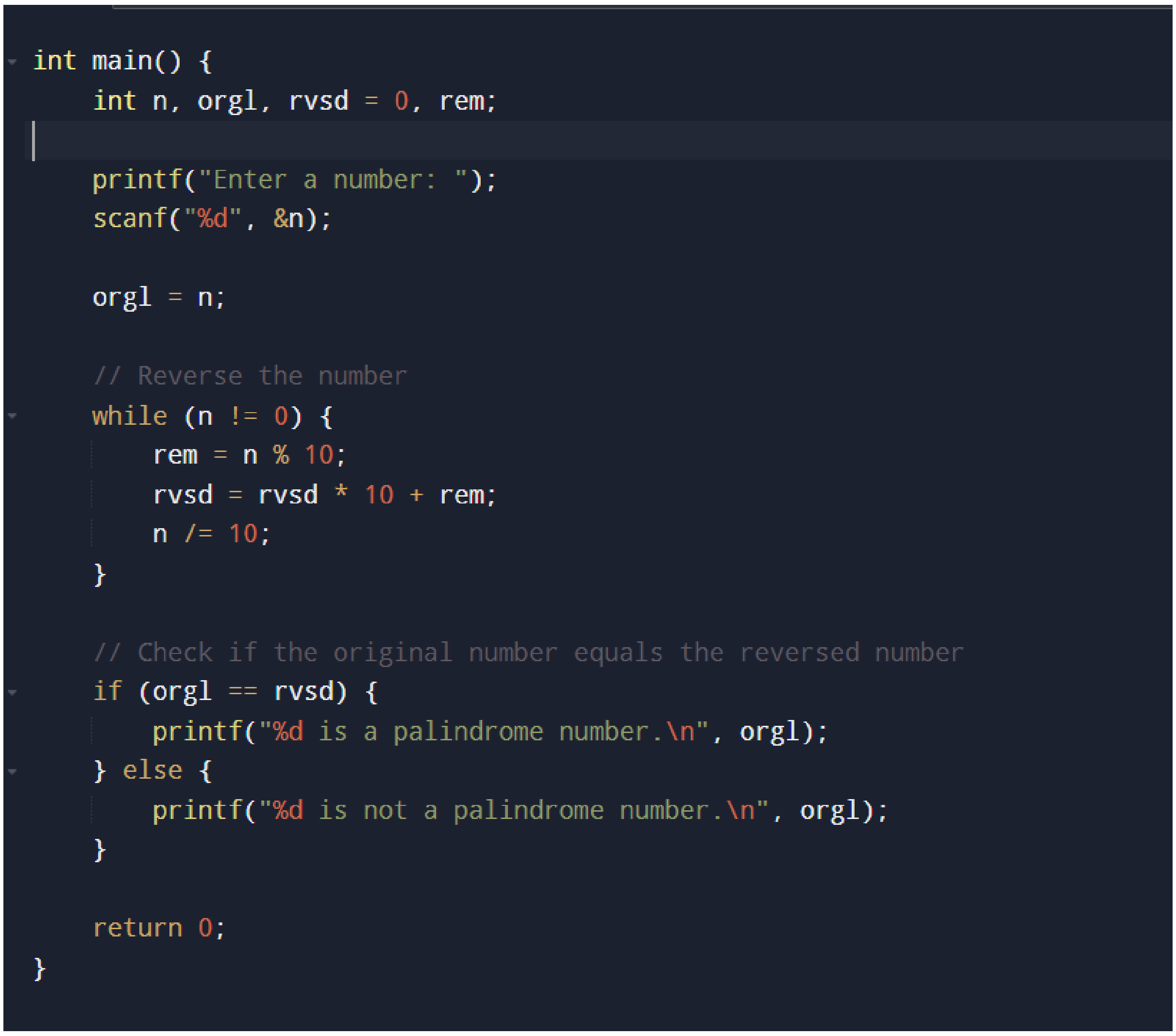
Output :



```
Enter an integer: 45
Reversed number: 54


=== Code Execution Successful ===
```

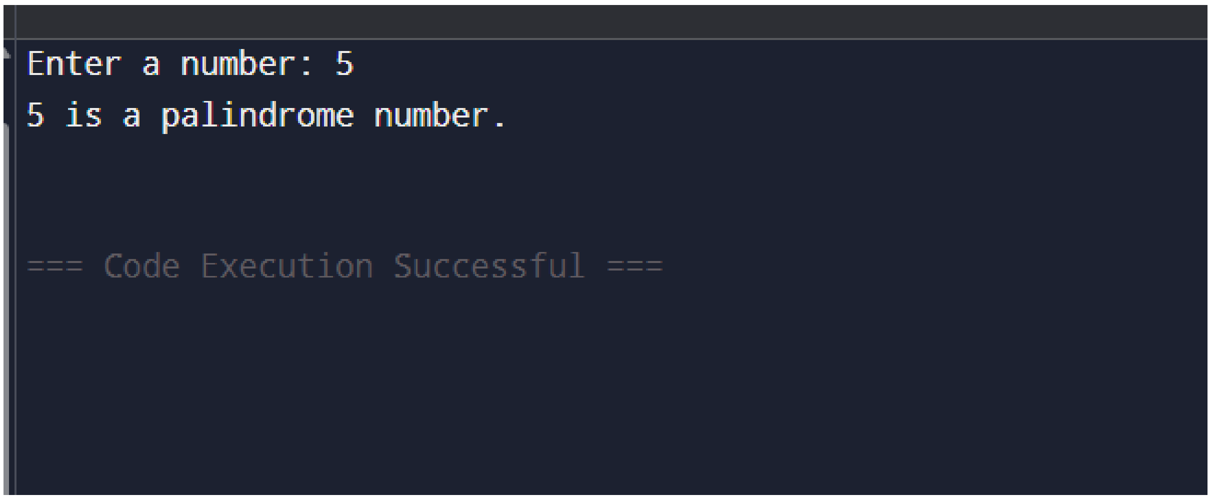
Q9. Palindrome Number Check

Scenario: ATM checks if PIN entered forward = reverse.



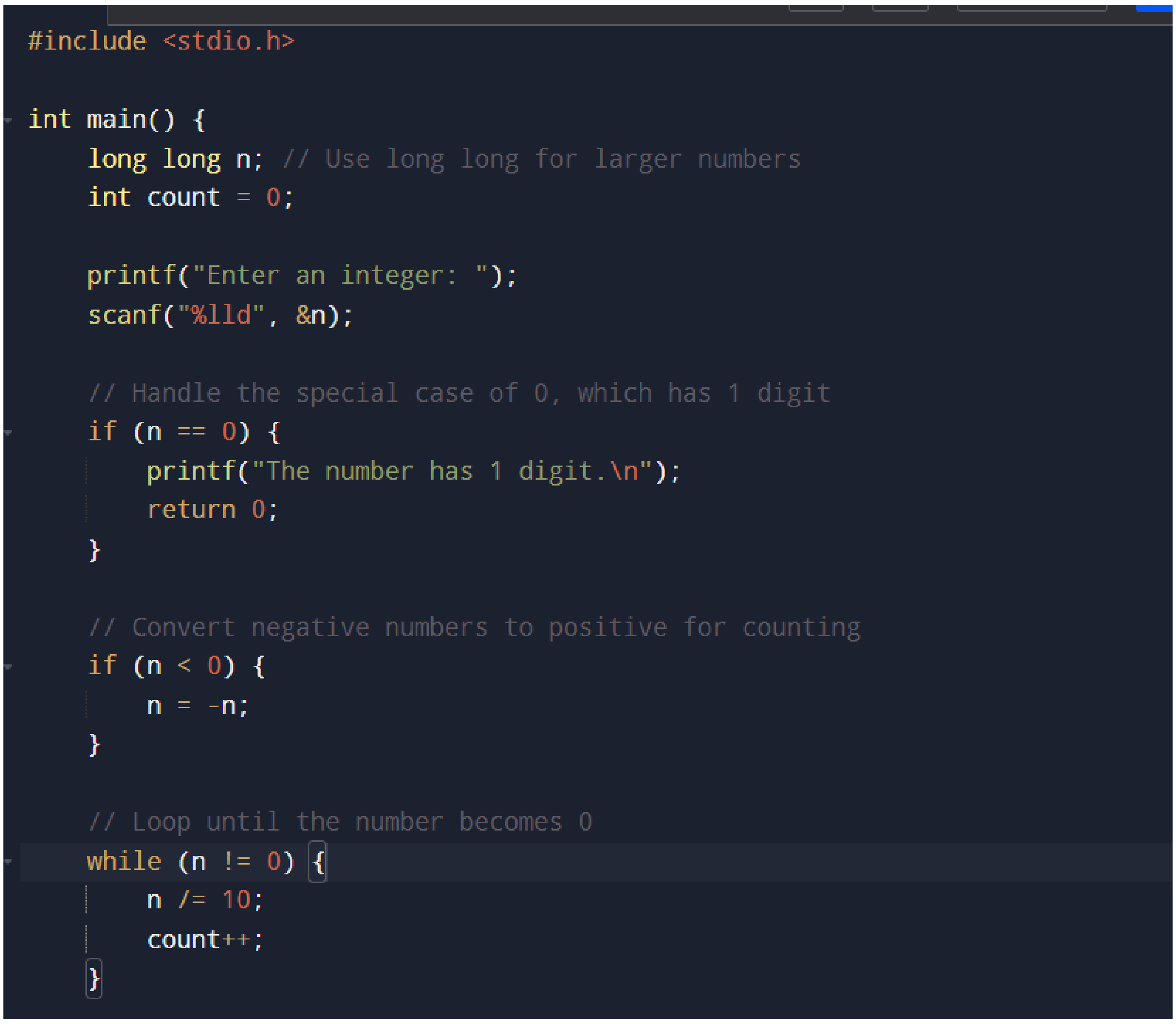
```c
int main() {
    int n, orgl, rvsd = 0, rem;

    printf("Enter a number: ");
    scanf("%d", &n);

    orgl = n;

    // Reverse the number
    while (n != 0) {
        rem = n % 10;
        rvsd = rvsd * 10 + rem;
        n /= 10;
    }

    // Check if the original number equals the reversed number
    if (orgl == rvsd) {
        printf("%d is a palindrome number.\n", orgl);
    } else {
        printf("%d is not a palindrome number.\n", orgl);
    }

    return 0;
}
```

Output:



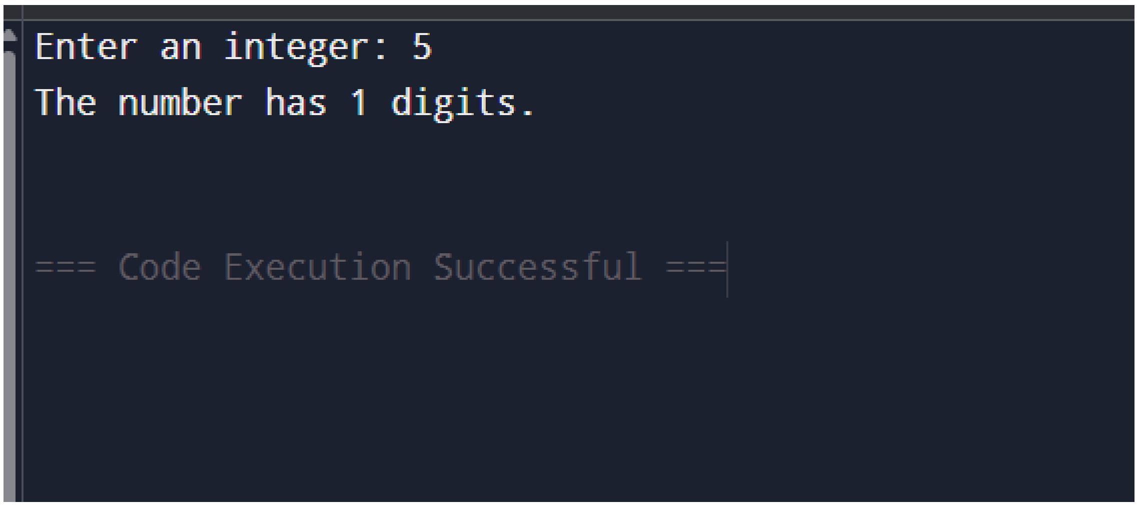
```
Enter a number: 5
5 is a palindrome number.


=== Code Execution Successful ===
```

Q10. Count Digits in a Number



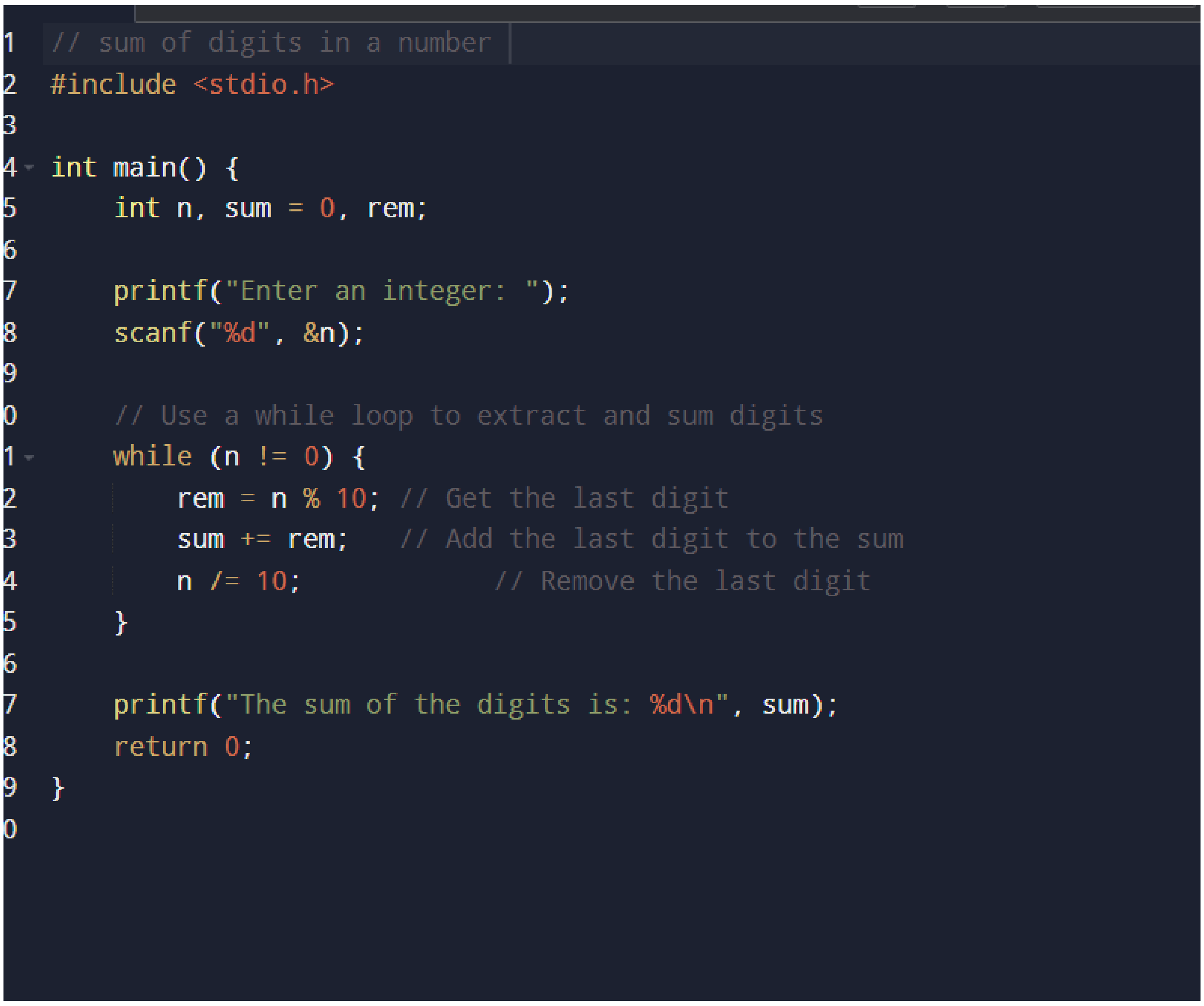
```c
#include <stdio.h>

int main() {
    long long n; // Use long long for larger numbers
    int count = 0;

    printf("Enter an integer: ");
    scanf("%lld", &n);

    // Handle the special case of 0, which has 1 digit
    if (n == 0) {
        printf("The number has 1 digit.\n");
        return 0;
    }

    // Convert negative numbers to positive for counting
    if (n < 0) {
        n = -n;
    }

    // Loop until the number becomes 0
    while (n != 0) {
        n /= 10;
        count++;
    }
```

Output:



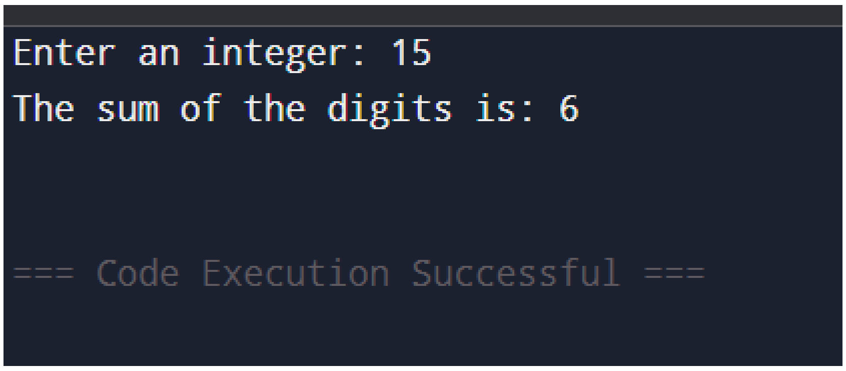
```
Enter an integer: 5
The number has 1 digits.


=== Code Execution Successful ===
```

Q11. Sum of Digits



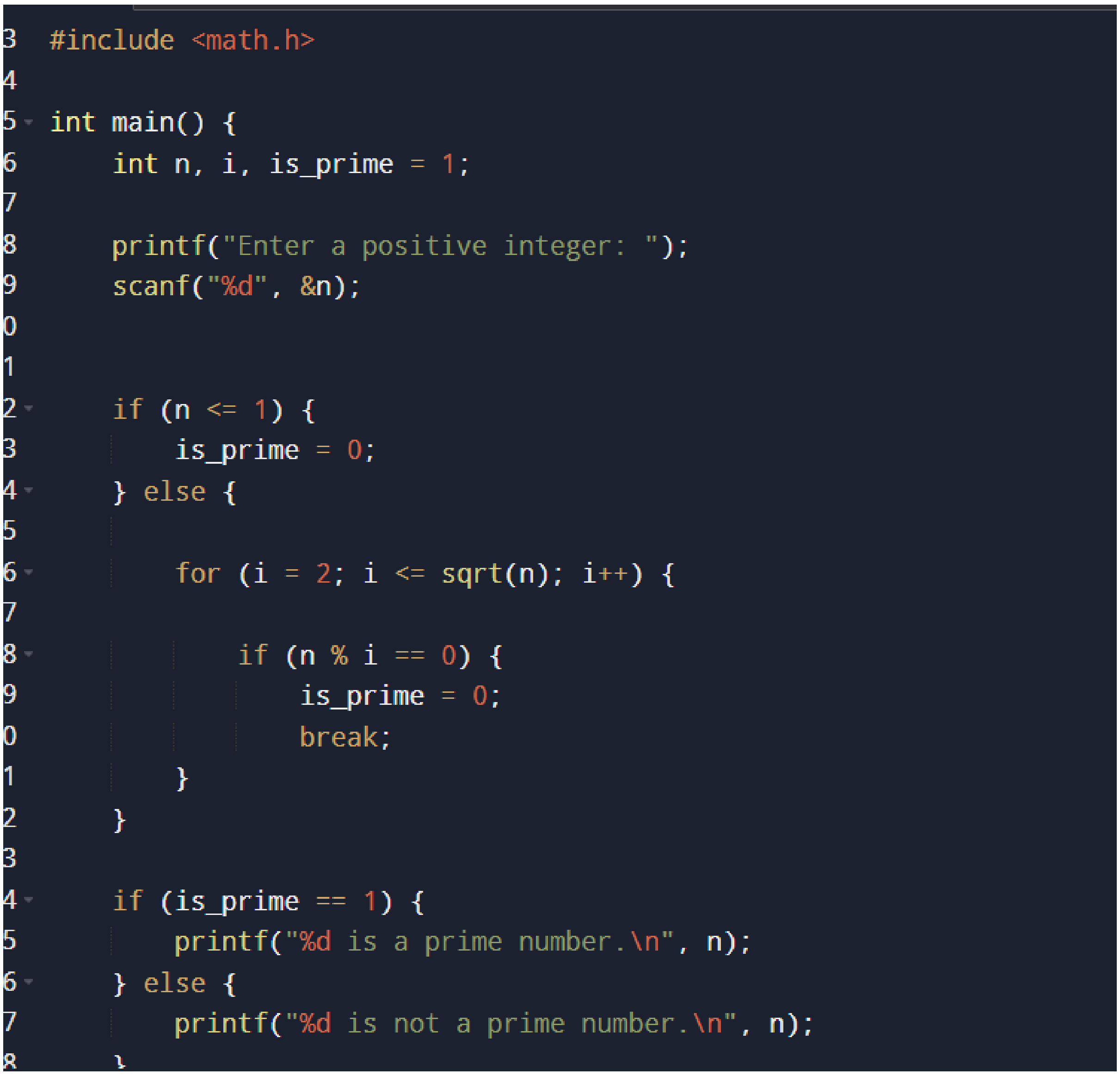
```c
// sum of digits in a number
#include <stdio.h>

int main() {
    int n, sum = 0, rem;

    printf("Enter an integer: ");
    scanf("%d", &n);

    // Use a while loop to extract and sum digits
    while (n != 0) {
        rem = n % 10; // Get the last digit
        sum += rem;   // Add the last digit to the sum
        n /= 10;            // Remove the last digit
    }

    printf("The sum of the digits is: %d\n", sum);
    return 0;
}
```

Output:



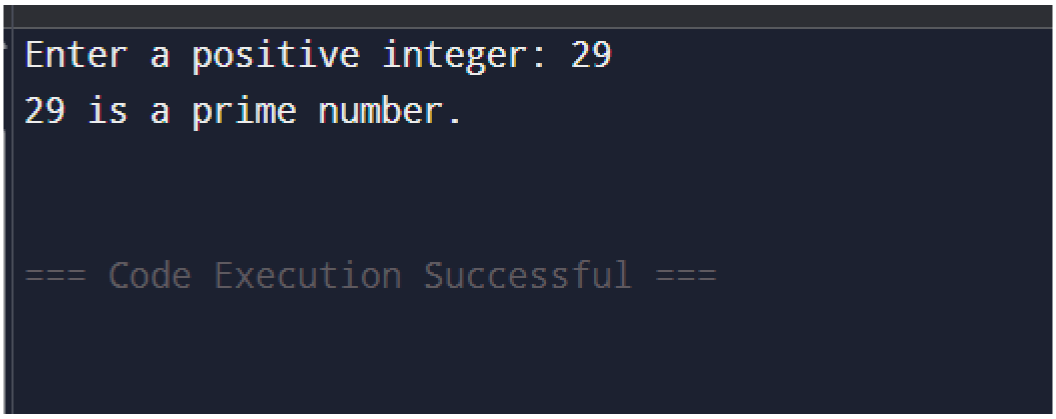
```
Enter an integer: 15
The sum of the digits is: 6


=== Code Execution Successful ===
```

Q12. Check Prime Number



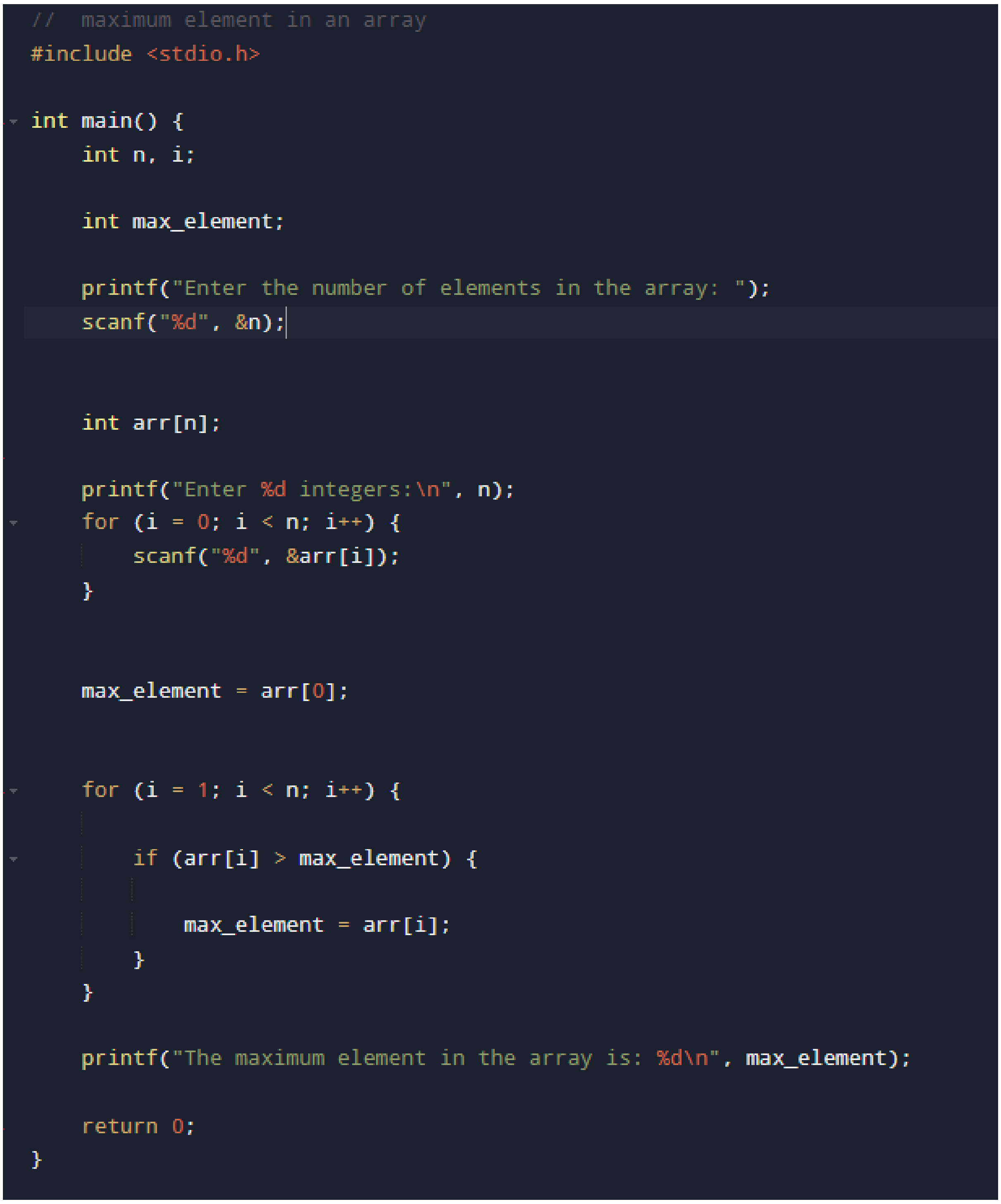
```c
#include <math.h>

int main() {
    int n, i, is_prime = 1;

    printf("Enter a positive integer: ");
    scanf("%d", &n);


    if (n <= 1) {
        is_prime = 0;
    } else {

        for (i = 2; i <= sqrt(n); i++) {

            if (n % i == 0) {
                is_prime = 0;
                break;
        }
    }

    if (is_prime == 1) {
        printf("%d is a prime number.\n", n);
    } else {
        printf("%d is not a prime number.\n", n);
    }
```

Output:



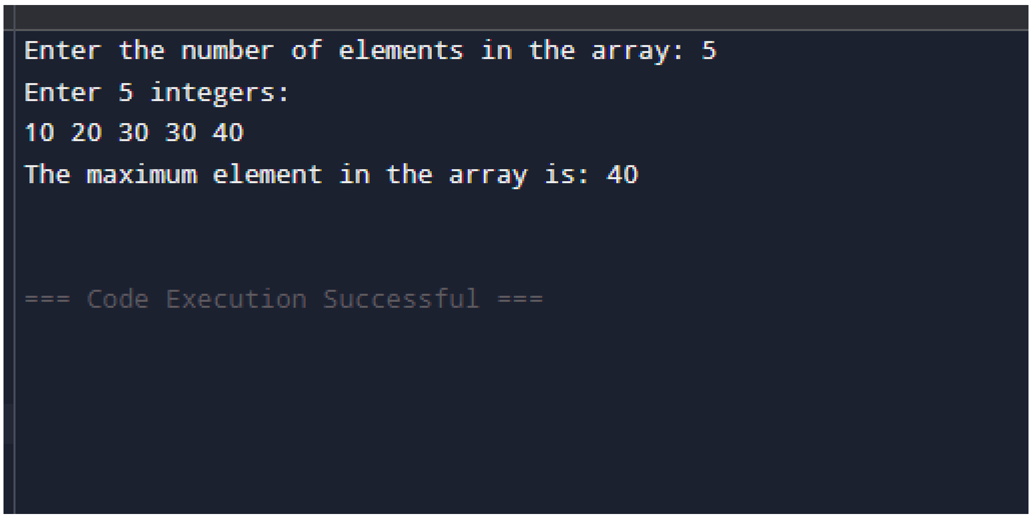
```
Enter a positive integer: 29
29 is a prime number.



=== Code Execution Successful ===
```

Q13. Array – Find Maximum Element



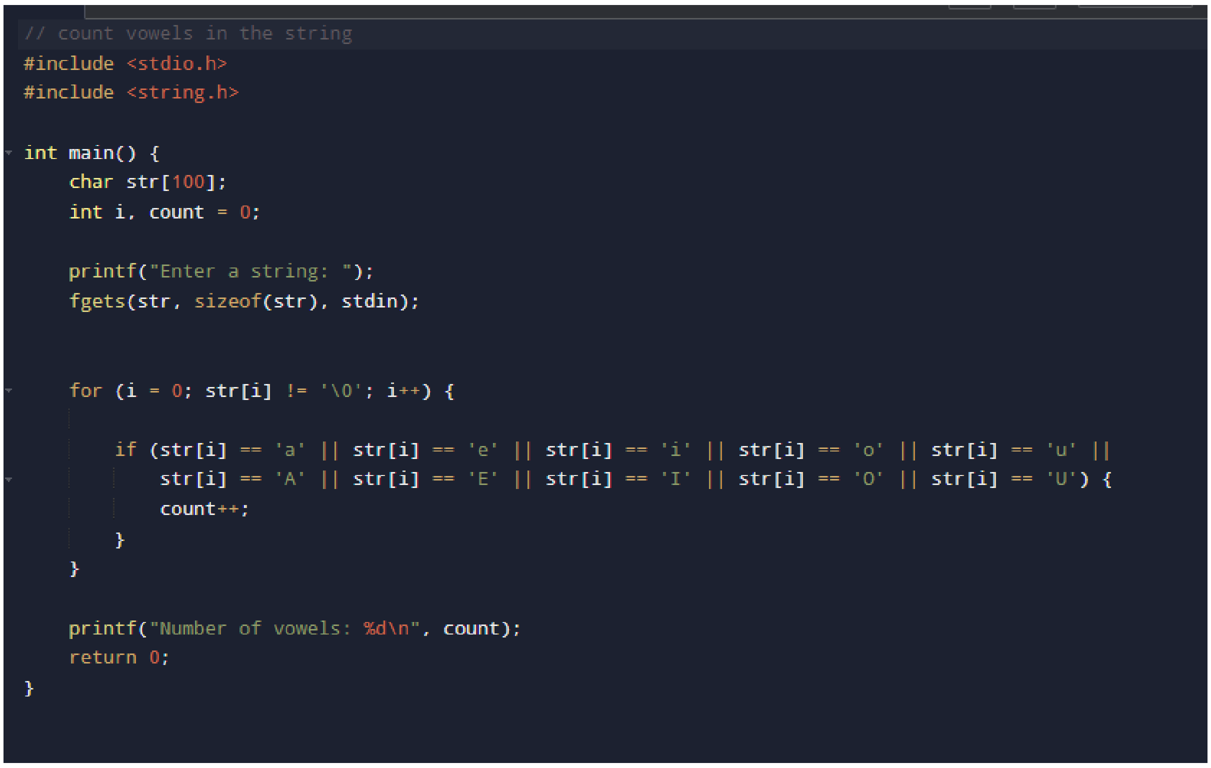
```c
//  maximum element in an array
#include <stdio.h>

int main() {
    int n, i;

    int max_element;

    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);


    int arr[n];

    printf("Enter %d integers:\n", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }


    max_element = arr[0];


    for (i = 1; i < n; i++) {

        if (arr[i] > max_element) {

            max_element = arr[i];
        }
    }

    printf("The maximum element in the array is: %d\n", max_element);

    return 0;
}
```

Output :



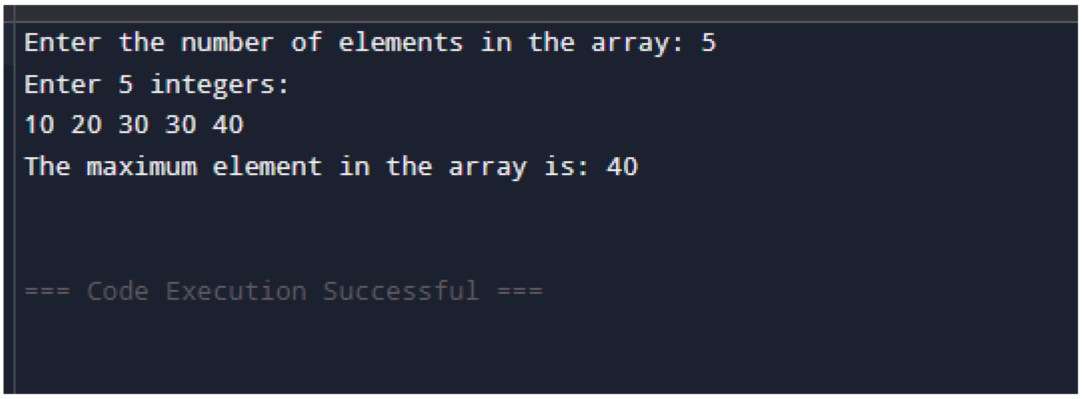
```
Enter the number of elements in the array: 5
Enter 5 integers:
10 20 30 30 40
The maximum element in the array is: 40


=== Code Execution Successful ===
```

Q14. String – Count Vowels



```c
// count vowels in the string
#include <stdio.h>
#include <string.h>

int main() {
    char str[100];
    int i, count = 0;

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);


    for (i = 0; str[i] != '\0'; i++) {

        if (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u' ||
            str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' || str[i] == 'U') {
            count++;
        }
    }

    printf("Number of vowels: %d\n", count);
    return 0;
}
```

Output :



```
Enter the number of elements in the array: 5
Enter 5 integers:
10 20 30 30 40
The maximum element in the array is: 40


=== Code Execution Successful ===
```

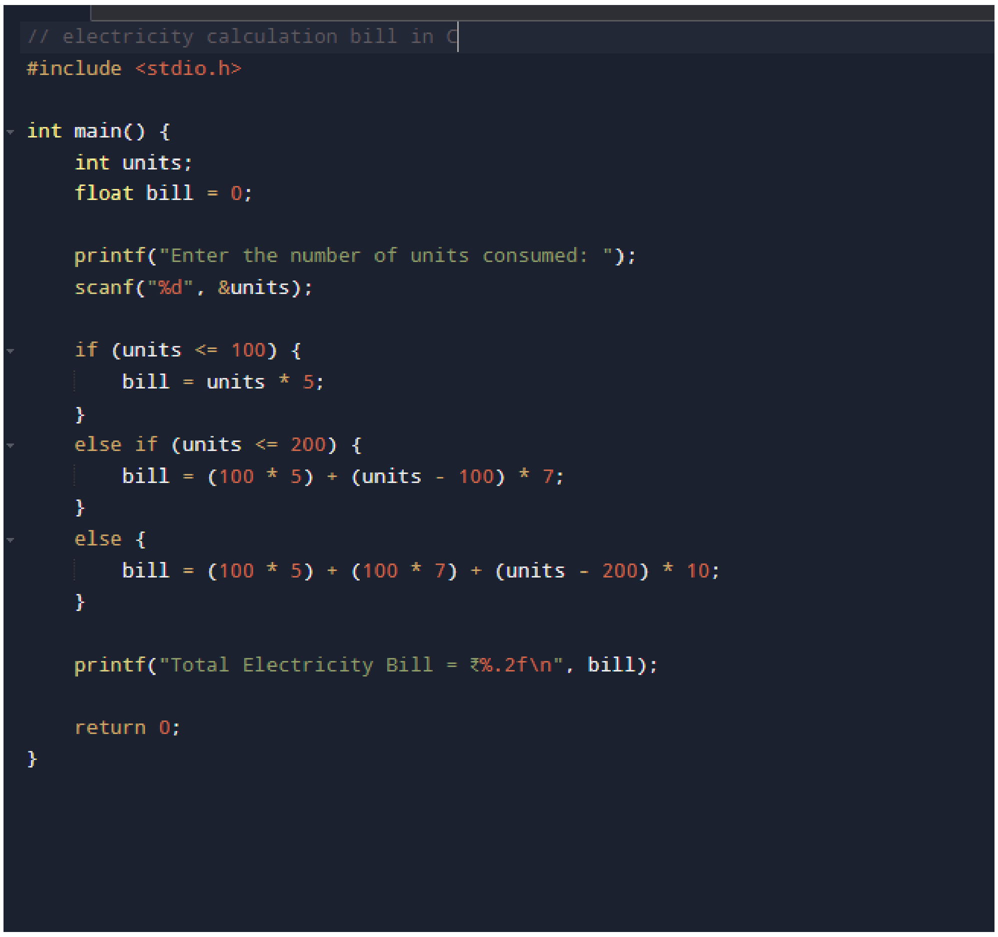
Q15. Scenario – Electricity Bill Calculation

Scenario: A company charges electricity bill as:

• For first 100 units: ₹5/unit

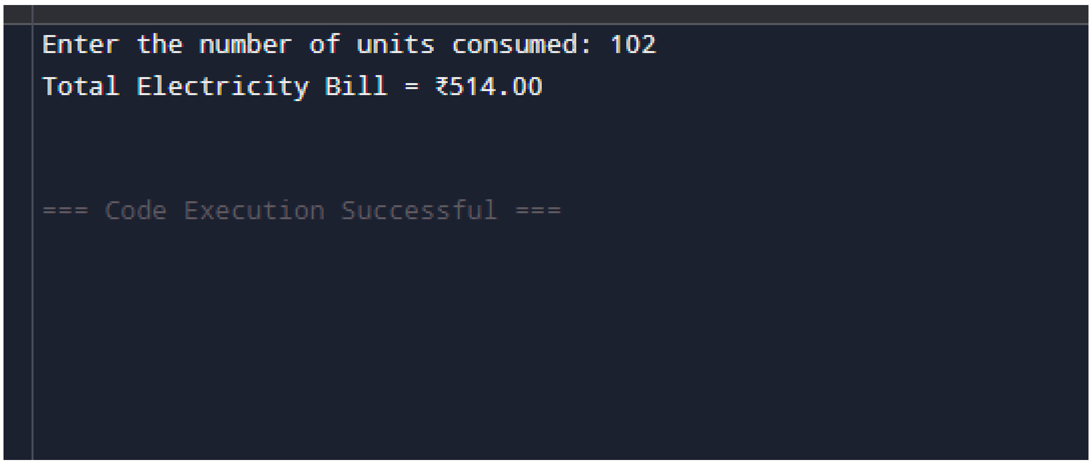
• Next 100 units: ₹7/unit

• Above 200 units: ₹10/unit



```c
// electricity calculation bill in C
#include <stdio.h>

int main() {
    int units;
    float bill = 0;

    printf("Enter the number of units consumed: ");
    scanf("%d", &units);

    if (units <= 100) {
        bill = units * 5;
    }
    else if (units <= 200) {
        bill = (100 * 5) + (units - 100) * 7;
    }
    else {
        bill = (100 * 5) + (100 * 7) + (units - 200) * 10;
    }

    printf("Total Electricity Bill = ₹%.2f\n", bill);

    return 0;
}
```

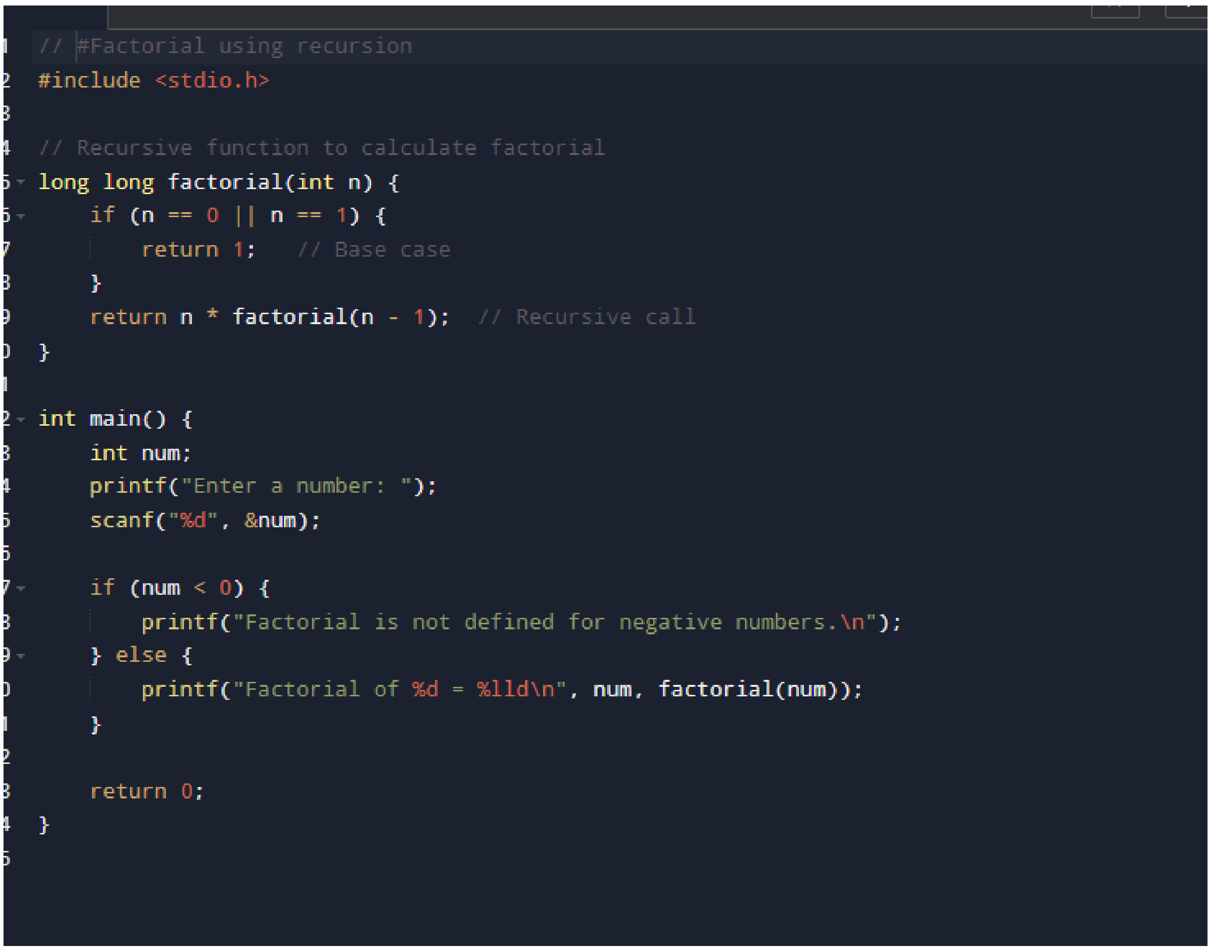
Output:



```
Enter the number of units consumed: 102
Total Electricity Bill = ₹514.00


=== Code Execution Successful ===
```

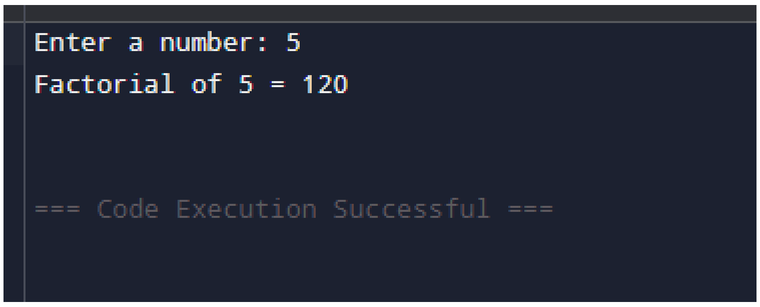
Q16. Factorial using Recursion

Problem: Write a recursive function to calculate the factorial of a given number



```c
// #Factorial using recursion
#include <stdio.h>

// Recursive function to calculate factorial
long long factorial(int n) {
    if (n == 0 || n == 1) {
        return 1;    // Base case
    }
    return n * factorial(n - 1);  // Recursive call
}

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    } else {
        printf("Factorial of %d = %lld\n", num, factorial(num));
    }

    return 0;
}
```

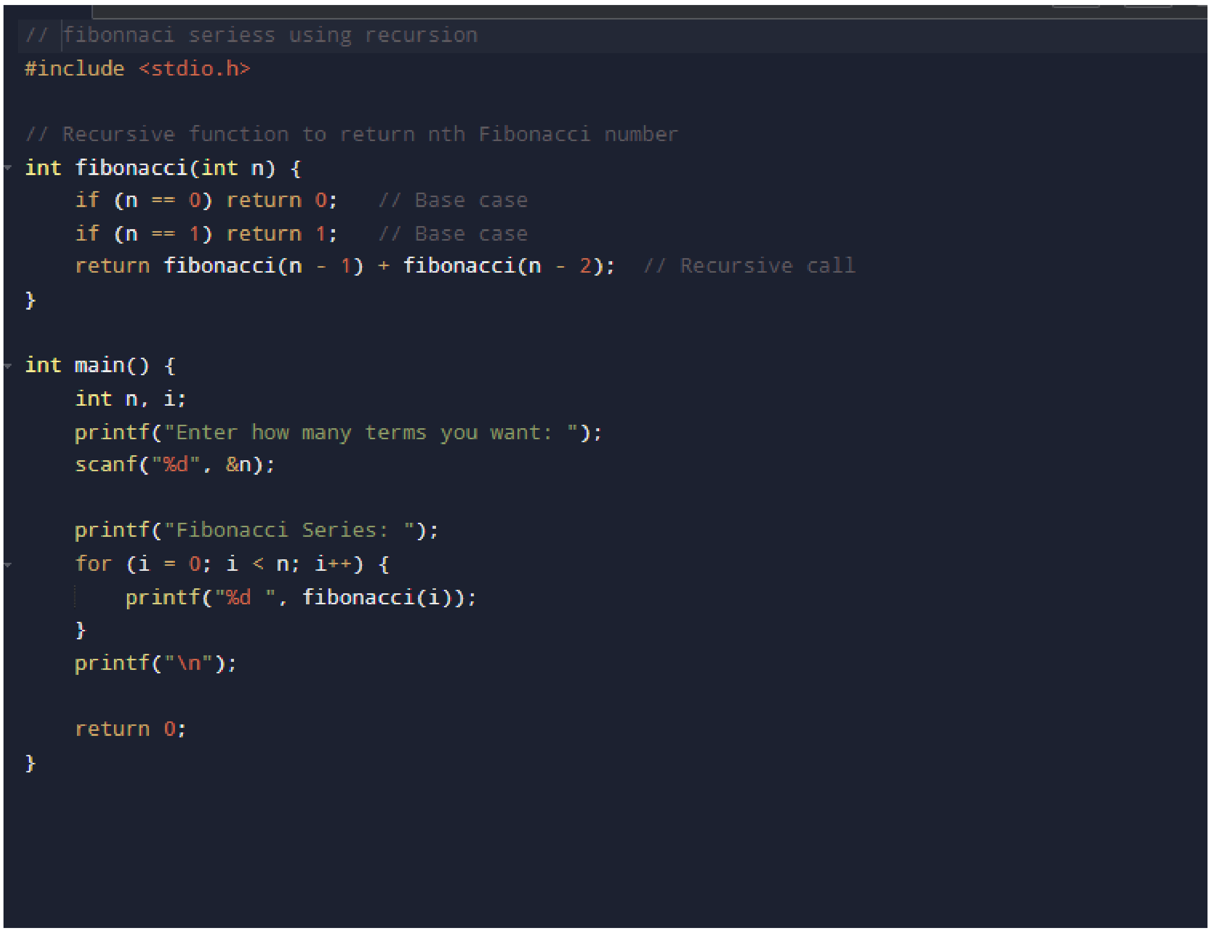
Output :



```
Enter a number: 5
Factorial of 5 = 120


=== Code Execution Successful ===
```

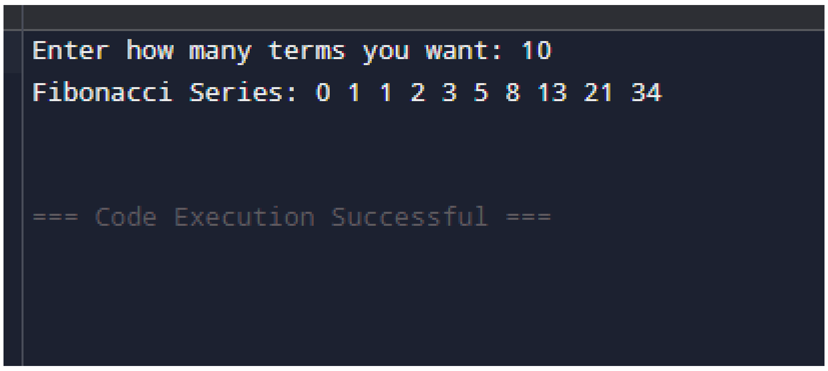
Q17. Fibonacci Series using Recursion

Problem: Print the first n Fibonacci numbers using recursion.



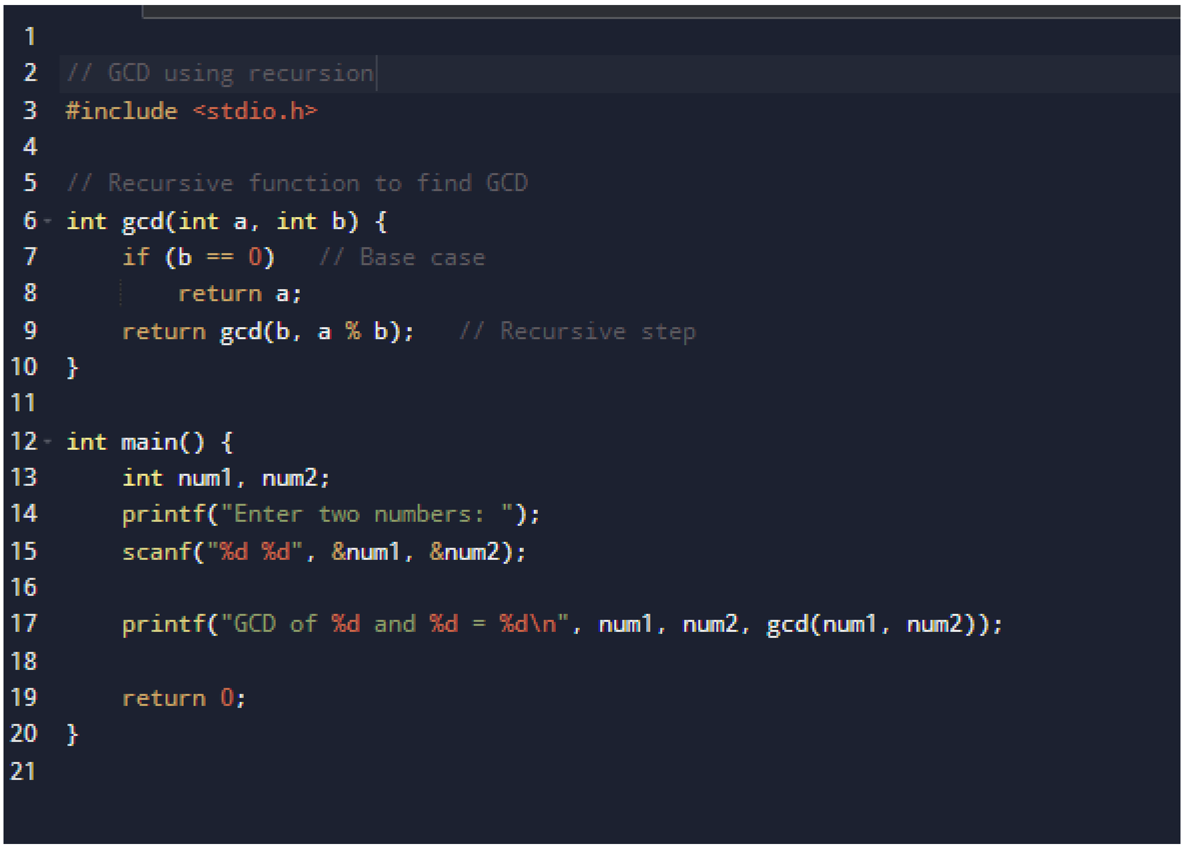
```c
// fibonnaci seriess using recursion
#include <stdio.h>

// Recursive function to return nth Fibonacci number
int fibonacci(int n) {
    if (n == 0) return 0;    // Base case
    if (n == 1) return 1;    // Base case
    return fibonacci(n - 1) + fibonacci(n - 2);  // Recursive call
}

int main() {
    int n, i;
    printf("Enter how many terms you want: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");
    for (i = 0; i < n; i++) {
        printf("%d ", fibonacci(i));
    }
    printf("\n");

    return 0;
}
```

Output:

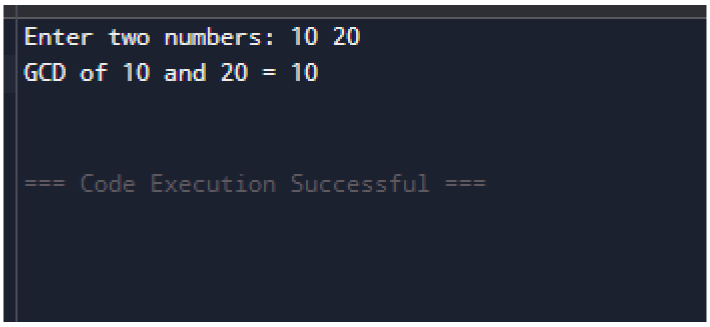

```
Enter how many terms you want: 10
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34


=== Code Execution Successful ===
```

Q18. GCD (Greatest Common Divisor) using Recursion

Scenario: A system needs to simplify fractions, so GCD of two numbers is required.

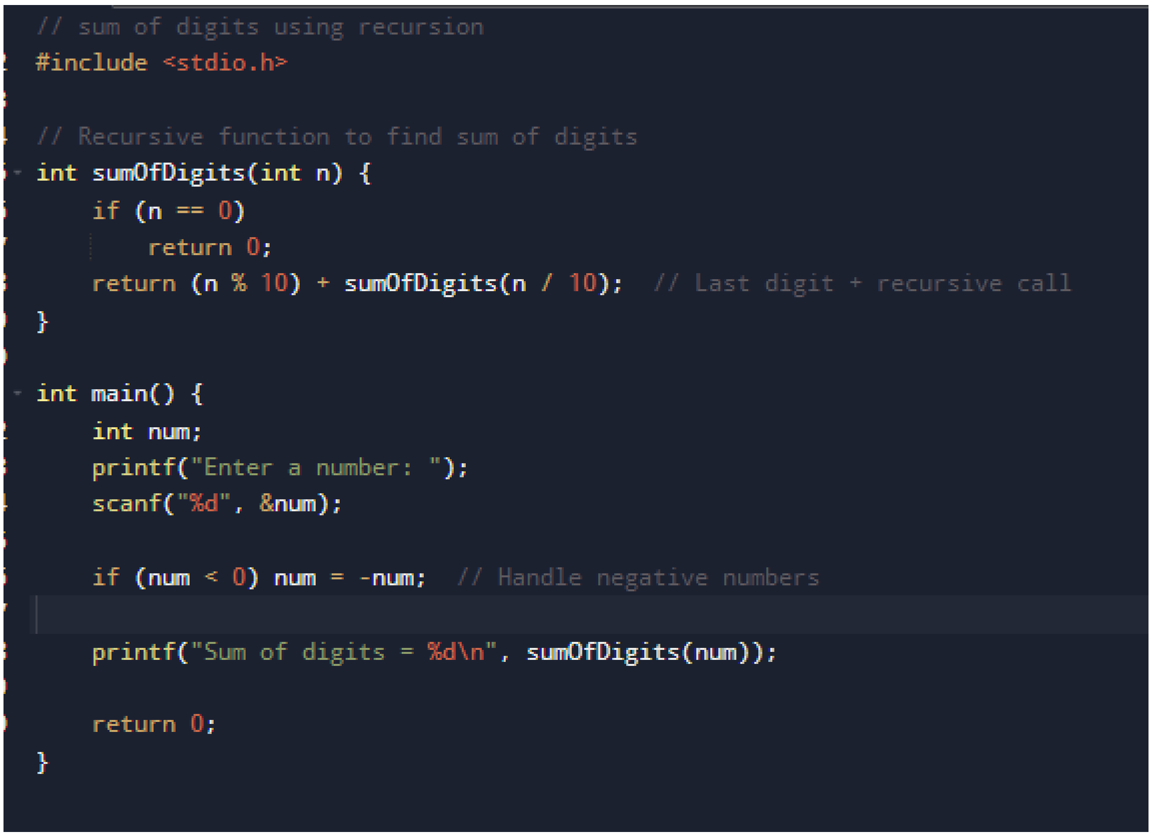

```c

// GCD using recursion
#include <stdio.h>

// Recursive function to find GCD
int gcd(int a, int b) {
    if (b == 0)   // Base case
        return a;
    return gcd(b, a % b);   // Recursive step
}

int main() {
    int num1, num2;
    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);

    printf("GCD of %d and %d = %d\n", num1, num2, gcd(num1, num2));

    return 0;
}

```

Output :


```
Enter two numbers: 10 20
GCD of 10 and 20 = 10


=== Code Execution Successful ===
```

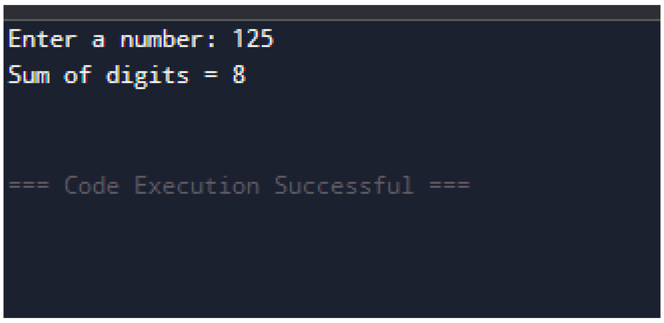
Q19. Sum of Digits using Recursion

Problem: Find the sum of digits of a number using recursion.



```c
// sum of digits using recursion
#include <stdio.h>

// Recursive function to find sum of digits
int sumOfDigits(int n) {
    if (n == 0)
        return 0;
    return (n % 10) + sumOfDigits(n / 10);  // Last digit + recursive call
}

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (num < 0) num = -num;  // Handle negative numbers

    printf("Sum of digits = %d\n", sumOfDigits(num));

    return 0;
}
```

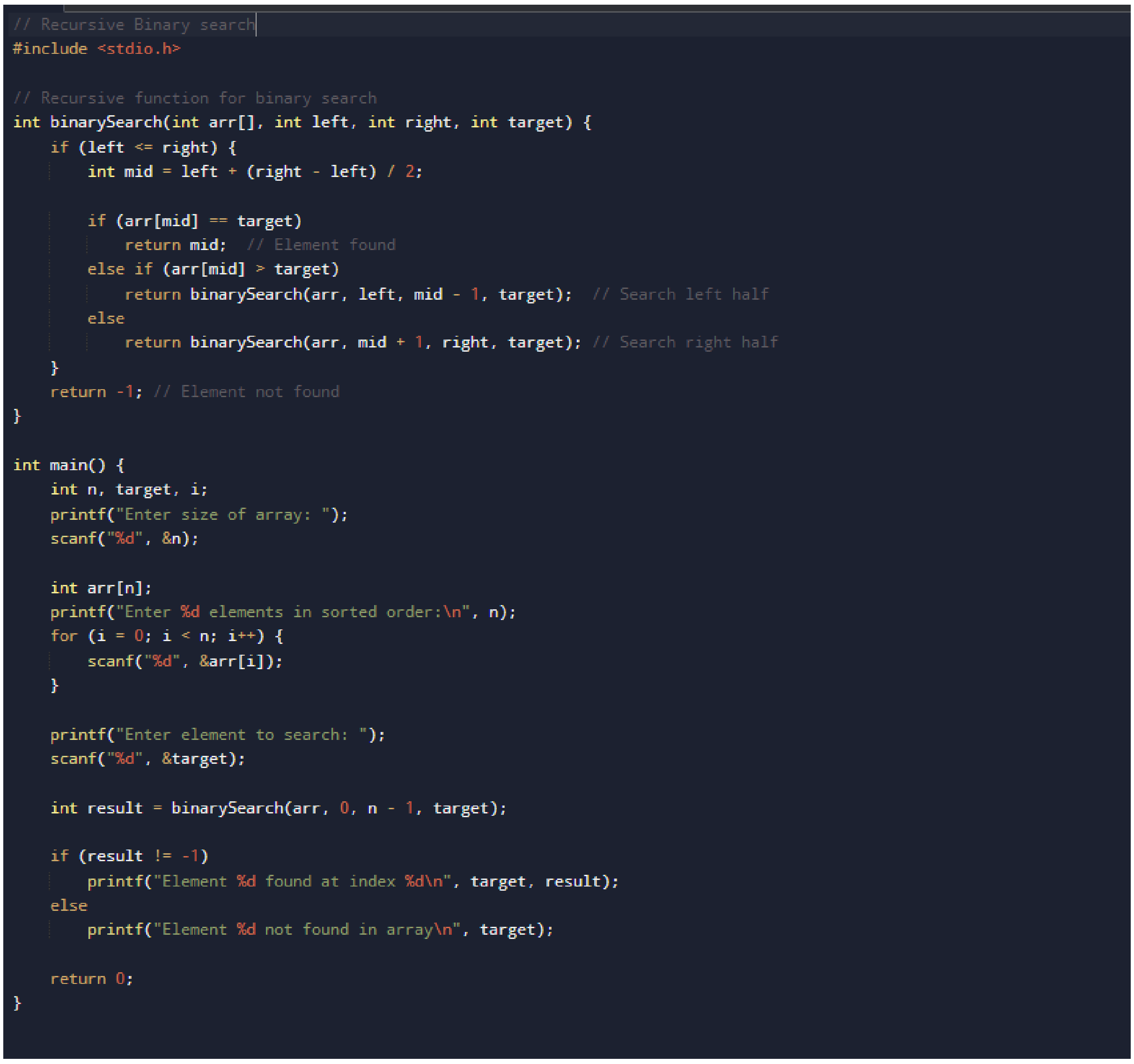
Output:



```
Enter a number: 125
Sum of digits = 8


=== Code Execution Successful ===
```

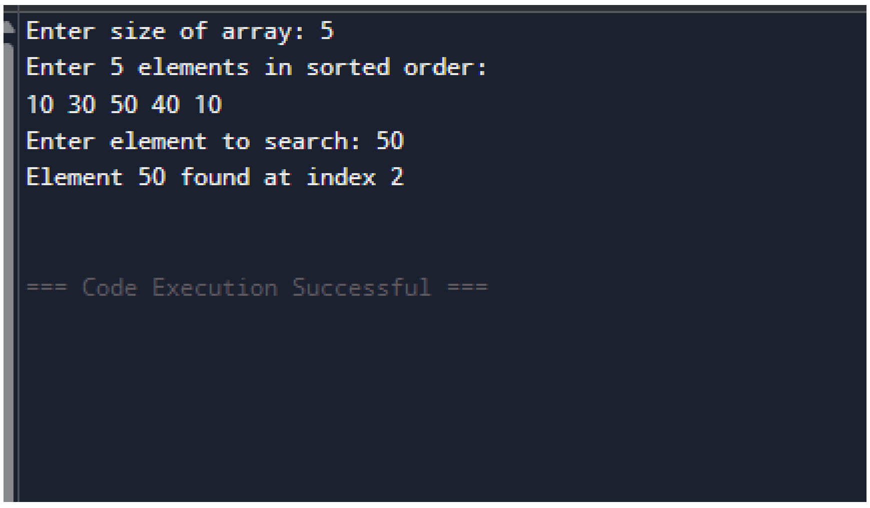
Q20. Recursive Binary Search

Scenario: A library system searches for a book ID in a sorted array of IDs. Implement binary search using recursion.



```c
// Recursive Binary search
#include <stdio.h>

// Recursive function for binary search
int binarySearch(int arr[], int left, int right, int target) {
    if (left <= right) {
        int mid = left + (right - left) / 2;

        if (arr[mid] == target)
            return mid;  // Element found
        else if (arr[mid] > target)
            return binarySearch(arr, left, mid - 1, target);  // Search left half
        else
            return binarySearch(arr, mid + 1, right, target); // Search right half
    }
    return -1; // Element not found
}

int main() {
    int n, target, i;
    printf("Enter size of array: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d elements in sorted order:\n", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Enter element to search: ");
    scanf("%d", &target);

    int result = binarySearch(arr, 0, n - 1, target);

    if (result != -1)
        printf("Element %d found at index %d\n", target, result);
    else
        printf("Element %d not found in array\n", target);

    return 0;
}
```

Output :



```
Enter size of array: 5
Enter 5 elements in sorted order:
10 30 50 40 10
Enter element to search: 50
Element 50 found at index 2


=== Code Execution Successful ===
```