Module-2

1. OVERVIEW OF C PROGRAMMING:

(1). Research and provide three real-world applications where C programming is extensively used, such as in embedded systems, operating systems, or game development.

Ans: Here are **three real-world applications** where **C programming** is extensively used:

1. Embedded Systems

- **Use Case:** C is widely used in programming microcontrollers and hardware devices.
- **Examples:**
  - Automotive systems (airbag controllers, ABS braking systems, infotainment).
  - Home appliances (washing machines, microwave ovens, smart TVs).
  - Medical devices (pacemakers, monitoring systems).
- **Reason for C:** It gives low-level hardware access, efficient performance, and runs with limited memory and processing power.

2. Operating Systems

- **Use Case:** Most modern operating systems have their kernel and core components written in C.
- **Examples:**
  - **Linux Kernel** (written mostly in C).
  - **Windows** (many system modules and drivers).
  - **macOS and iOS** (built on XNU kernel, largely C-based).
- **Reason for C:** Direct memory management, portability, and ability to write system-level code that interacts closely with hardware.

3. Game Development

- **Use Case:** Performance-critical game engines and graphics libraries rely on C.
- **Examples:**
  - **Doom (1993)** and **Quake** series (written in C).
  - **Unreal Engine** (early versions heavily used C/C++).
  - **Nintendo consoles** (firmware and many game engines built using C).
- **Reason for C:** Provides speed and control over system resources, crucial for real-time graphics and physics calculations in games.

2. SETTING UP ENVIORMENT:

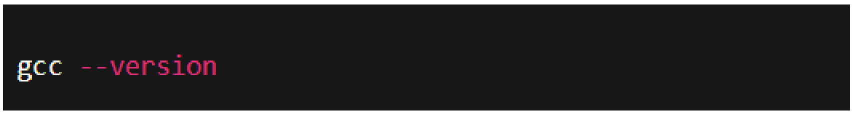
(2). Install a C compiler on your system and configure the IDE. Write your first program to print "Hello, World!" and run it.

Ans: **Step 1: Install a C Compiler**

Depending on your OS:

Windows:

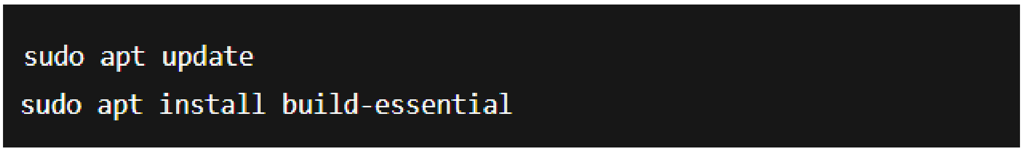
1. Download **MinGW** (Minimalist GNU for Windows): MinGW-w64
2. Install it and add the bin folder (e.g., C:\MinGW\bin) to your **PATH** environment variable.
3. To check installation, open **Command Prompt** and type:



```
gcc --version
```

Linux:

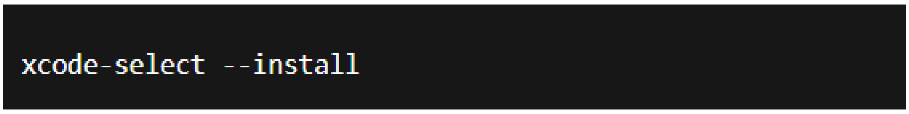
Most Linux distros already have GCC. If not, install with:



```
sudo apt update
sudo apt install build-essential
```

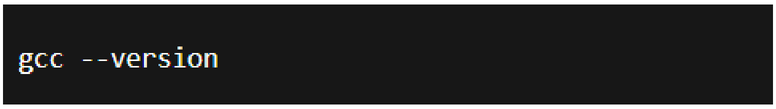
macOS

1. Install **Xcode Command Line Tools**:



```
xcode-select --install
```

2. Verify with:



```
gcc --version
```

Step 2: Install and Configure an IDE

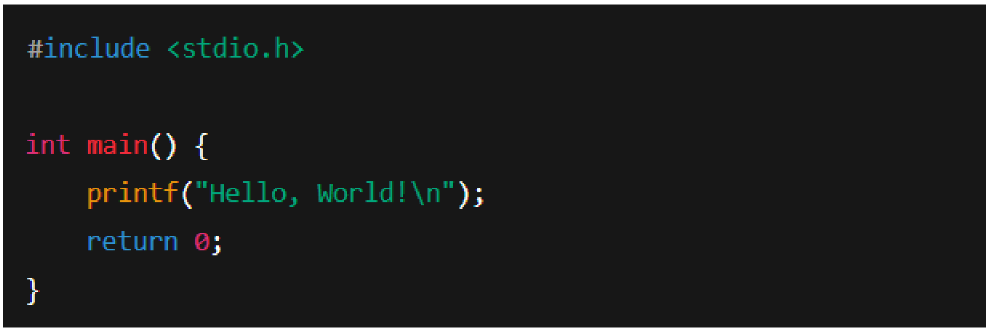
You can use:

- **Code::Blocks** (beginner-friendly, free).
- **Visual Studio Code (VS Code)** + C/C++ extension.
- **Dev-C++** (lightweight, simple).

In the IDE, make sure it points to the installed compiler (e.g., GCC).

Step 3: Write Your First Program

Open your IDE and create a new **C file** (hello.c), then type:

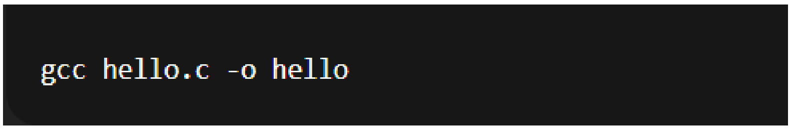


```
#include <stdio.h>

int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Step 4: Compile and Run

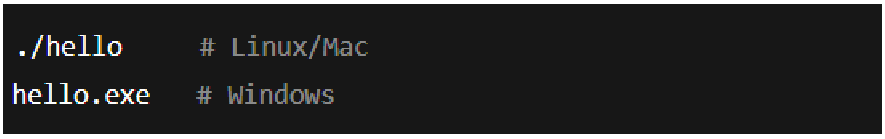
If using command line:

1. Save as hello.c.
2. Compile with:



```
gcc hello.c -o hello
```

Run it:

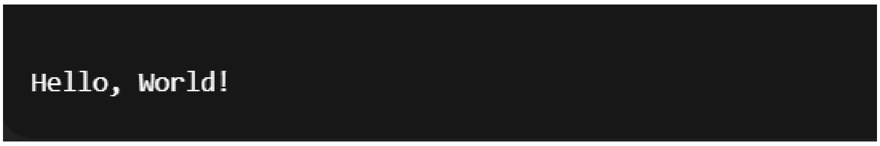


```
./hello      # Linux/Mac
hello.exe    # Windows
```

If using IDE:

- Just click **Build → Run** (or the play button).

You should see:

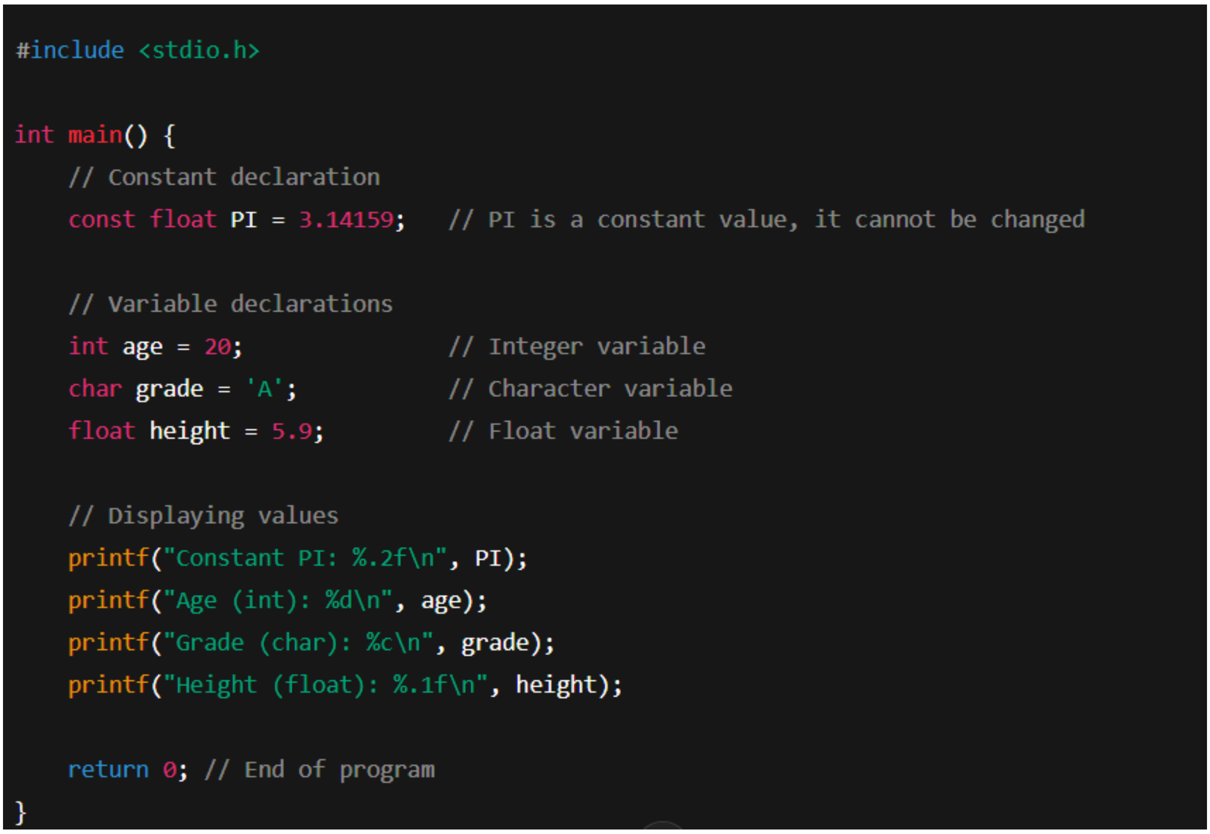


```
Hello, World!
```

3. BASIC STRUCTURE OF C:

(3). Write a C program that includes variables, constants, and comments. Declare and use different data types (int, char, float) and display their values.

Ans:



```
#include <stdio.h>

int main() {
    // Constant declaration
    const float PI = 3.14159;   // PI is a constant value, it cannot be changed

    // Variable declarations
    int age = 20;               // Integer variable
    char grade = 'A';           // Character variable
    float height = 5.9;         // Float variable

    // Displaying values
    printf("Constant PI: %.2f\n", PI);
    printf("Age (int): %d\n", age);
    printf("Grade (char): %c\n", grade);
    printf("Height (float): %.1f\n", height);

    return 0; // End of program
}
```

const float PI = 3.14159; → A constant value for π (cannot be changed later).

int age = 20; → An integer variable.

char grade = 'A'; → A character variable.

float height = 5.9; → A floating-point variable.

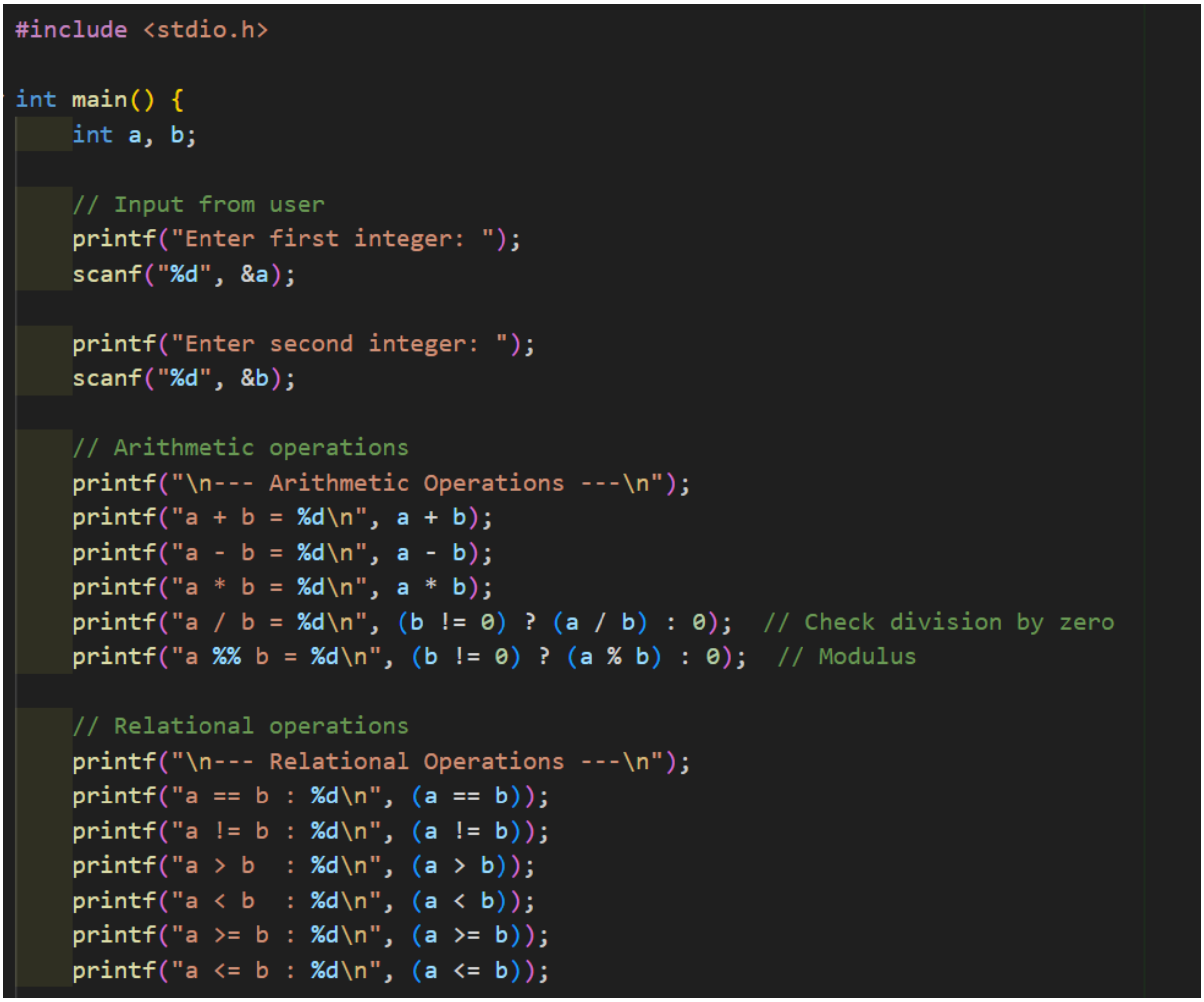
printf statements display the values.

- %.2f → prints float with 2 decimal places.
- %d → prints integer.
- %c → prints character.

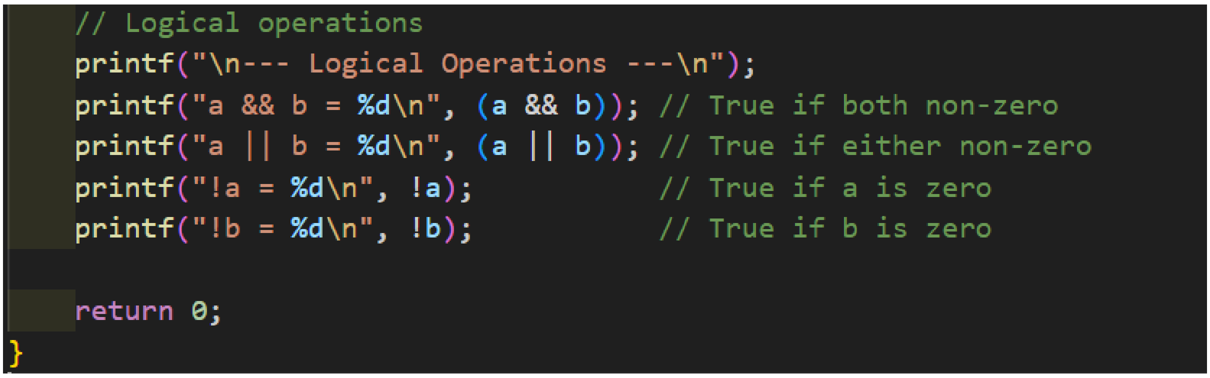
4. OPERATORS IN C:

(4). Write a C program that accepts two integers from the user and performs arithmetic, relational, and logical operations on them. Display the results.

Ans:



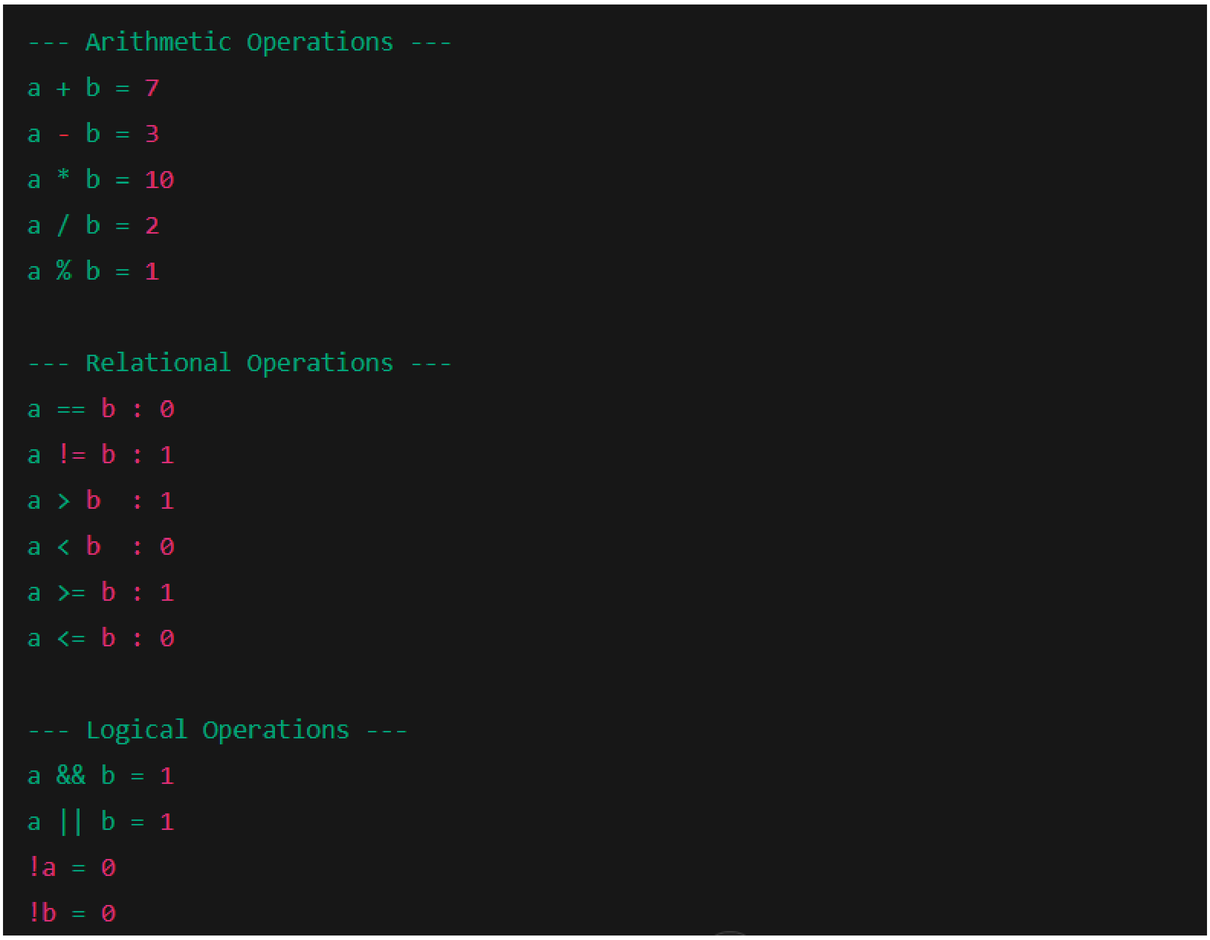
```c
#include <stdio.h>

int main() {
    int a, b;

    // Input from user
    printf("Enter first integer: ");
    scanf("%d", &a);

    printf("Enter second integer: ");
    scanf("%d", &b);

    // Arithmetic operations
    printf("\n--- Arithmetic Operations ---\n");
    printf("a + b = %d\n", a + b);
    printf("a - b = %d\n", a - b);
    printf("a * b = %d\n", a * b);
    printf("a / b = %d\n", (b != 0) ? (a / b) : 0);  // Check division by zero
    printf("a %% b = %d\n", (b != 0) ? (a % b) : 0);  // Modulus

    // Relational operations
    printf("\n--- Relational Operations ---\n");
    printf("a == b : %d\n", (a == b));
    printf("a != b : %d\n", (a != b));
    printf("a > b  : %d\n", (a > b));
    printf("a < b  : %d\n", (a < b));
    printf("a >= b : %d\n", (a >= b));
    printf("a <= b : %d\n", (a <= b));
```



```c
    // Logical operations
    printf("\n--- Logical Operations ---\n");
    printf("a && b = %d\n", (a && b)); // True if both non-zero
    printf("a || b = %d\n", (a || b)); // True if either non-zero
    printf("!a = %d\n", !a);           // True if a is zero
    printf("!b = %d\n", !b);           // True if b is zero

    return 0;
}
```

Example Output:

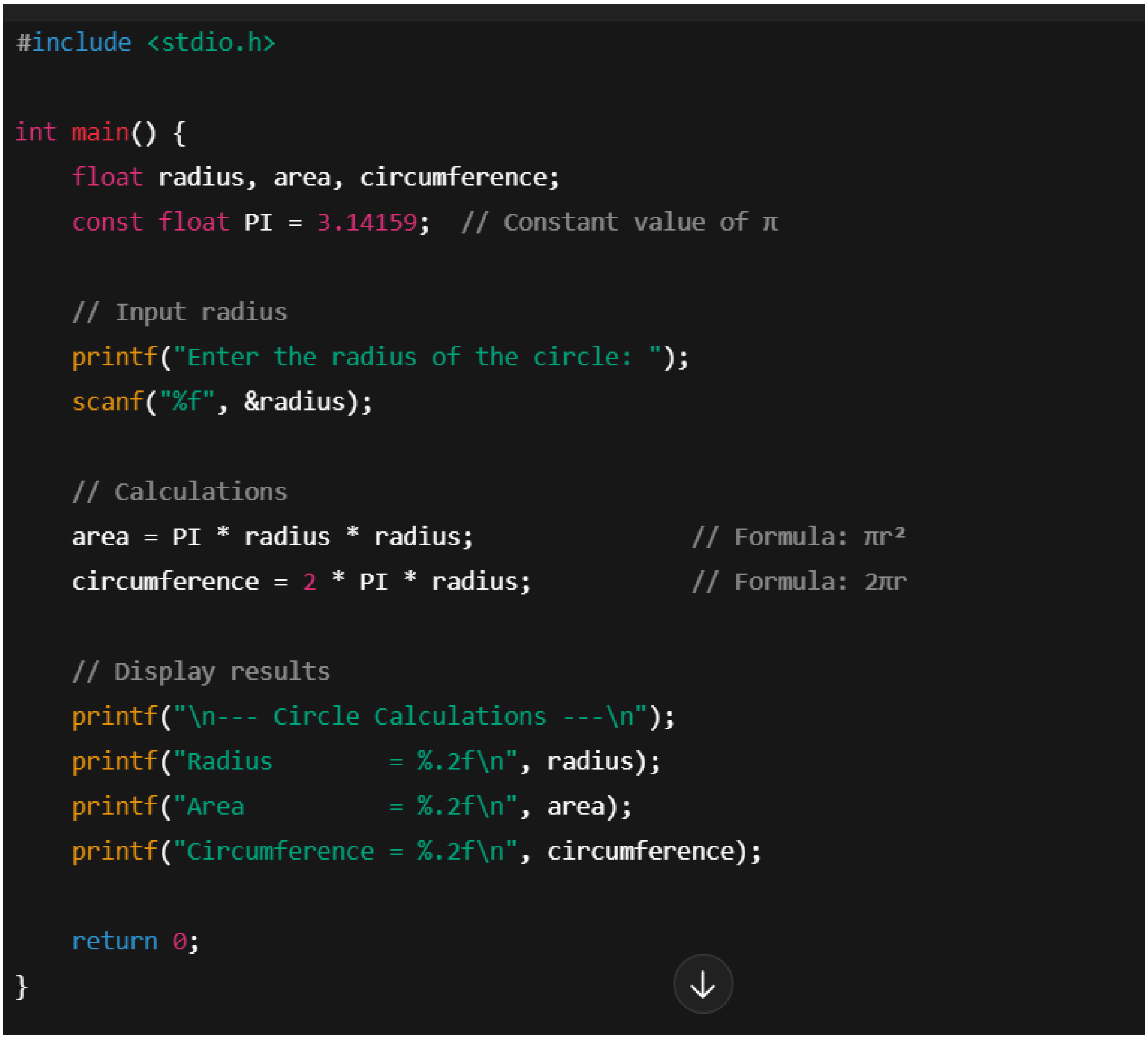
If user enters a = 5 and b = 2



```
--- Arithmetic Operations ---
a + b = 7
a - b = 3
a * b = 10
a / b = 2
a % b = 1

--- Relational Operations ---
a == b : 0
a != b : 1
a > b  : 1
a < b  : 0
a >= b : 1
a <= b : 0

--- Logical Operations ---
a && b = 1
a || b = 1
!a = 0
!b = 0
```

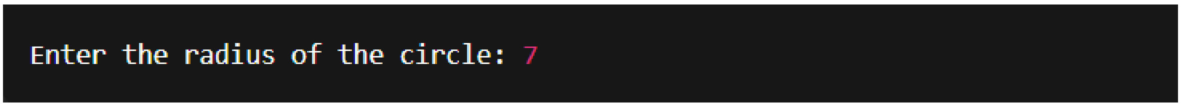
(5). WAP to Find Area And Circumference of Circle.

Ans:



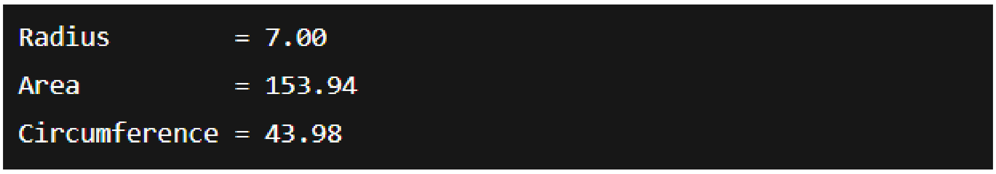
```c
#include <stdio.h>

int main() {
    float radius, area, circumference;
    const float PI = 3.14159;  // Constant value of π

    // Input radius
    printf("Enter the radius of the circle: ");
    scanf("%f", &radius);

    // Calculations
    area = PI * radius * radius;               // Formula: πr²
    circumference = 2 * PI * radius;           // Formula: 2πr

    // Display results
    printf("\n--- Circle Calculations ---\n");
    printf("Radius        = %.2f\n", radius);
    printf("Area          = %.2f\n", area);
    printf("Circumference = %.2f\n", circumference);

    return 0;
}
```

Input:



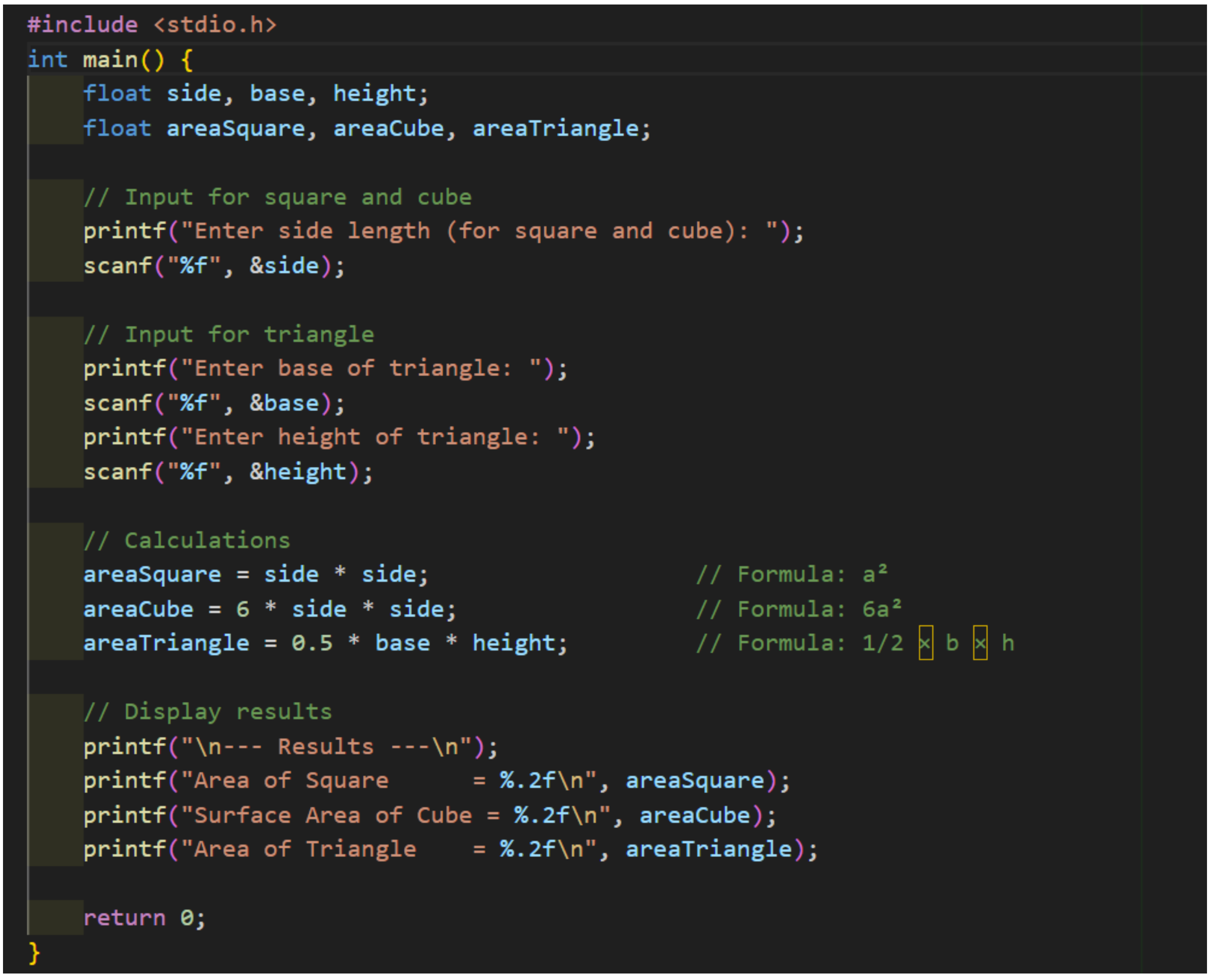
```
Enter the radius of the circle: 7
```

Output:



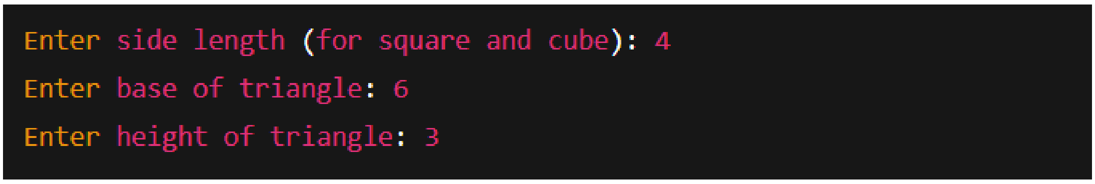
```
Radius        = 7.00
Area          = 153.94
Circumference = 43.98
```

(6). Find Area of Square formula : a = a2 5. Find Area of Cube formula : a = 6a2 6. Find area of Triangle Formula : A = 1/2 × b × h



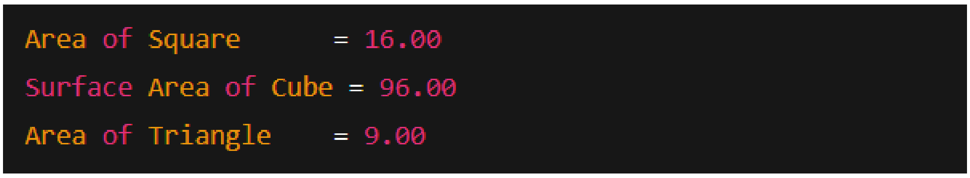
```c
#include <stdio.h>
int main() {
    float side, base, height;
    float areaSquare, areaCube, areaTriangle;

    // Input for square and cube
    printf("Enter side length (for square and cube): ");
    scanf("%f", &side);

    // Input for triangle
    printf("Enter base of triangle: ");
    scanf("%f", &base);
    printf("Enter height of triangle: ");
    scanf("%f", &height);

    // Calculations
    areaSquare = side * side;                   // Formula: a²
    areaCube = 6 * side * side;                 // Formula: 6a²
    areaTriangle = 0.5 * base * height;         // Formula: 1/2 × b × h

    // Display results
    printf("\n--- Results ---\n");
    printf("Area of Square      = %.2f\n", areaSquare);
    printf("Surface Area of Cube = %.2f\n", areaCube);
    printf("Area of Triangle    = %.2f\n", areaTriangle);

    return 0;
}
```

Input:



```
Enter side length (for square and cube): 4
Enter base of triangle: 6
Enter height of triangle: 3
```

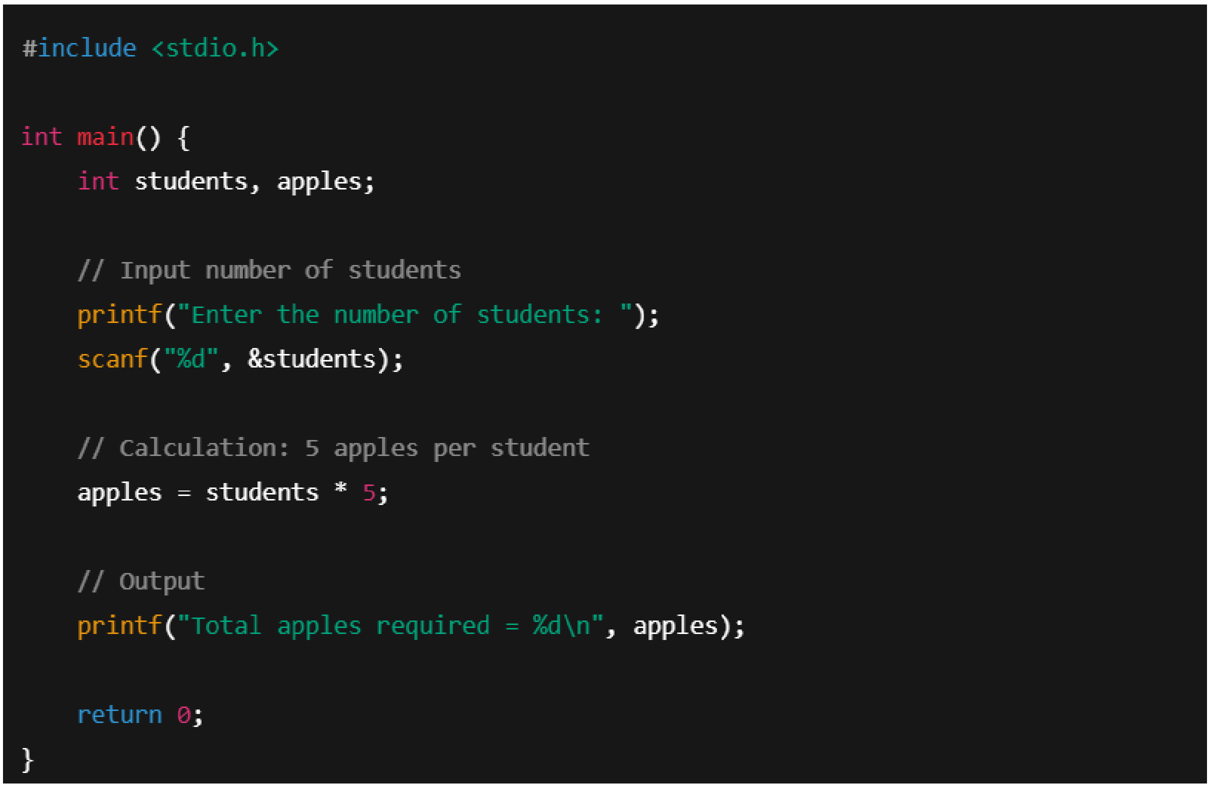
Output:



```
Area of Square       = 16.00
Surface Area of Cube = 96.00
Area of Triangle     = 9.00
```

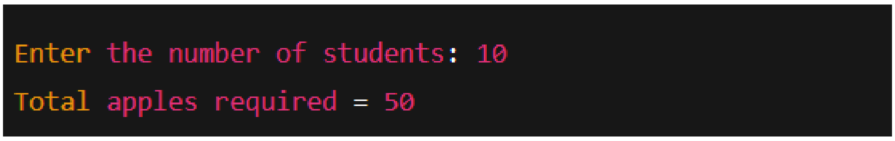
(7). Accept number of students from user. I need to give 5 apples to each student. How many apples are required?

Ans:



```
#include <stdio.h>

int main() {
    int students, apples;

    // Input number of students
    printf("Enter the number of students: ");
    scanf("%d", &students);

    // Calculation: 5 apples per student
    apples = students * 5;

    // Output
    printf("Total apples required = %d\n", apples);

    return 0;
}
```

Output:

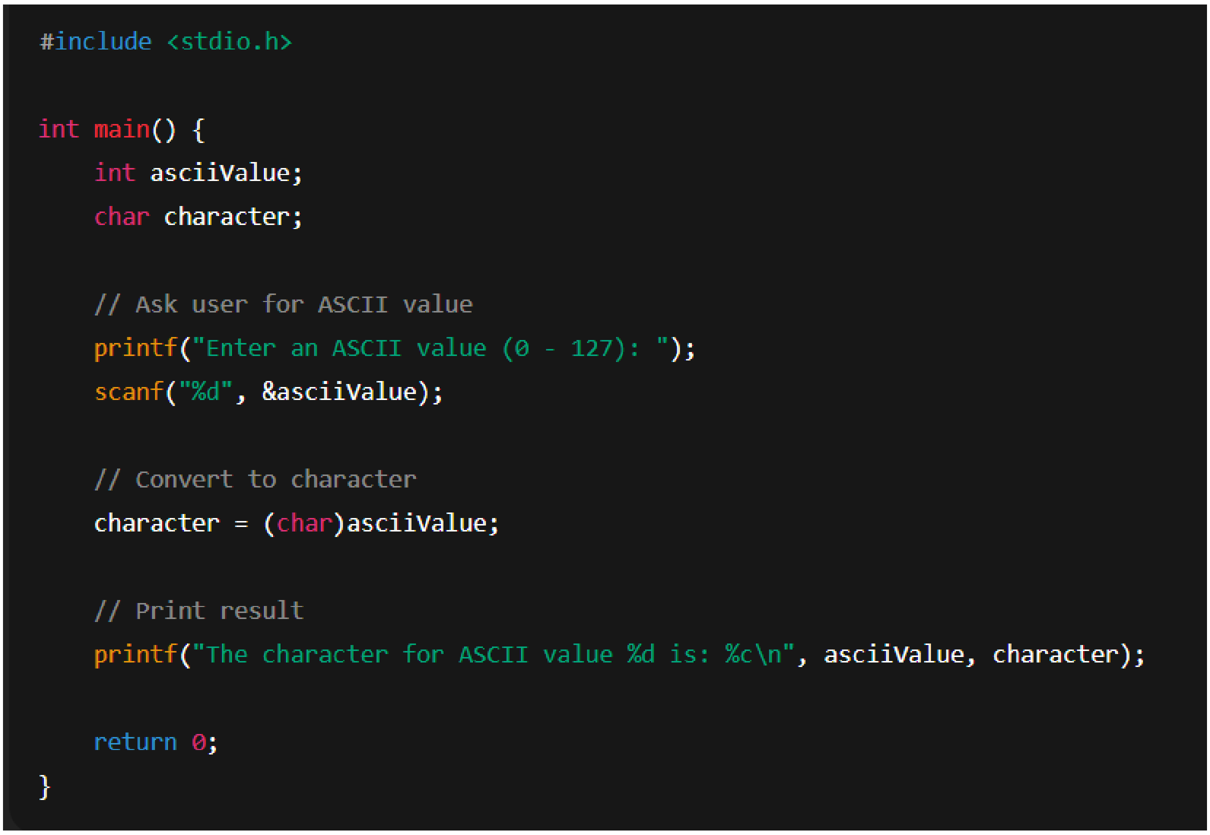


```
Enter the number of students: 10
Total apples required = 50
```

(8). Find character value from ascii.

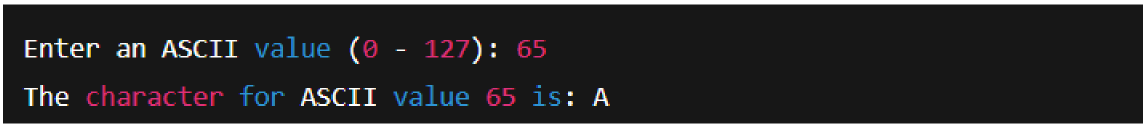
Ans: In **C language**, every character has an **ASCII value** (an integer code). If you want to **find the character from an ASCII value**, you can simply use a char variable and assign the ASCII number to it.

Here’s an example program:



```c
#include <stdio.h>

int main() {
    int asciiValue;
    char character;

    // Ask user for ASCII value
    printf("Enter an ASCII value (0 - 127): ");
    scanf("%d", &asciiValue);

    // Convert to character
    character = (char)asciiValue;

    // Print result
    printf("The character for ASCII value %d is: %c\n", asciiValue, character);

    return 0;
}
```

Example Output:



```
Enter an ASCII value (0 - 127): 65
The character for ASCII value 65 is: A
```

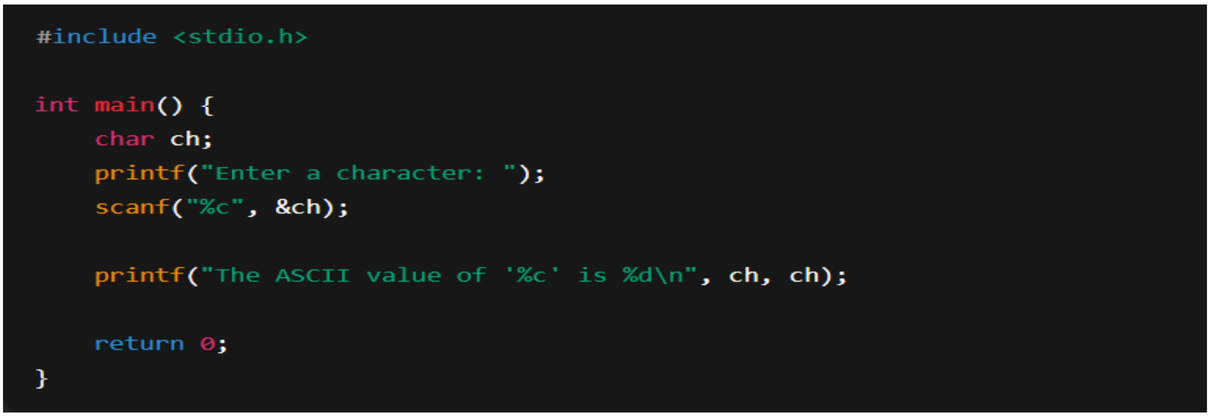
Explanation:

- ASCII value 65 corresponds to character 'A'.
- Type casting (char)asciiValue converts integer to character.

(9). Find ascii value of given number.

Ans: I think you mean **find the ASCII value of a given character in C language** (since ASCII values are defined for characters, not numbers directly).

Here's a simple C program:



```c
#include <stdio.h>

int main() {
    char ch;
    printf("Enter a character: ");
    scanf("%c", &ch);

    printf("The ASCII value of '%c' is %d\n", ch, ch);

    return 0;
}
```

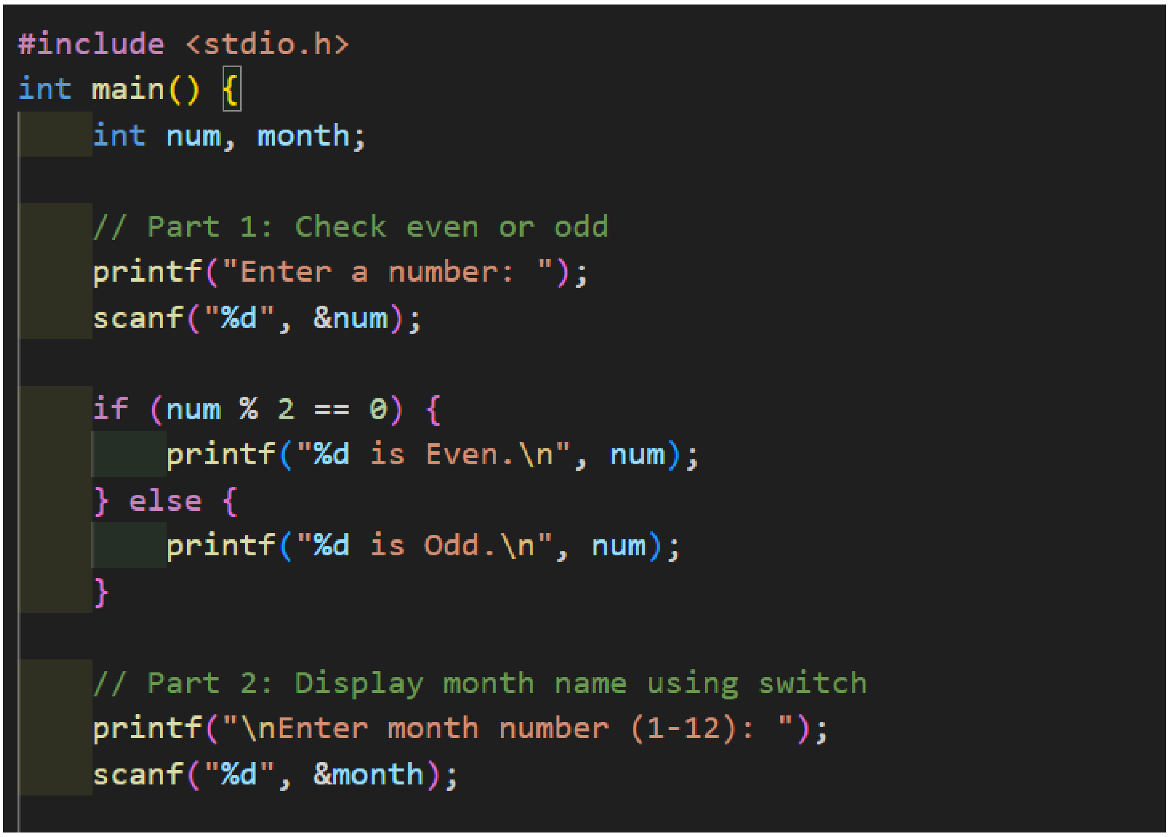
Example Run:

```
Enter a character: A
The ASCII value of 'A' is 65
```

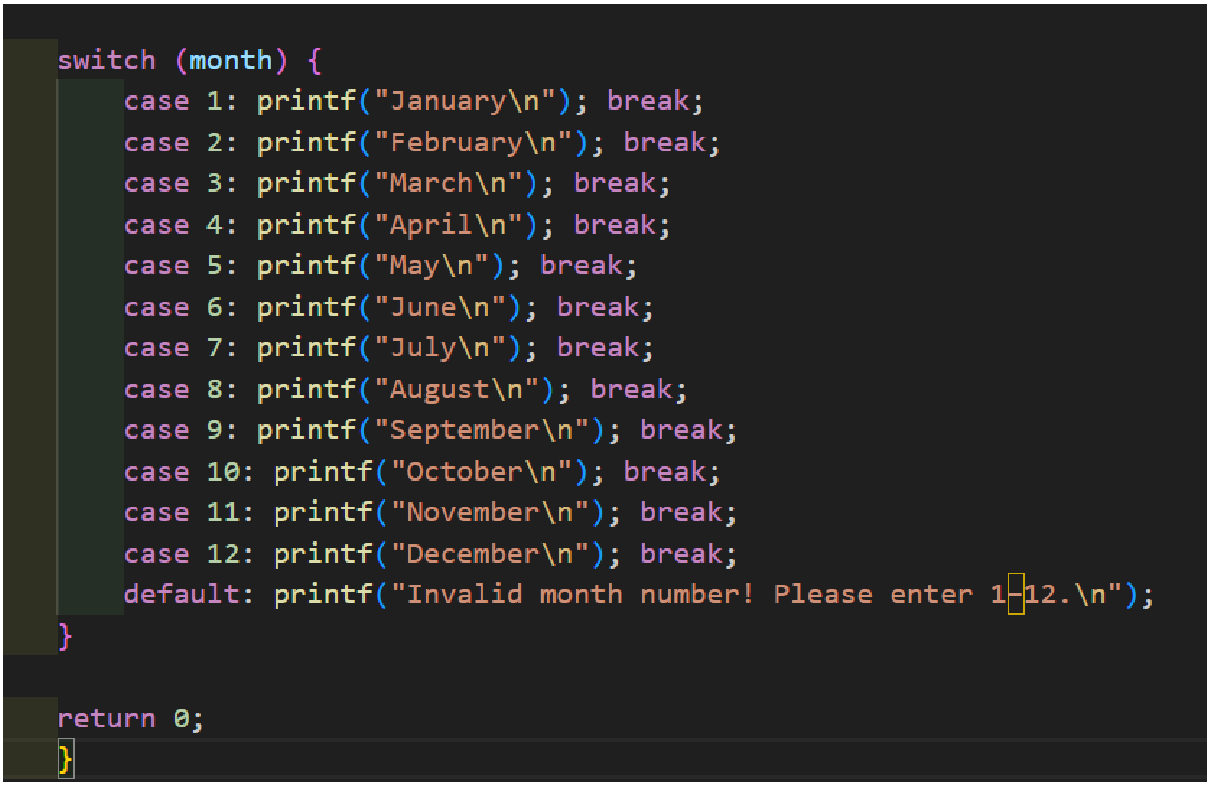
5. CONTROL FLOW STATEMENTS:

(10). Write a C program to check if a number is even or odd using an if-else statement. Extend the program using a switch statement to display the month name based on the user's input (1 for January, 2 for February, etc.).

Ans:

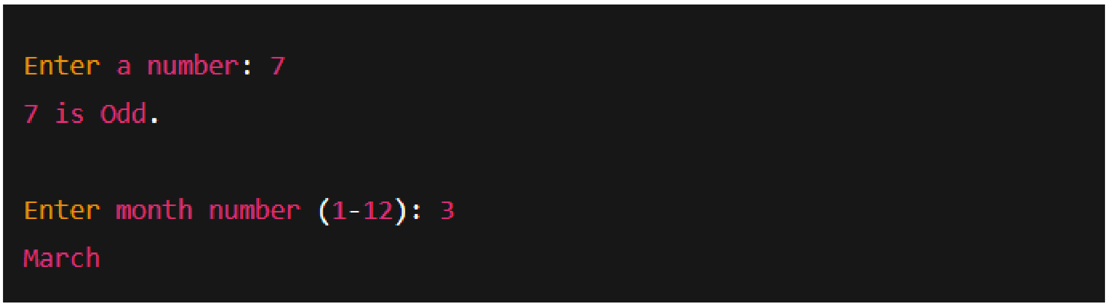


```c
#include <stdio.h>
int main() {
    int num, month;

    // Part 1: Check even or odd
    printf("Enter a number: ");
    scanf("%d", &num);

    if (num % 2 == 0) {
        printf("%d is Even.\n", num);
    } else {
        printf("%d is Odd.\n", num);
    }

    // Part 2: Display month name using switch
    printf("\nEnter month number (1-12): ");
    scanf("%d", &month);

```



```c
switch (month) {
    case 1: printf("January\n"); break;
    case 2: printf("February\n"); break;
    case 3: printf("March\n"); break;
    case 4: printf("April\n"); break;
    case 5: printf("May\n"); break;
    case 6: printf("June\n"); break;
    case 7: printf("July\n"); break;
    case 8: printf("August\n"); break;
    case 9: printf("September\n"); break;
    case 10: printf("October\n"); break;
    case 11: printf("November\n"); break;
    case 12: printf("December\n"); break;
    default: printf("Invalid month number! Please enter 1-12.\n");
}

return 0;
}
```

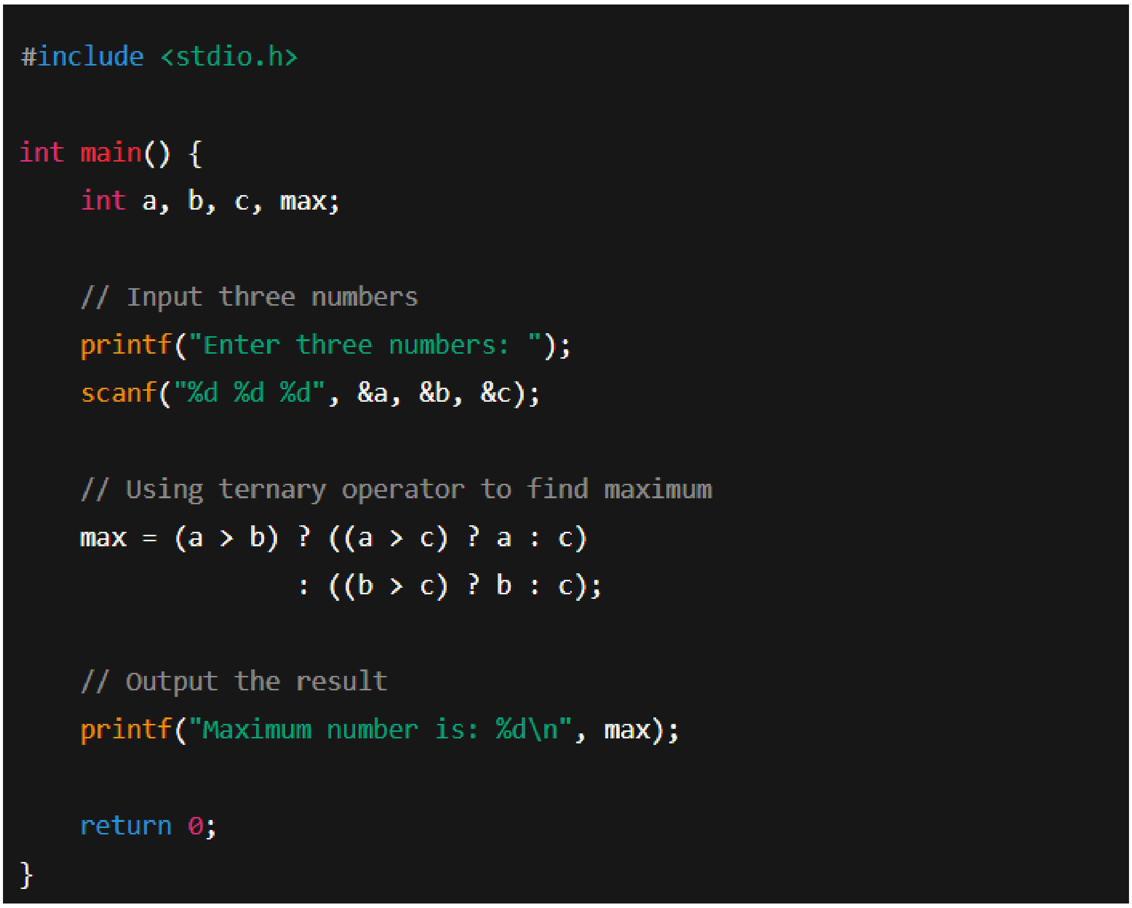
Example Run:



```
Enter a number: 7
7 is Odd.

Enter month number (1-12): 3
March
```

(11). WAP to find maximum number among 3 numbers using ternary operator.

Ans:



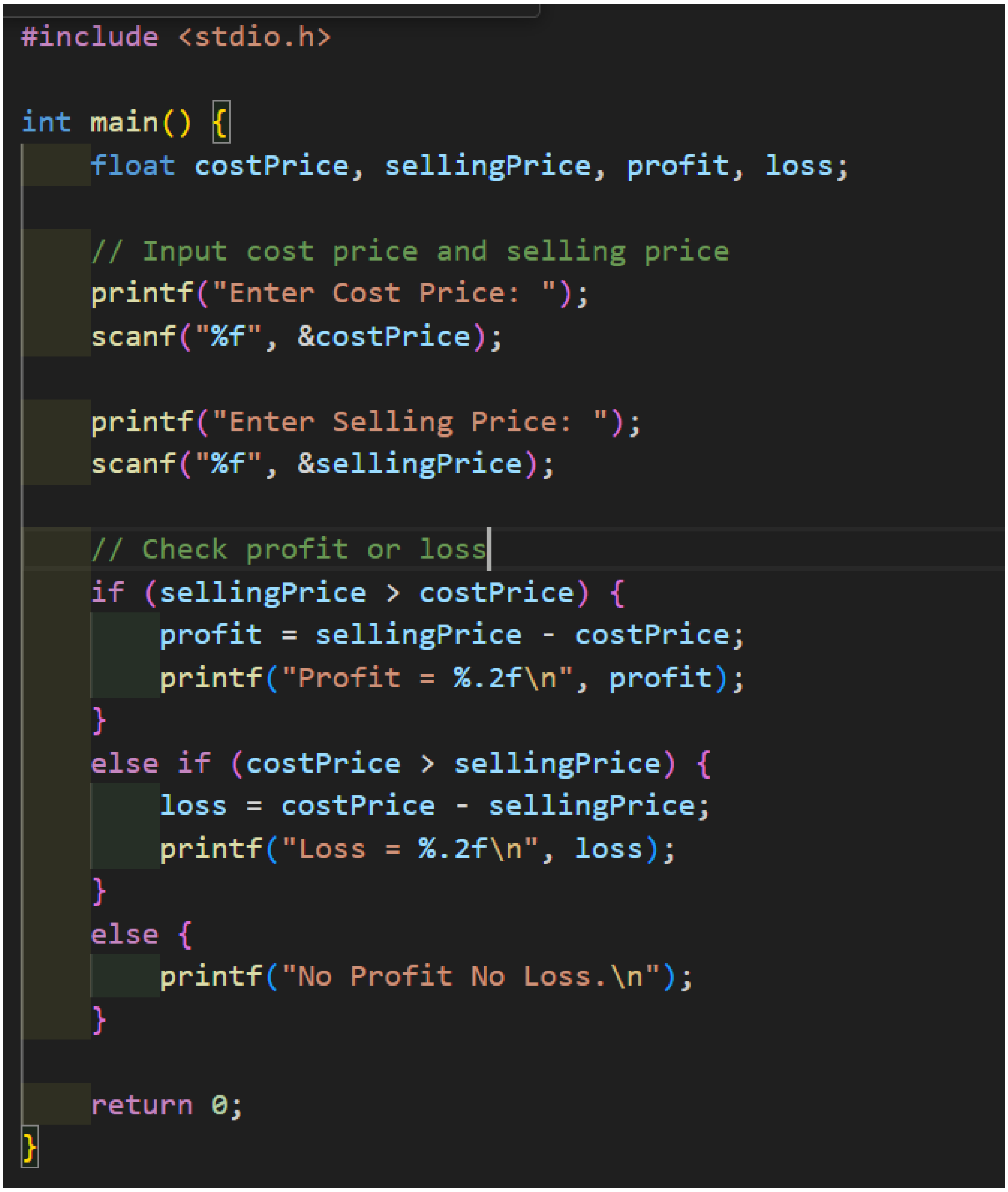
```
#include <stdio.h>

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

    // Input three numbers
    printf("Enter three numbers: ");
    scanf("%d %d %d", &a, &b, &c);

    // Using ternary operator to find maximum
    max = (a > b) ? ((a > c) ? a : c)
                  : ((b > c) ? b : c);

    // Output the result
    printf("Maximum number is: %d\n", max);

    return 0;
}
```

Explanation:

1. (a > b) : → checks if a is greater than b.
2. If true → compare a with c.
3. If false → compare b with c.
4. Final result is stored in max.

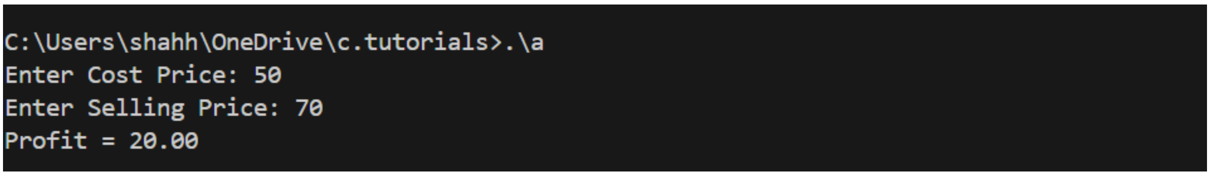
(12). Write a C program to calculate profit and loss on a transaction.

Ans:



```c
#include <stdio.h>

int main() {
    float costPrice, sellingPrice, profit, loss;

    // Input cost price and selling price
    printf("Enter Cost Price: ");
    scanf("%f", &costPrice);

    printf("Enter Selling Price: ");
    scanf("%f", &sellingPrice);

    // Check profit or loss
    if (sellingPrice > costPrice) {
        profit = sellingPrice - costPrice;
        printf("Profit = %.2f\n", profit);
    }
    else if (costPrice > sellingPrice) {
        loss = costPrice - sellingPrice;
        printf("Loss = %.2f\n", loss);
    }
    else {
        printf("No Profit No Loss.\n");
    }

    return 0;
}
```

Output In Terminal:

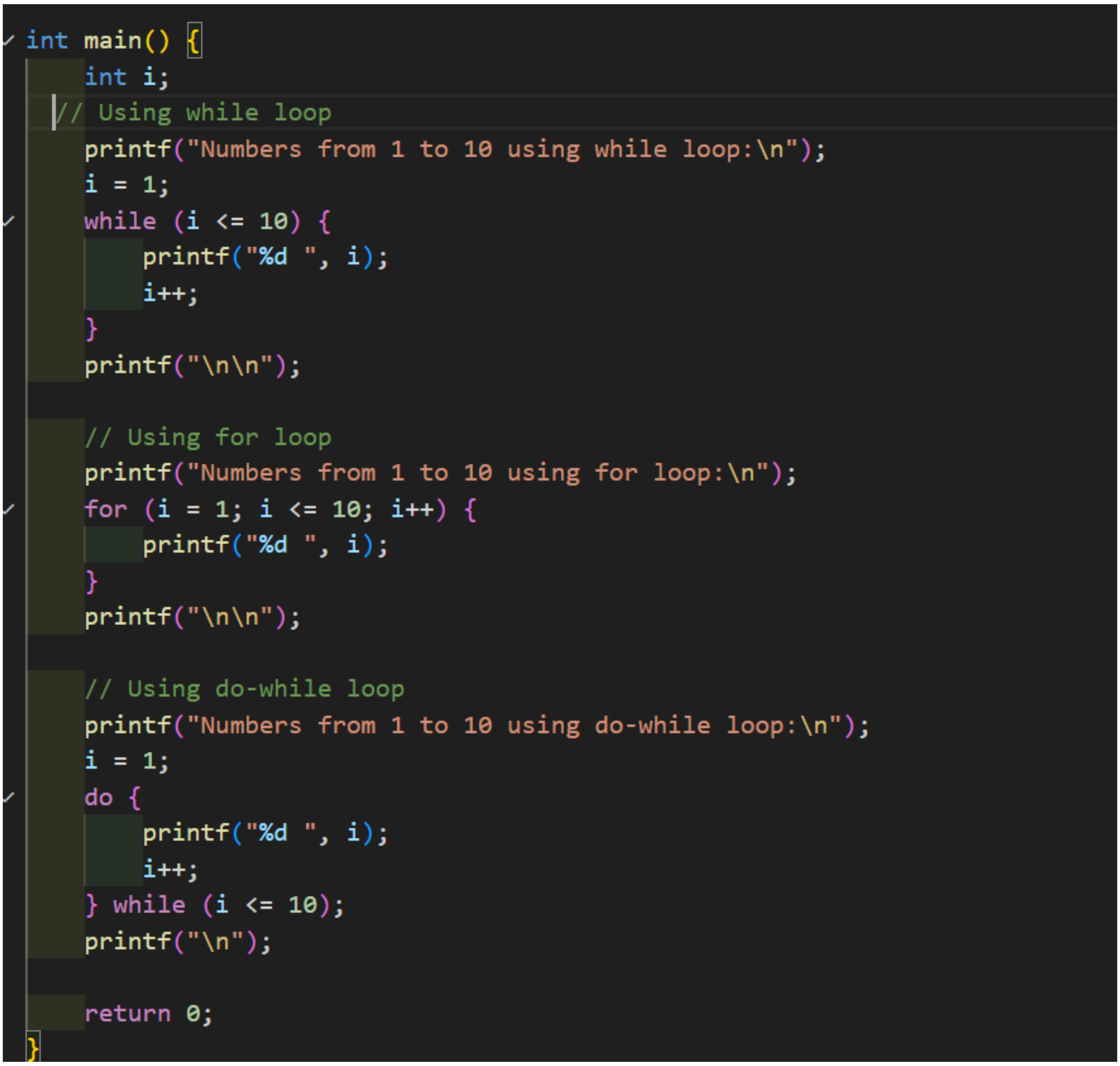


```
C:\Users\shahh\OneDrive\c.tutorials>.\a
Enter Cost Price: 50
Enter Selling Price: 70
Profit = 20.00
```

6. Looping in C:

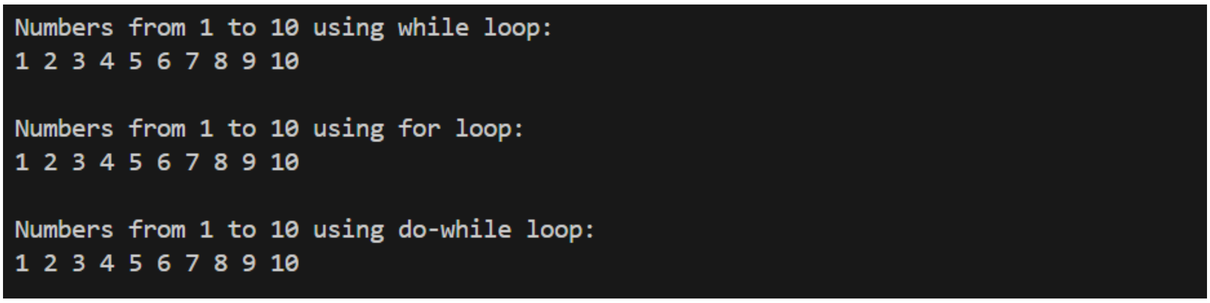
(13). Write a C program to print numbers from 1 to 10 using all three types of loops (while, for, do-while).

Ans:



```c
int main() {
    int i;
    // Using while loop
    printf("Numbers from 1 to 10 using while loop:\n");
    i = 1;
    while (i <= 10) {
        printf("%d ", i);
        i++;
    }
    printf("\n\n");

    // Using for loop
    printf("Numbers from 1 to 10 using for loop:\n");
    for (i = 1; i <= 10; i++) {
        printf("%d ", i);
    }
    printf("\n\n");

    // Using do-while loop
    printf("Numbers from 1 to 10 using do-while loop:\n");
    i = 1;
    do {
        printf("%d ", i);
        i++;
    } while (i <= 10);
    printf("\n");

    return 0;
}
```

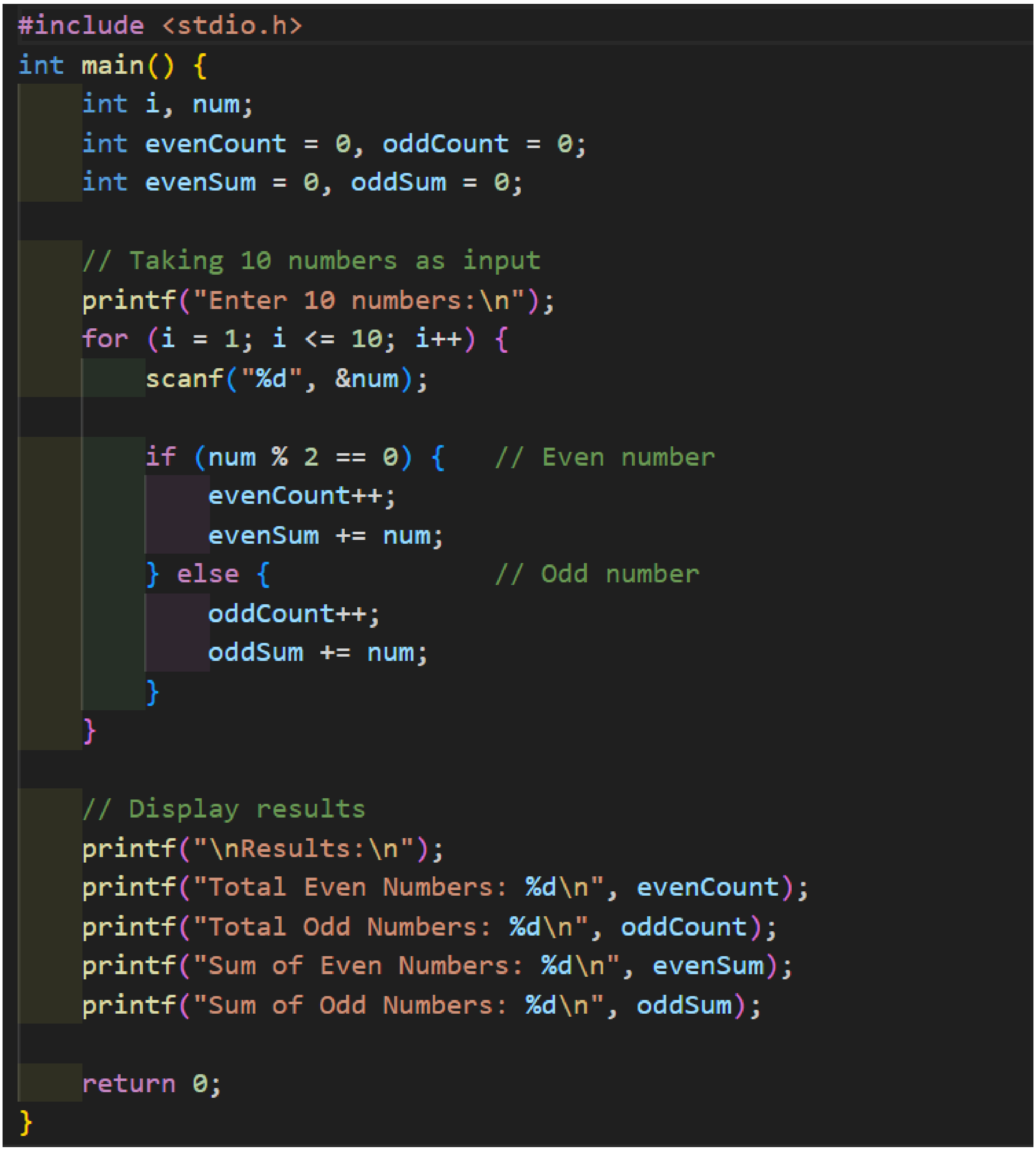
Output:



```
Numbers from 1 to 10 using while loop:
1 2 3 4 5 6 7 8 9 10

Numbers from 1 to 10 using for loop:
1 2 3 4 5 6 7 8 9 10

Numbers from 1 to 10 using do-while loop:
1 2 3 4 5 6 7 8 9 10
```

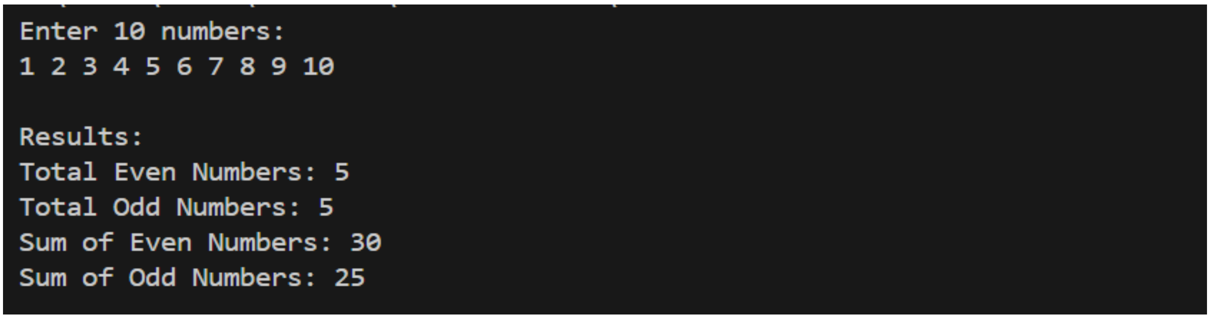
(14). WAP to take 10 no. Input from user find out below values a. How many Even numbers are there b. How many odd numbers are there c. Sum of even numbers d. Sum of odd numbers?

Ans:



```c
#include <stdio.h>
int main() {
    int i, num;
    int evenCount = 0, oddCount = 0;
    int evenSum = 0, oddSum = 0;

    // Taking 10 numbers as input
    printf("Enter 10 numbers:\n");
    for (i = 1; i <= 10; i++) {
        scanf("%d", &num);

        if (num % 2 == 0) {    // Even number
            evenCount++;
            evenSum += num;
        } else {               // Odd number
            oddCount++;
            oddSum += num;
        }
    }

    // Display results
    printf("\nResults:\n");
    printf("Total Even Numbers: %d\n", evenCount);
    printf("Total Odd Numbers: %d\n", oddCount);
    printf("Sum of Even Numbers: %d\n", evenSum);
    printf("Sum of Odd Numbers: %d\n", oddSum);

    return 0;
}
```

Output:

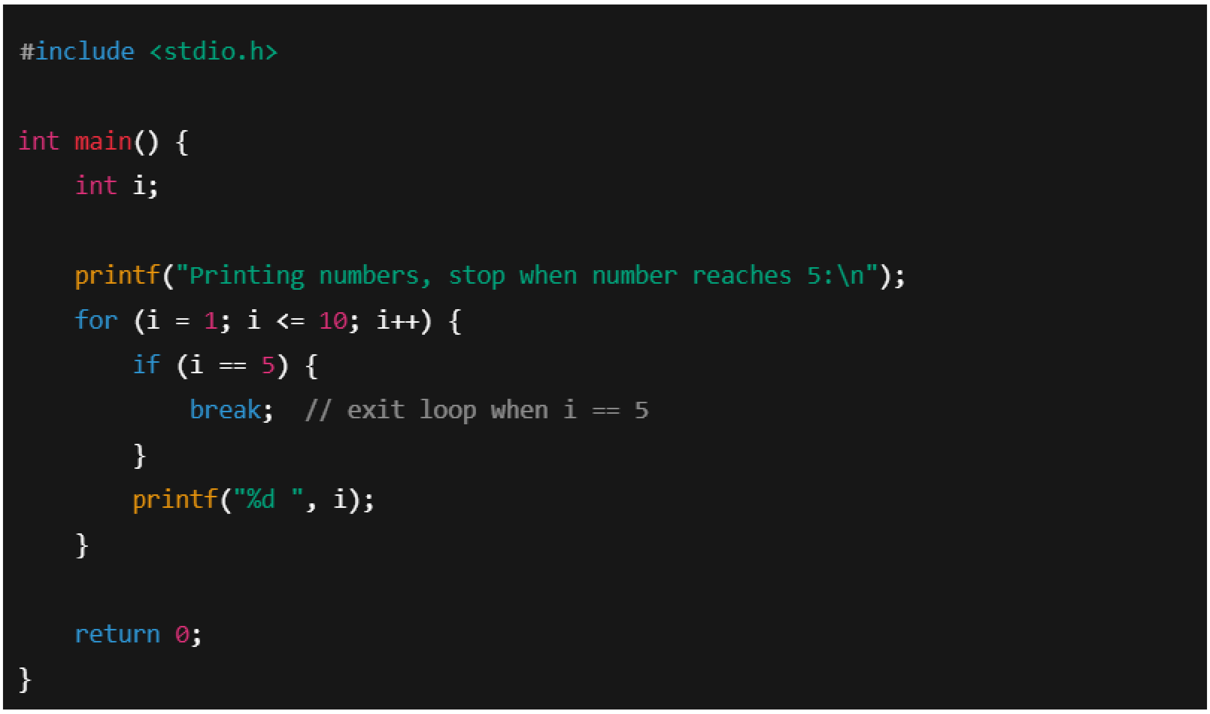


```
Enter 10 numbers:
1 2 3 4 5 6 7 8 9 10

Results:
Total Even Numbers: 5
Total Odd Numbers: 5
Sum of Even Numbers: 30
Sum of Odd Numbers: 25
```

7. Loop Control Statements:

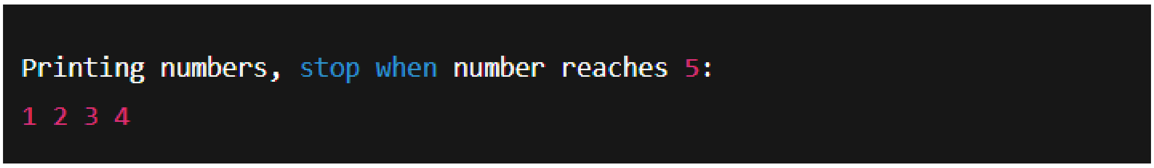
(15). Write a C program that uses the break statement to stop printing numbers when it reaches 5. Modify the program to skip printing the number 3 using the continue statement.

Ans: Part 1: Using break (stop when number reaches 5):



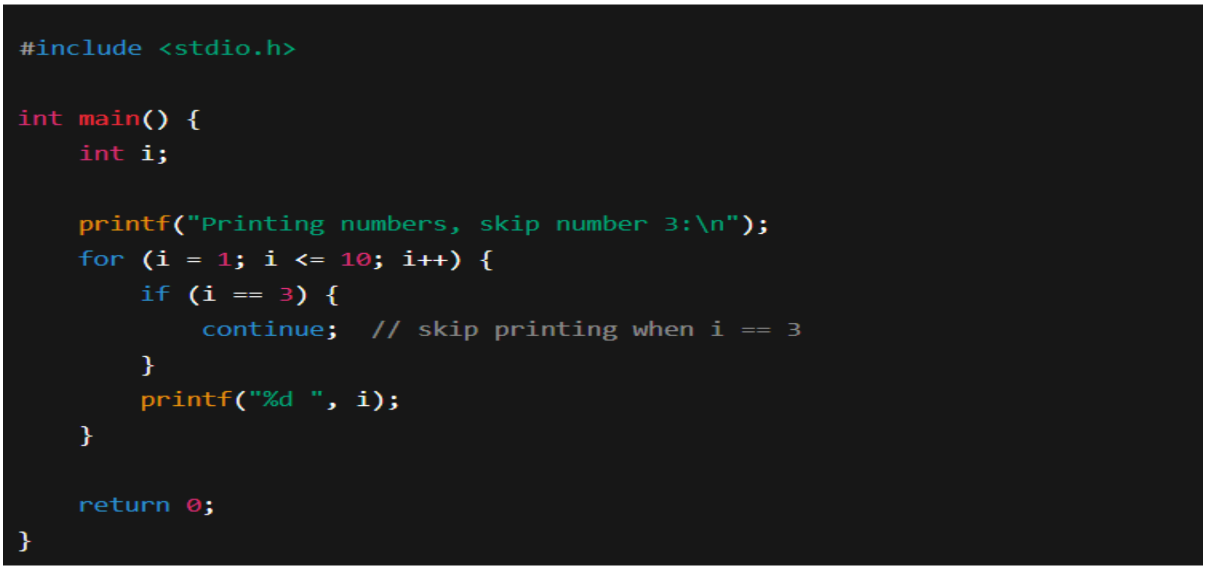
```c
#include <stdio.h>

int main() {
    int i;

    printf("Printing numbers, stop when number reaches 5:\n");
    for (i = 1; i <= 10; i++) {
        if (i == 5) {
            break;  // exit loop when i == 5
        }
        printf("%d ", i);
    }

    return 0;
}
```

Output:



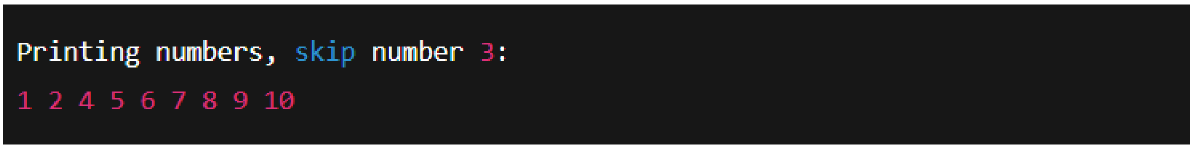
```
Printing numbers, stop when number reaches 5:
1 2 3 4
```

Part 2: Using continue (skip printing number 3):



```c
#include <stdio.h>

int main() {
    int i;

    printf("Printing numbers, skip number 3:\n");
    for (i = 1; i <= 10; i++) {
        if (i == 3) {
            continue;  // skip printing when i == 3
        }
        printf("%d ", i);
    }

    return 0;
}
```

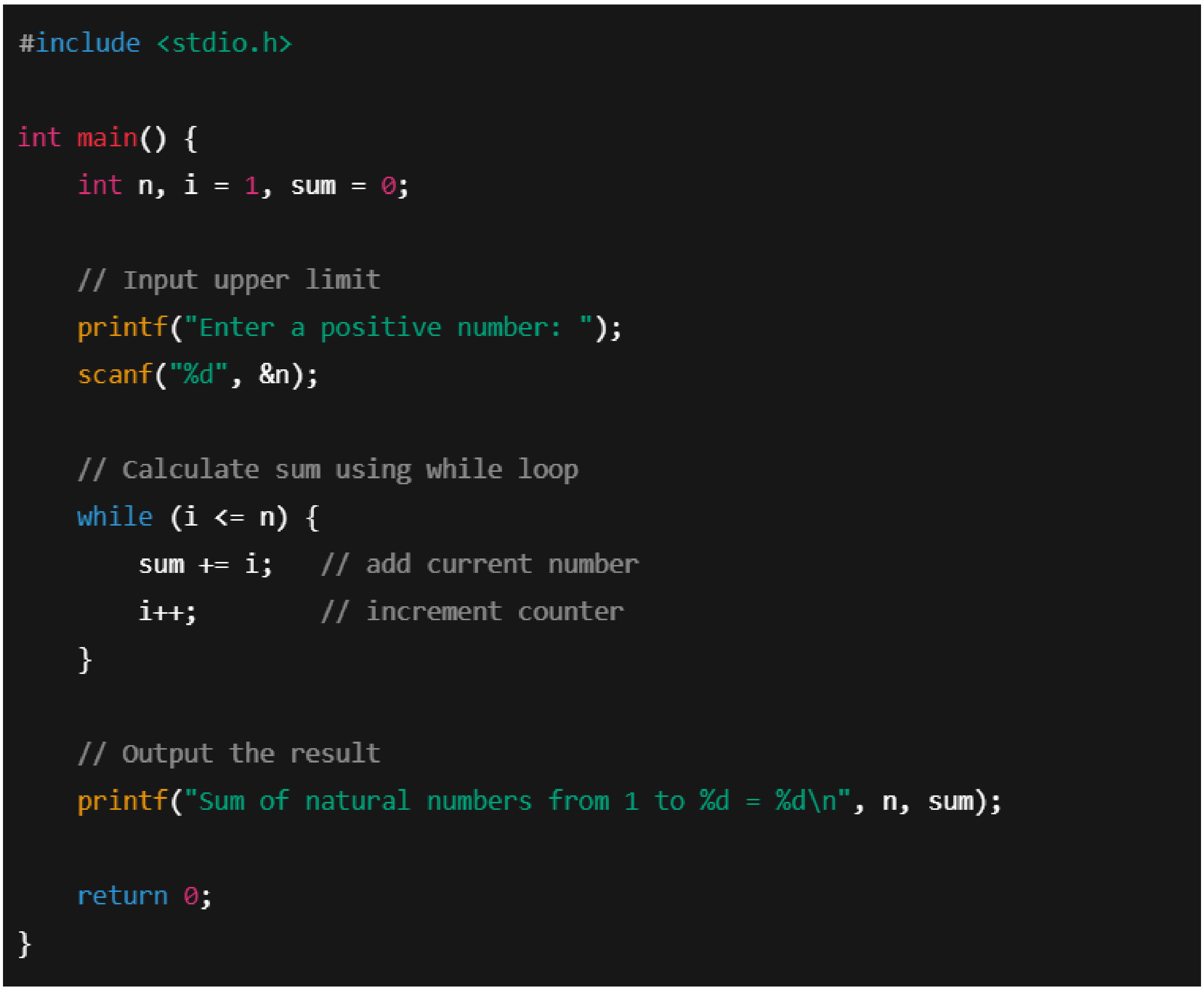
Output:



```
Printing numbers, skip number 3:
1 2 4 5 6 7 8 9 10
```

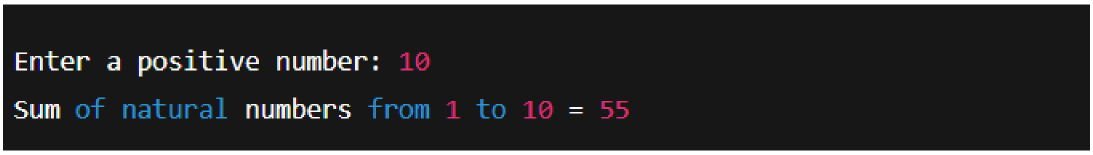
(16). Calculate the Sum of Natural Numbers Using the While Loop.

Ans:



```c
#include <stdio.h>

int main() {
    int n, i = 1, sum = 0;

    // Input upper limit
    printf("Enter a positive number: ");
    scanf("%d", &n);

    // Calculate sum using while loop
    while (i <= n) {
        sum += i;   // add current number
        i++;        // increment counter
    }

    // Output the result
    printf("Sum of natural numbers from 1 to %d = %d\n", n, sum);

    return 0;
}
```

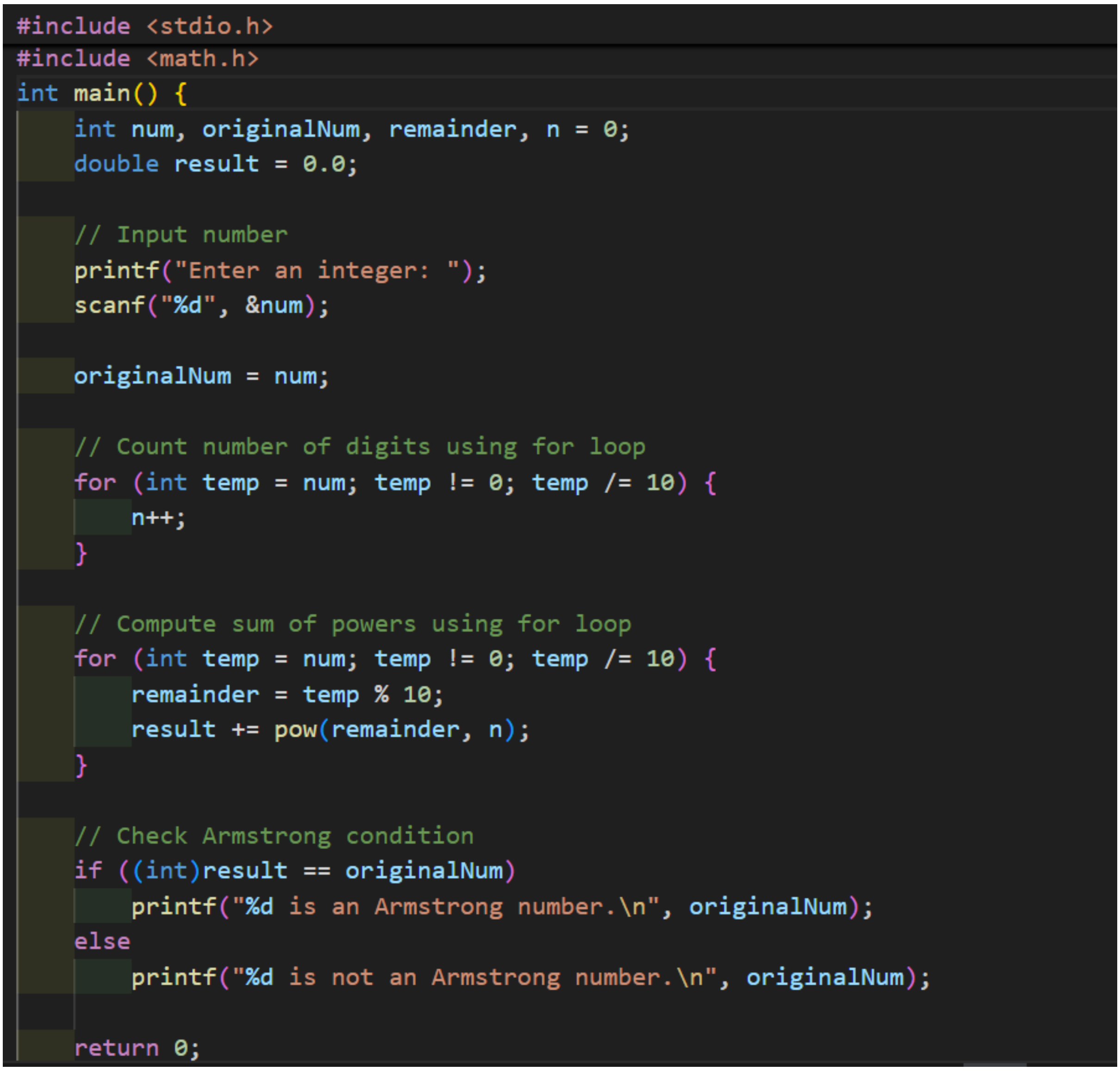
Output:



```
Enter a positive number: 10
Sum of natural numbers from 1 to 10 = 55
```

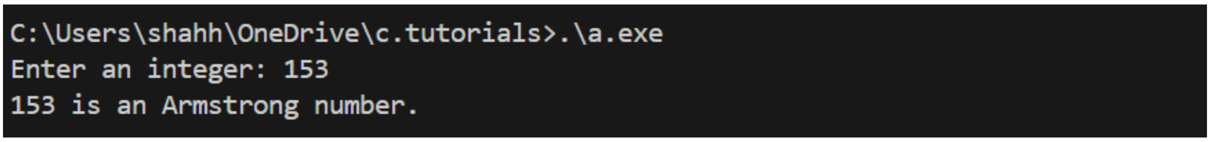
(17). Program of Armstrong Number in C Using For Loop & While Loop.

Ans: Program: Armstrong Number using **for loop:**



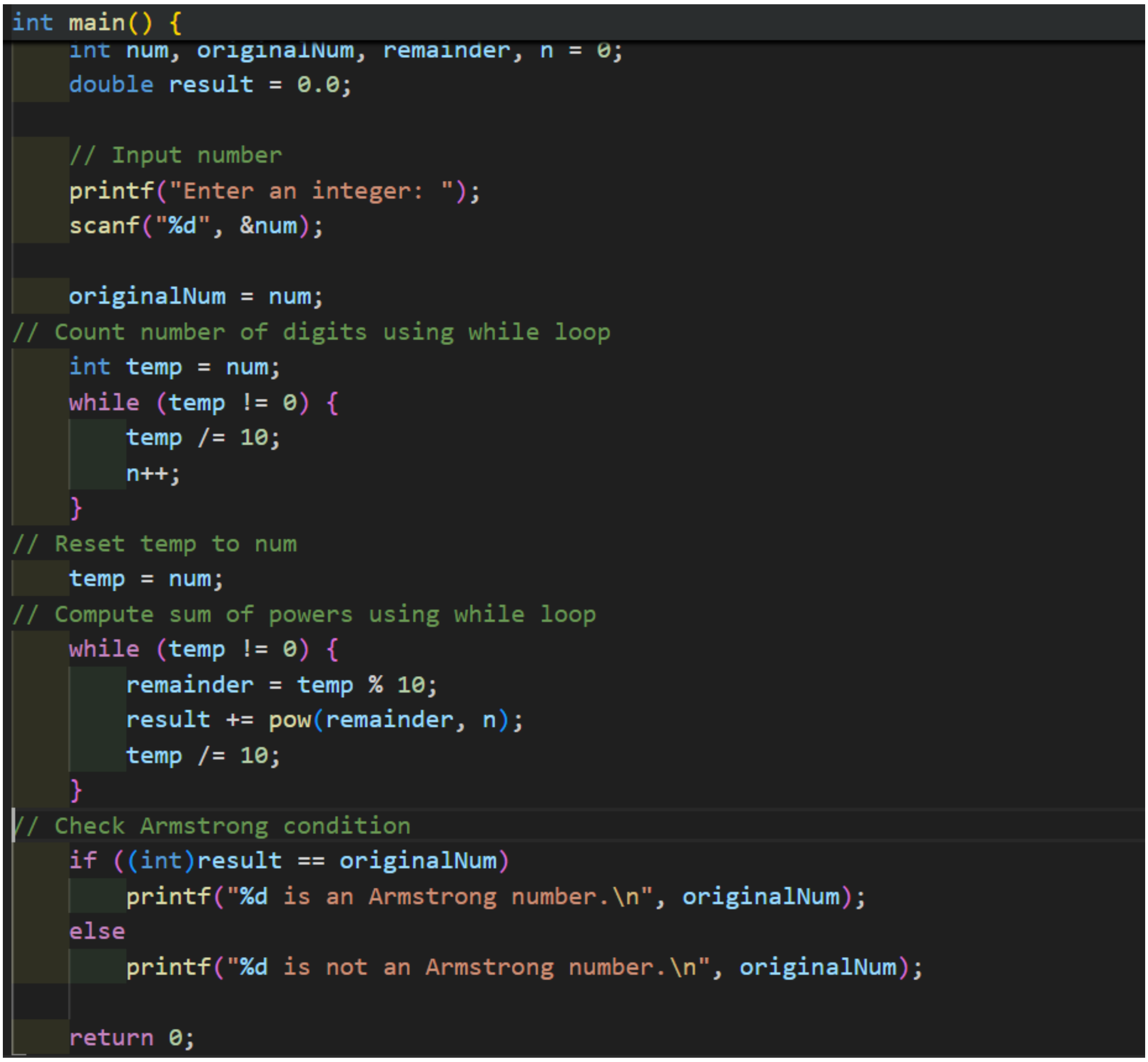
```c
#include <stdio.h>
#include <math.h>
int main() {
    int num, originalNum, remainder, n = 0;
    double result = 0.0;

    // Input number
    printf("Enter an integer: ");
    scanf("%d", &num);

    originalNum = num;

    // Count number of digits using for loop
    for (int temp = num; temp != 0; temp /= 10) {
        n++;
    }

    // Compute sum of powers using for loop
    for (int temp = num; temp != 0; temp /= 10) {
        remainder = temp % 10;
        result += pow(remainder, n);
    }

    // Check Armstrong condition
    if ((int)result == originalNum)
        printf("%d is an Armstrong number.\n", originalNum);
    else
        printf("%d is not an Armstrong number.\n", originalNum);

    return 0;
```

Output:



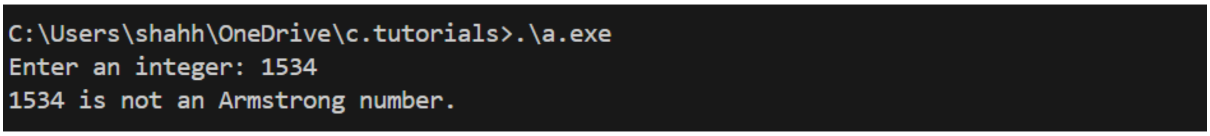
```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Enter an integer: 153
153 is an Armstrong number.
```

Program: Armstrong Number using **while loop:**



```c
int main() {
    int num, originalNum, remainder, n = 0;
    double result = 0.0;

    // Input number
    printf("Enter an integer: ");
    scanf("%d", &num);

    originalNum = num;
// Count number of digits using while loop
    int temp = num;
    while (temp != 0) {
        temp /= 10;
        n++;
    }
// Reset temp to num
    temp = num;
// Compute sum of powers using while loop
    while (temp != 0) {
        remainder = temp % 10;
        result += pow(remainder, n);
        temp /= 10;
    }
// Check Armstrong condition
    if ((int)result == originalNum)
        printf("%d is an Armstrong number.\n", originalNum);
    else
        printf("%d is not an Armstrong number.\n", originalNum);

    return 0;
```

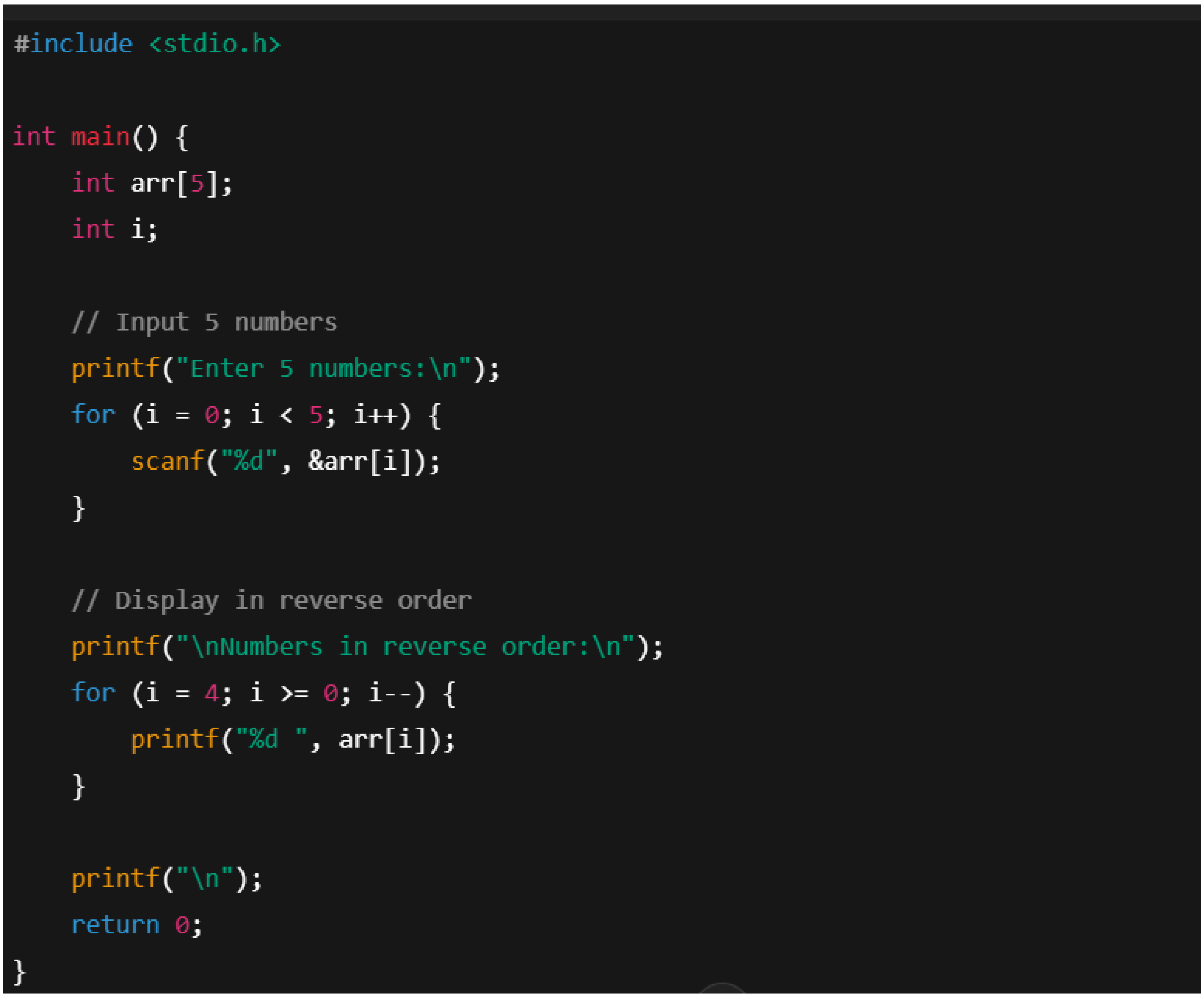
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Enter an integer: 1534
1534 is not an Armstrong number.
```

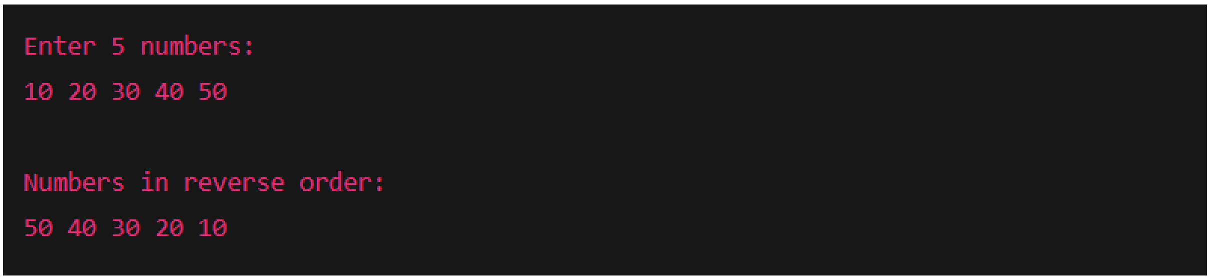
(18). WAP to accept 5 numbers from user and display in reverse order using for loop and array.

Ans:



```c
#include <stdio.h>

int main() {
    int arr[5];
    int i;

    // Input 5 numbers
    printf("Enter 5 numbers:\n");
    for (i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }

    // Display in reverse order
    printf("\nNumbers in reverse order:\n");
    for (i = 4; i >= 0; i--) {
        printf("%d ", arr[i]);
    }

    printf("\n");
    return 0;
}
```

Output:

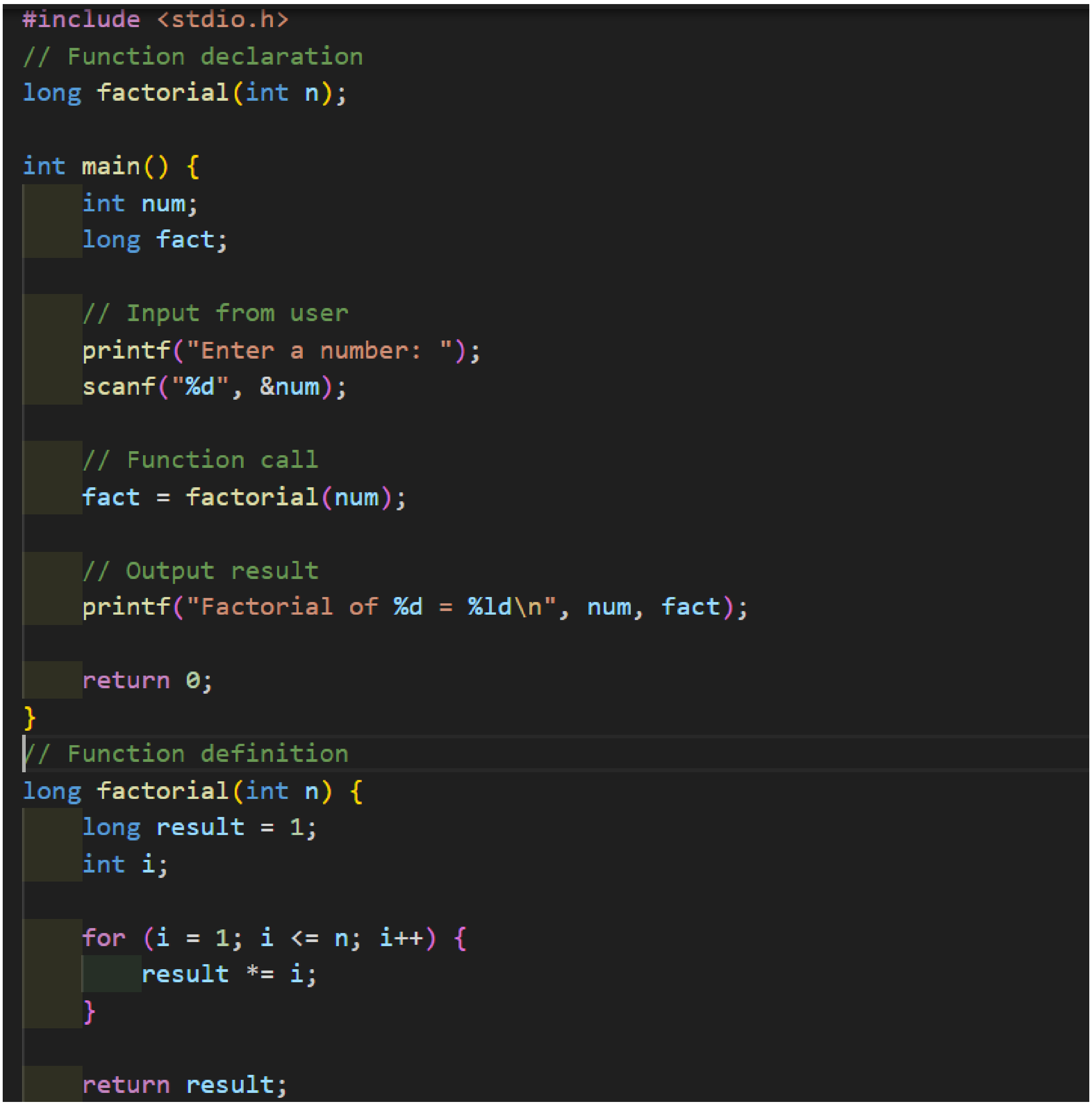


```
Enter 5 numbers:
10 20 30 40 50

Numbers in reverse order:
50 40 30 20 10
```

8. Functions in C:

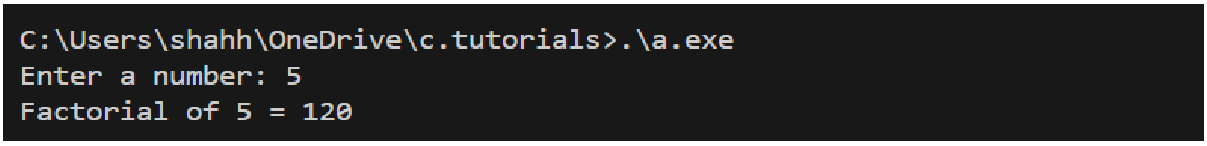
(19). Write a C program that calculates the factorial of a number using a function Include function declaration, definition, and call

Ans:



```c
#include <stdio.h>
// Function declaration
long factorial(int n);

int main() {
    int num;
    long fact;

    // Input from user
    printf("Enter a number: ");
    scanf("%d", &num);

    // Function call
    fact = factorial(num);

    // Output result
    printf("Factorial of %d = %ld\n", num, fact);

    return 0;
}
// Function definition
long factorial(int n) {
    long result = 1;
    int i;

    for (i = 1; i <= n; i++) {
        result *= i;
    }

    return result;
```

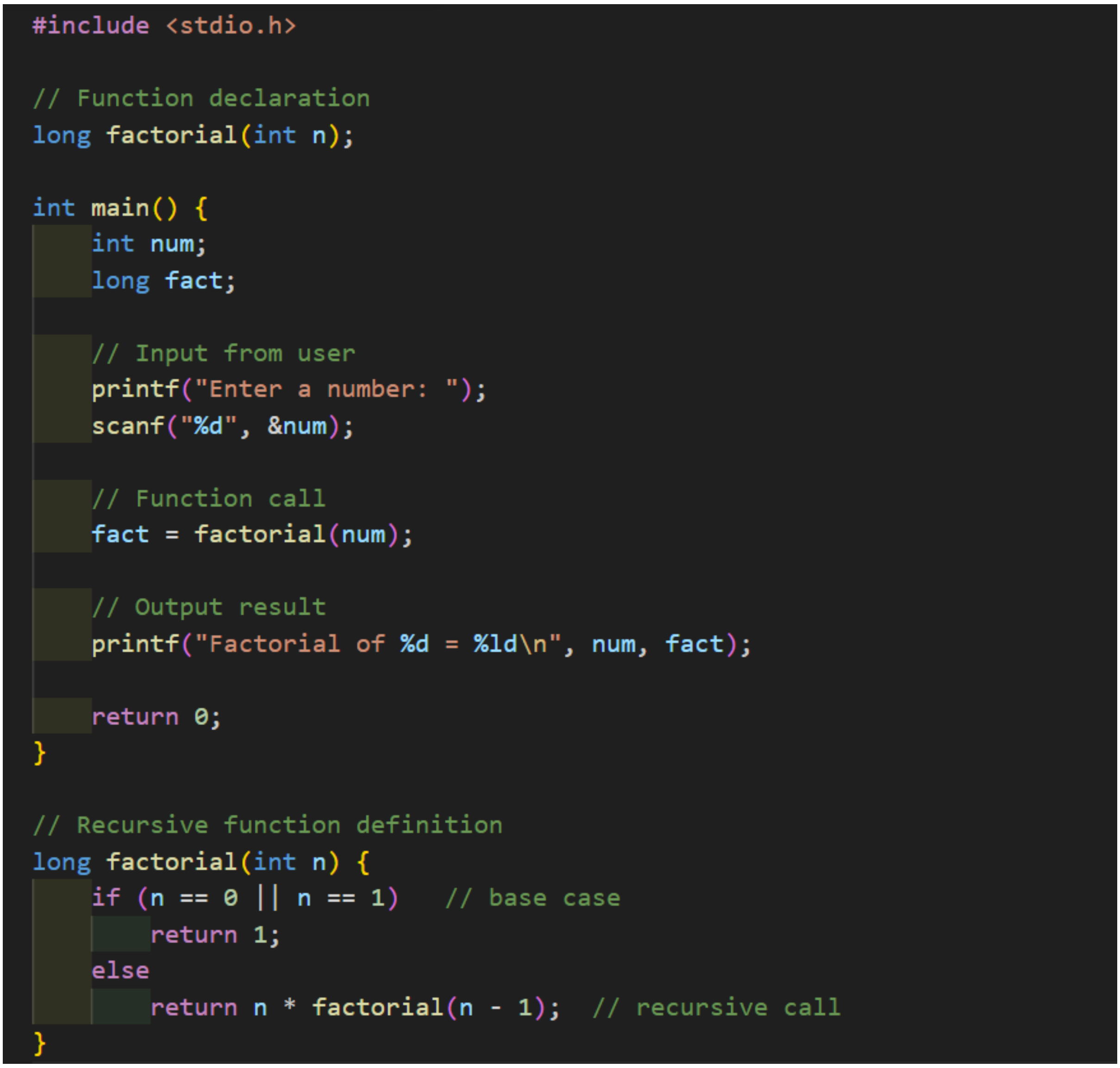
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Enter a number: 5
Factorial of 5 = 120
```

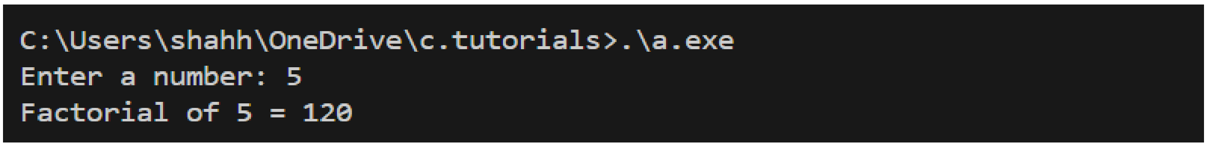
(20). WAP to find factorial using recursion.

Ans:



```c
#include <stdio.h>

// Function declaration
long factorial(int n);

int main() {
    int num;
    long fact;

    // Input from user
    printf("Enter a number: ");
    scanf("%d", &num);

    // Function call
    fact = factorial(num);

    // Output result
    printf("Factorial of %d = %ld\n", num, fact);

    return 0;
}

// Recursive function definition
long factorial(int n) {
    if (n == 0 || n == 1)   // base case
        return 1;
    else
        return n * factorial(n - 1);  // recursive call
}
```

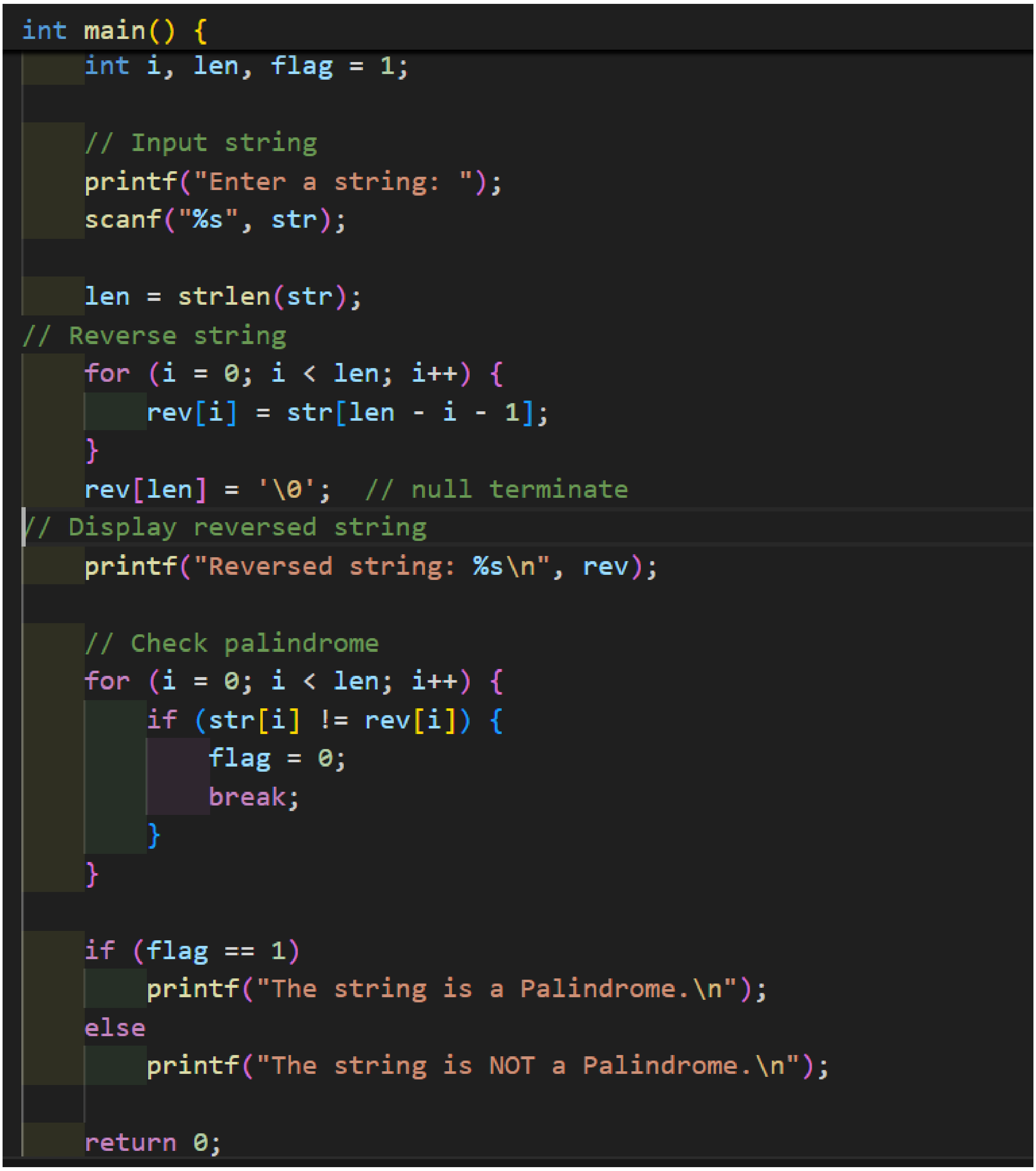
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Enter a number: 5
Factorial of 5 = 120
```

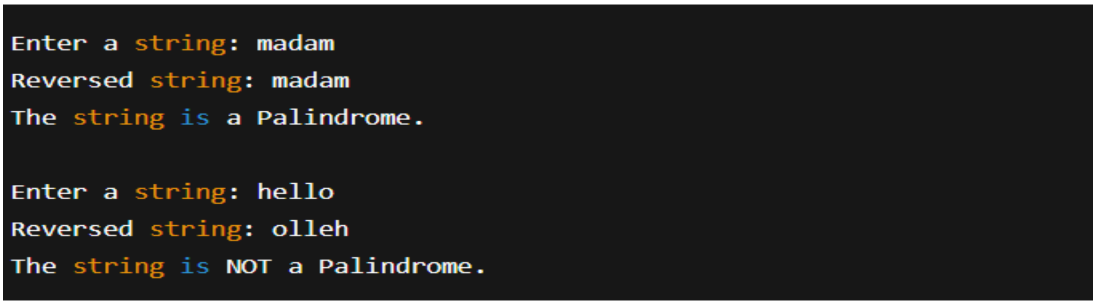
(21). WAP to reverse a string and check that the string is palindrome or no.

Ans:



```
int main() {
    int i, len, flag = 1;

    // Input string
    printf("Enter a string: ");
    scanf("%s", str);

    len = strlen(str);
// Reverse string
    for (i = 0; i < len; i++) {
        rev[i] = str[len - i - 1];
    }
    rev[len] = '\0';  // null terminate
// Display reversed string
    printf("Reversed string: %s\n", rev);

    // Check palindrome
    for (i = 0; i < len; i++) {
        if (str[i] != rev[i]) {
            flag = 0;
            break;
        }
    }

    if (flag == 1)
        printf("The string is a Palindrome.\n");
    else
        printf("The string is NOT a Palindrome.\n");

    return 0;
```

Output:

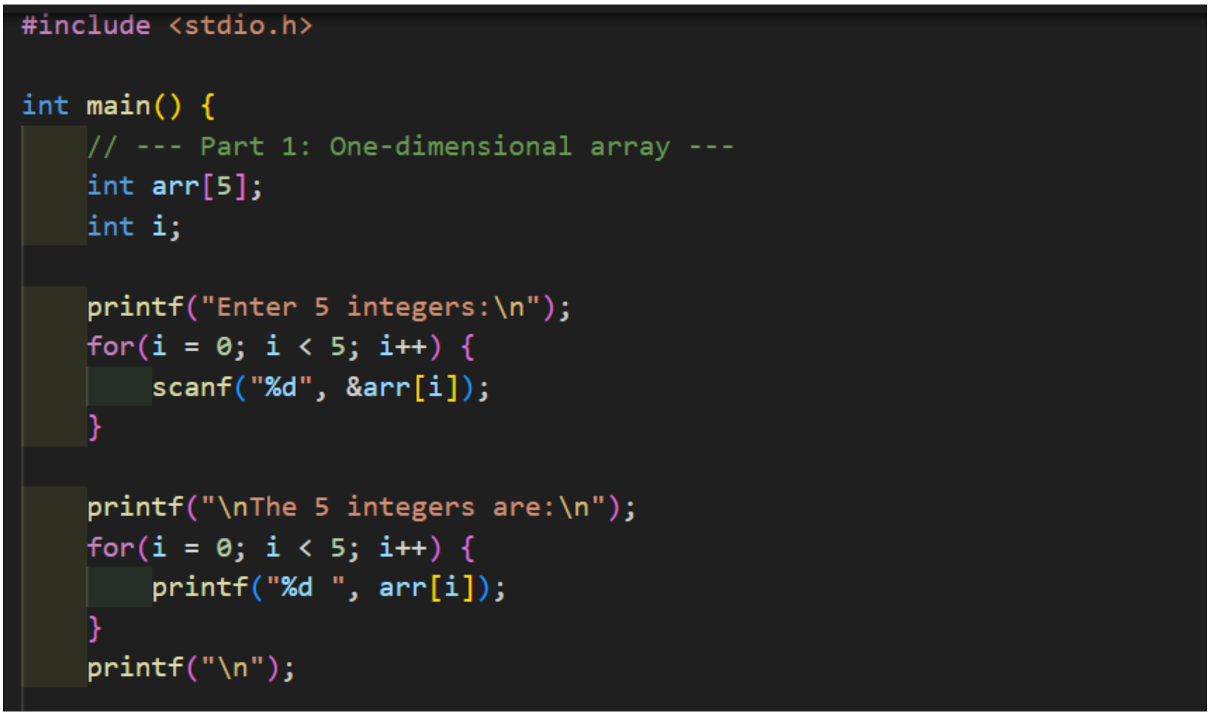


```
Enter a string: madam
Reversed string: madam
The string is a Palindrome.

Enter a string: hello
Reversed string: olleh
The string is NOT a Palindrome.
```

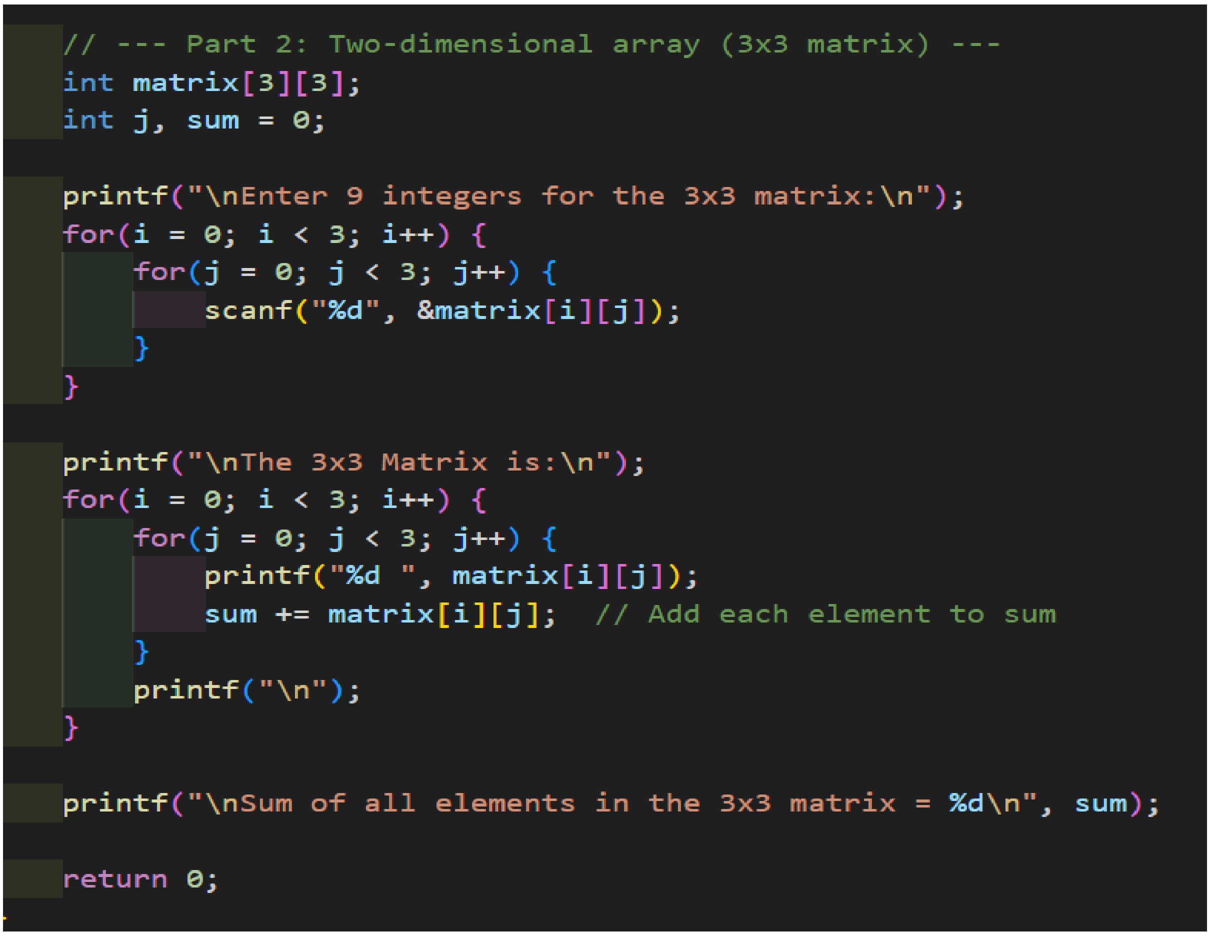
9. Array in C:

(22). Write a C program that stores 5 integers in a one-dimensional array and prints them. Extend this to handle a two-dimensional array (3x3 matrix) and calculate the sum of all elements.

Ans:

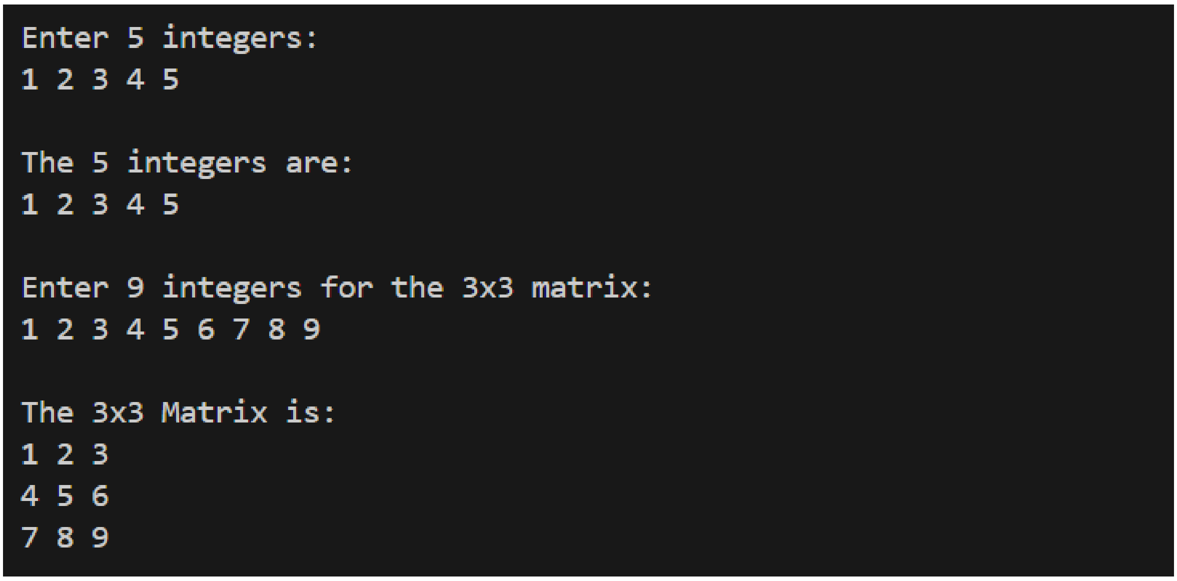


```c
#include <stdio.h>

int main() {
    // --- Part 1: One-dimensional array ---
    int arr[5];
    int i;

    printf("Enter 5 integers:\n");
    for(i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }

    printf("\nThe 5 integers are:\n");
    for(i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
```



```c
    // --- Part 2: Two-dimensional array (3x3 matrix) ---
    int matrix[3][3];
    int j, sum = 0;

    printf("\nEnter 9 integers for the 3x3 matrix:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }

    printf("\nThe 3x3 Matrix is:\n");
    for(i = 0; i < 3; i++) {
        for(j = 0; j < 3; j++) {
            printf("%d ", matrix[i][j]);
            sum += matrix[i][j];  // Add each element to sum
        }
        printf("\n");
    }

    printf("\nSum of all elements in the 3x3 matrix = %d\n", sum);

    return 0;
```

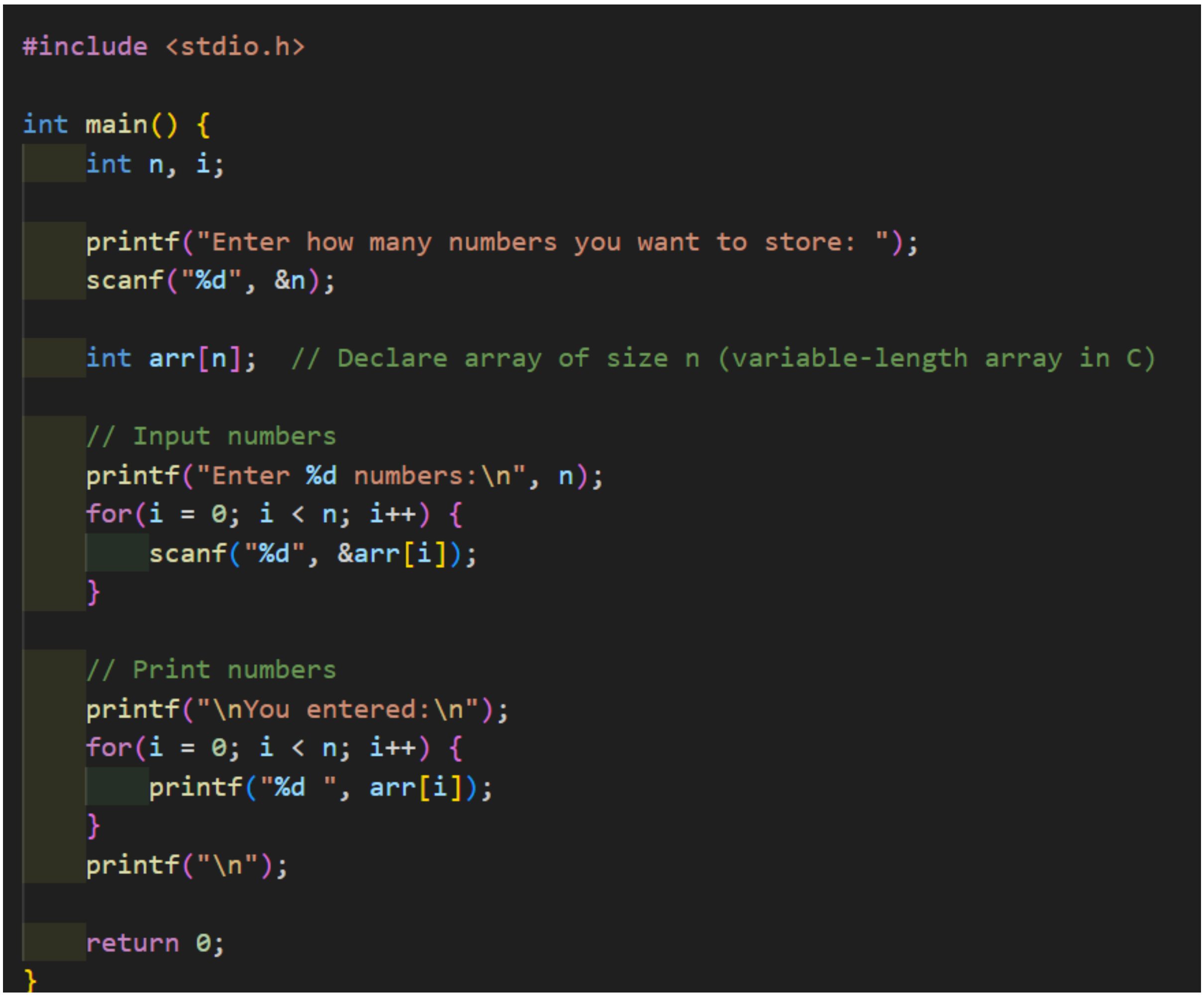
Output:



```
Enter 5 integers:
1 2 3 4 5

The 5 integers are:
1 2 3 4 5

Enter 9 integers for the 3x3 matrix:
1 2 3 4 5 6 7 8 9

The 3x3 Matrix is:
1 2 3
4 5 6
7 8 9
```

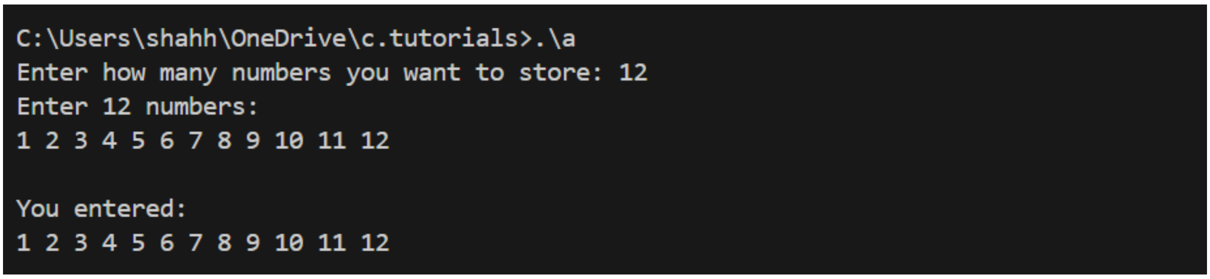
(23). Accept number from user store in array.

Ans:



```c
#include <stdio.h>

int main() {
    int n, i;

    printf("Enter how many numbers you want to store: ");
    scanf("%d", &n);

    int arr[n];  // Declare array of size n (variable-length array in C)

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

    // Print numbers
    printf("\nYou entered:\n");
    for(i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

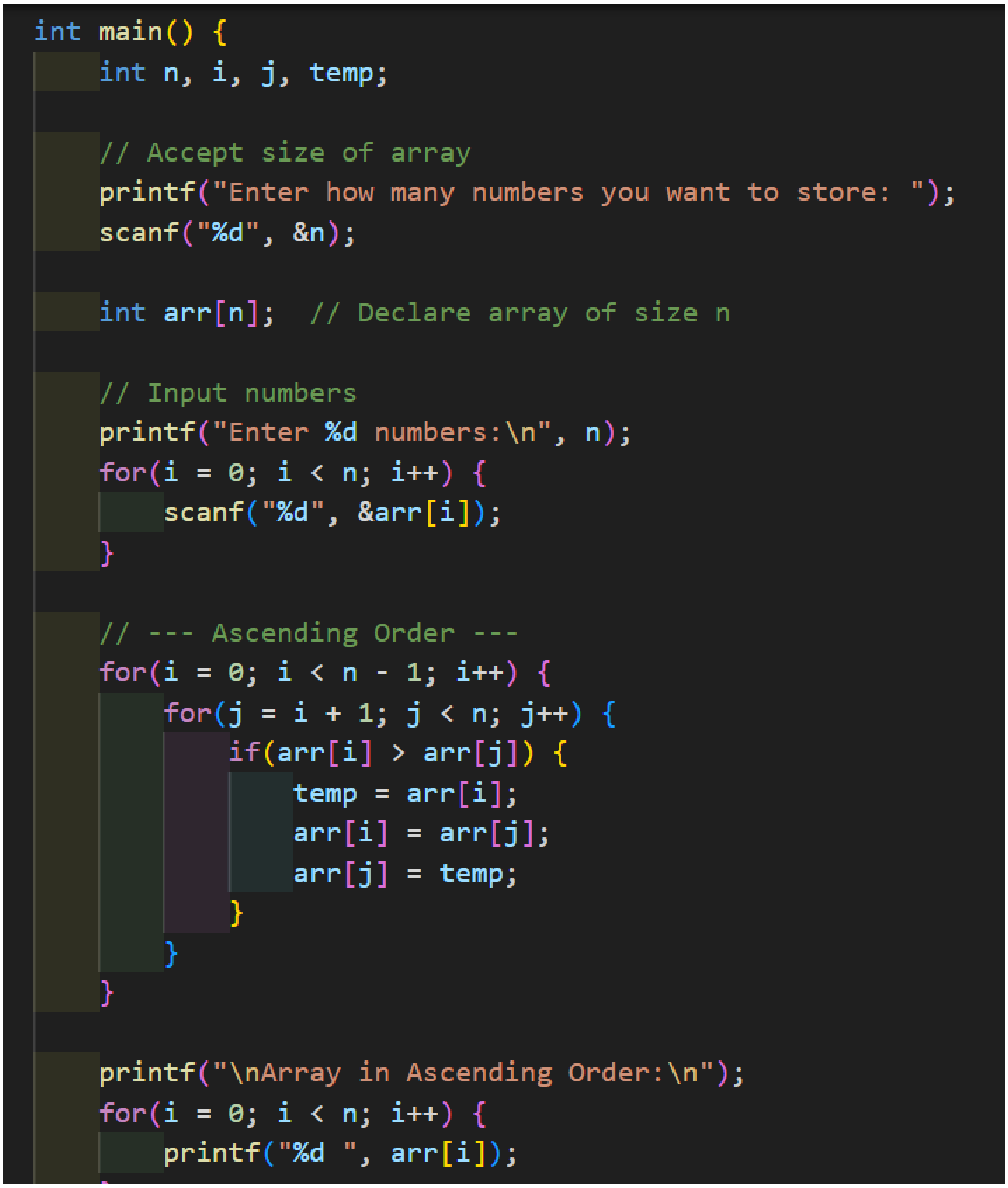
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a
Enter how many numbers you want to store: 12
Enter 12 numbers:
1 2 3 4 5 6 7 8 9 10 11 12

You entered:
1 2 3 4 5 6 7 8 9 10 11 12
```

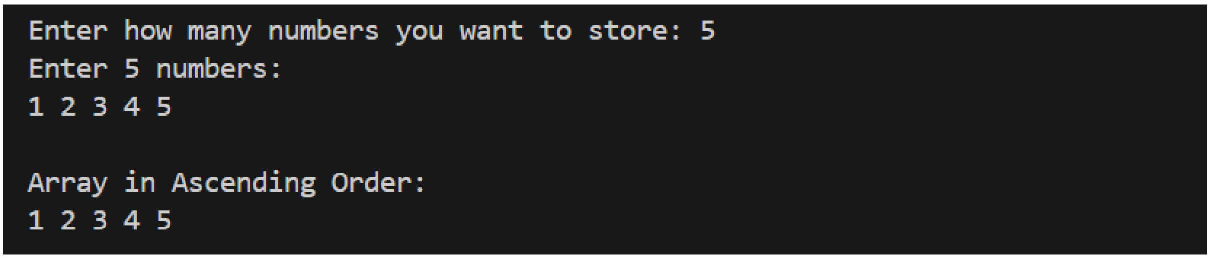
(24). Convert array into asce and dec order.

Ans: Ascending order:



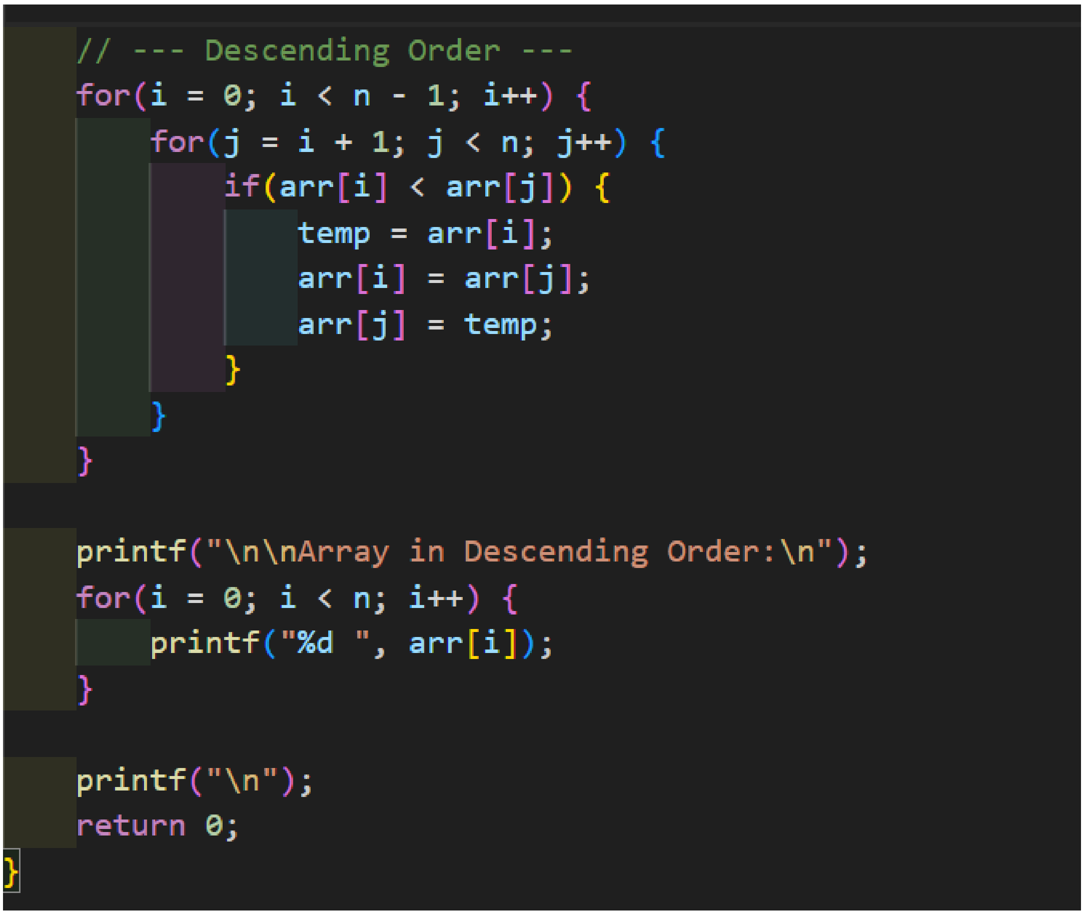
```c
int main() {
    int n, i, j, temp;

    // Accept size of array
    printf("Enter how many numbers you want to store: ");
    scanf("%d", &n);

    int arr[n];  // Declare array of size n

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

    // --- Ascending Order ---
    for(i = 0; i < n - 1; i++) {
        for(j = i + 1; j < n; j++) {
            if(arr[i] > arr[j]) {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }

    printf("\nArray in Ascending Order:\n");
    for(i = 0; i < n; i++) {
        printf("%d ", arr[i]);
```

Output:



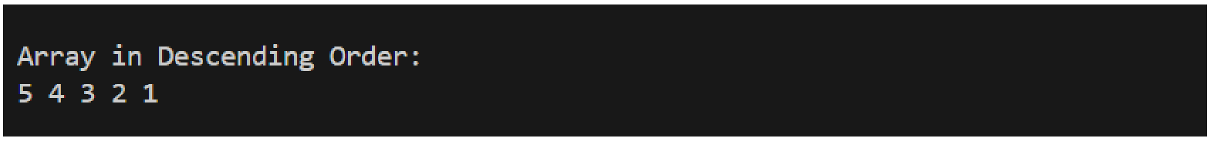
```
Enter how many numbers you want to store: 5
Enter 5 numbers:
1 2 3 4 5

Array in Ascending Order:
1 2 3 4 5
```

Descending order:



```c
    // --- Descending Order ---
    for(i = 0; i < n - 1; i++) {
        for(j = i + 1; j < n; j++) {
            if(arr[i] < arr[j]) {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }

    printf("\n\nArray in Descending Order:\n");
    for(i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

    printf("\n");
    return 0;
}
```

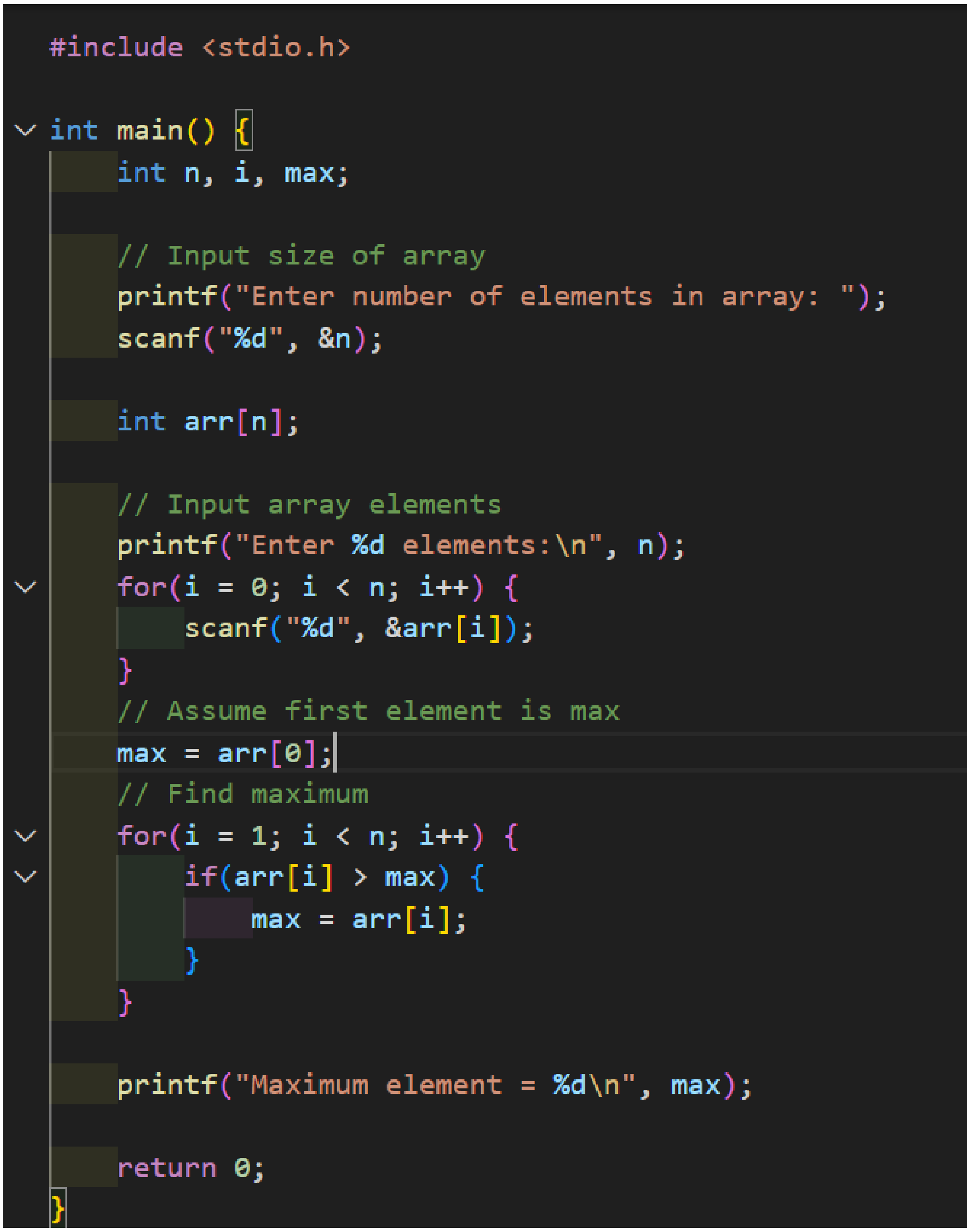
Output:



```

Array in Descending Order:
5 4 3 2 1
```

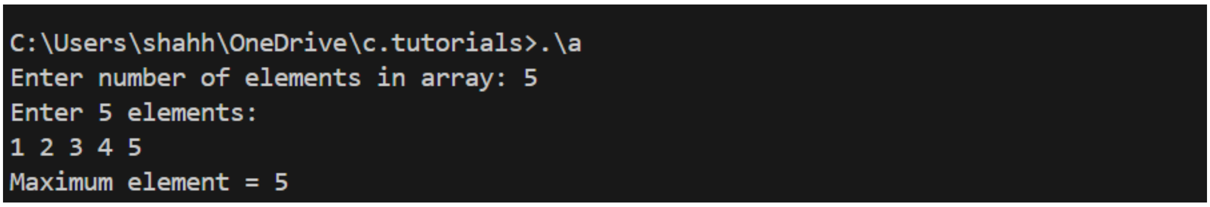
(25). Find max element from the array.

Ans: Code :-



```c
#include <stdio.h>

int main() {
    int n, i, max;

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

    int arr[n];

    // Input array elements
    printf("Enter %d elements:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    // Assume first element is max
    max = arr[0];
    // Find maximum
    for(i = 1; i < n; i++) {
        if(arr[i] > max) {
            max = arr[i];
        }
    }

    printf("Maximum element = %d\n", max);

    return 0;
}
```

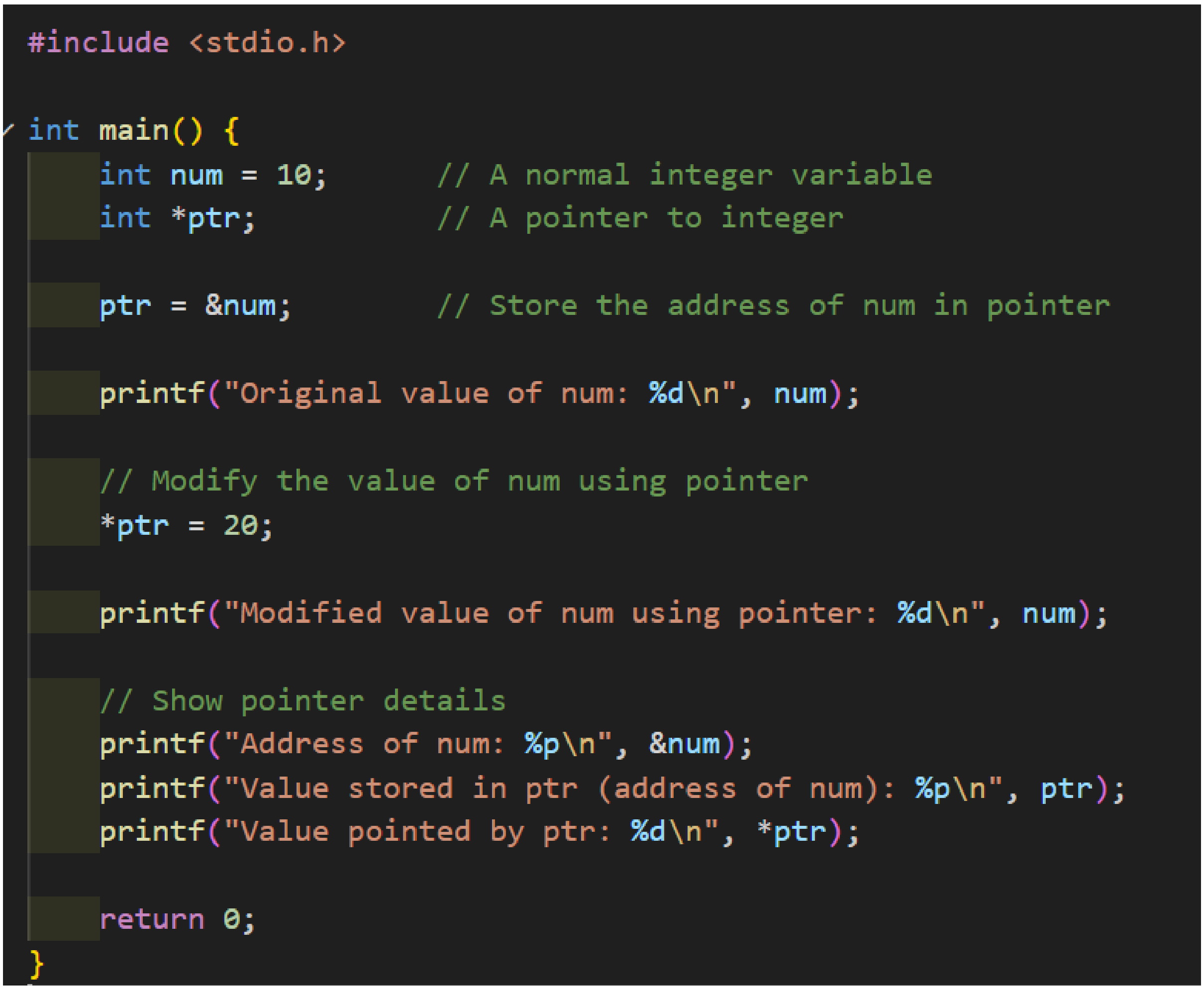
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a
Enter number of elements in array: 5
Enter 5 elements:
1 2 3 4 5
Maximum element = 5
```

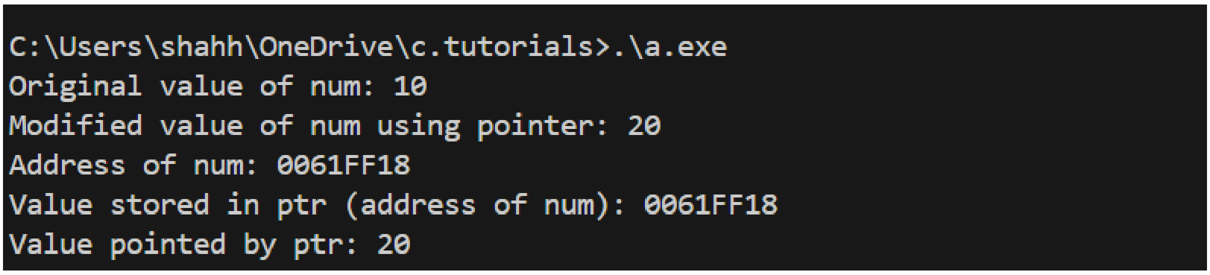
(26). Write a C program to demonstrate pointer usage. Use a pointer to modify the value of a variable and print the result.

Ans:



```c
#include <stdio.h>

int main() {
    int num = 10;       // A normal integer variable
    int *ptr;           // A pointer to integer

    ptr = &num;         // Store the address of num in pointer

    printf("Original value of num: %d\n", num);

    // Modify the value of num using pointer
    *ptr = 20;

    printf("Modified value of num using pointer: %d\n", num);

    // Show pointer details
    printf("Address of num: %p\n", &num);
    printf("Value stored in ptr (address of num): %p\n", ptr);
    printf("Value pointed by ptr: %d\n", *ptr);

    return 0;
}
```

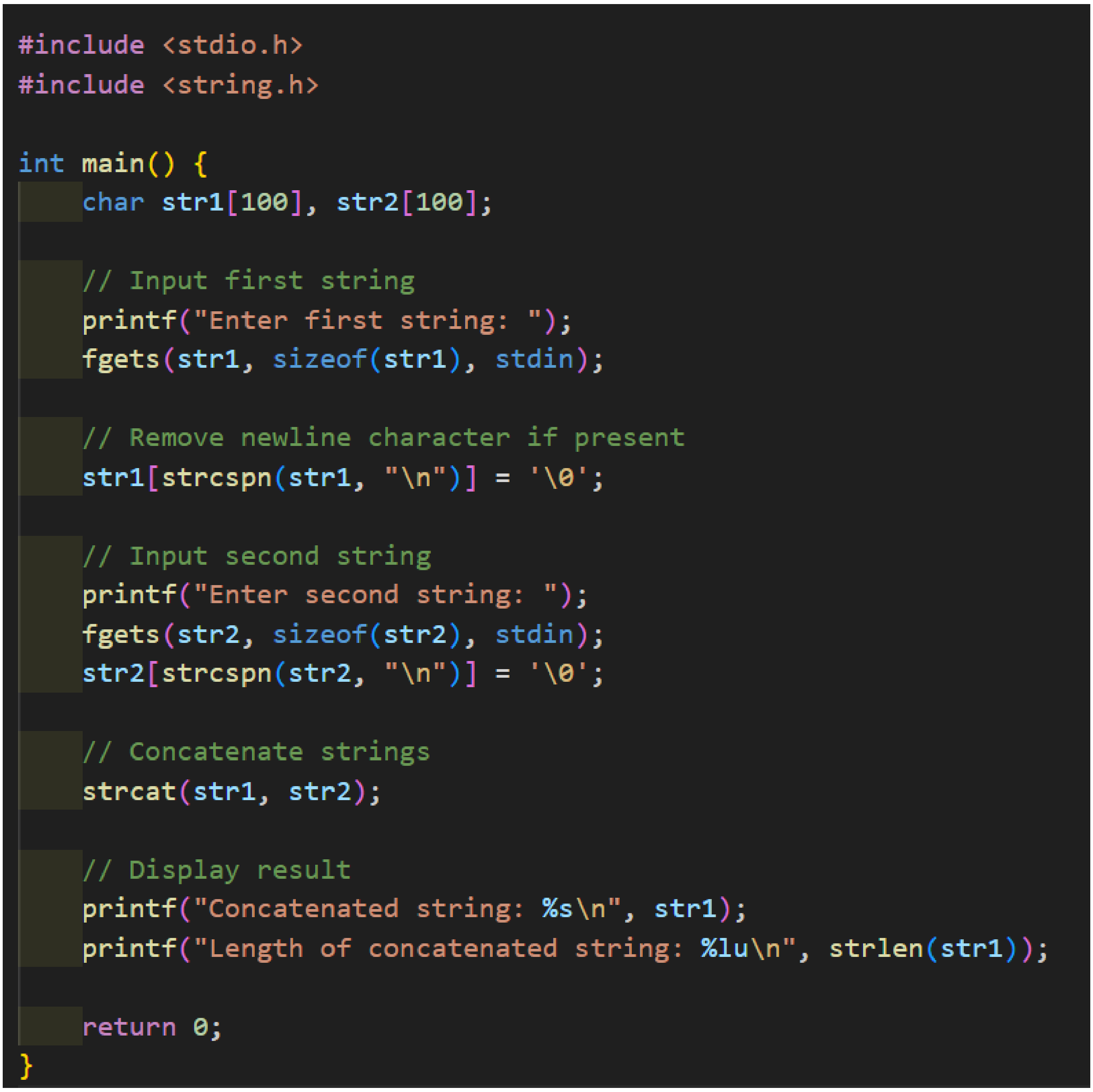
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Original value of num: 10
Modified value of num using pointer: 20
Address of num: 0061FF18
Value stored in ptr (address of num): 0061FF18
Value pointed by ptr: 20
```

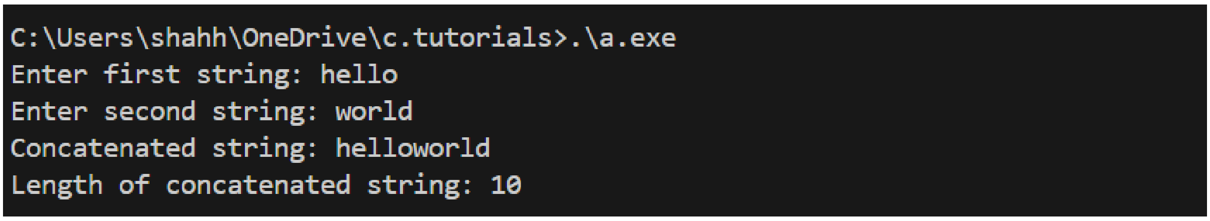
(27). Write a C program that takes two strings from the user and concatenates them using strcat(). Display the concatenated string and its length using strlen().

Ans:



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

int main() {
    char str1[100], str2[100];

    // Input first string
    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);

    // Remove newline character if present
    str1[strcspn(str1, "\n")] = '\0';

    // Input second string
    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);
    str2[strcspn(str2, "\n")] = '\0';

    // Concatenate strings
    strcat(str1, str2);

    // Display result
    printf("Concatenated string: %s\n", str1);
    printf("Length of concatenated string: %lu\n", strlen(str1));

    return 0;
}
```

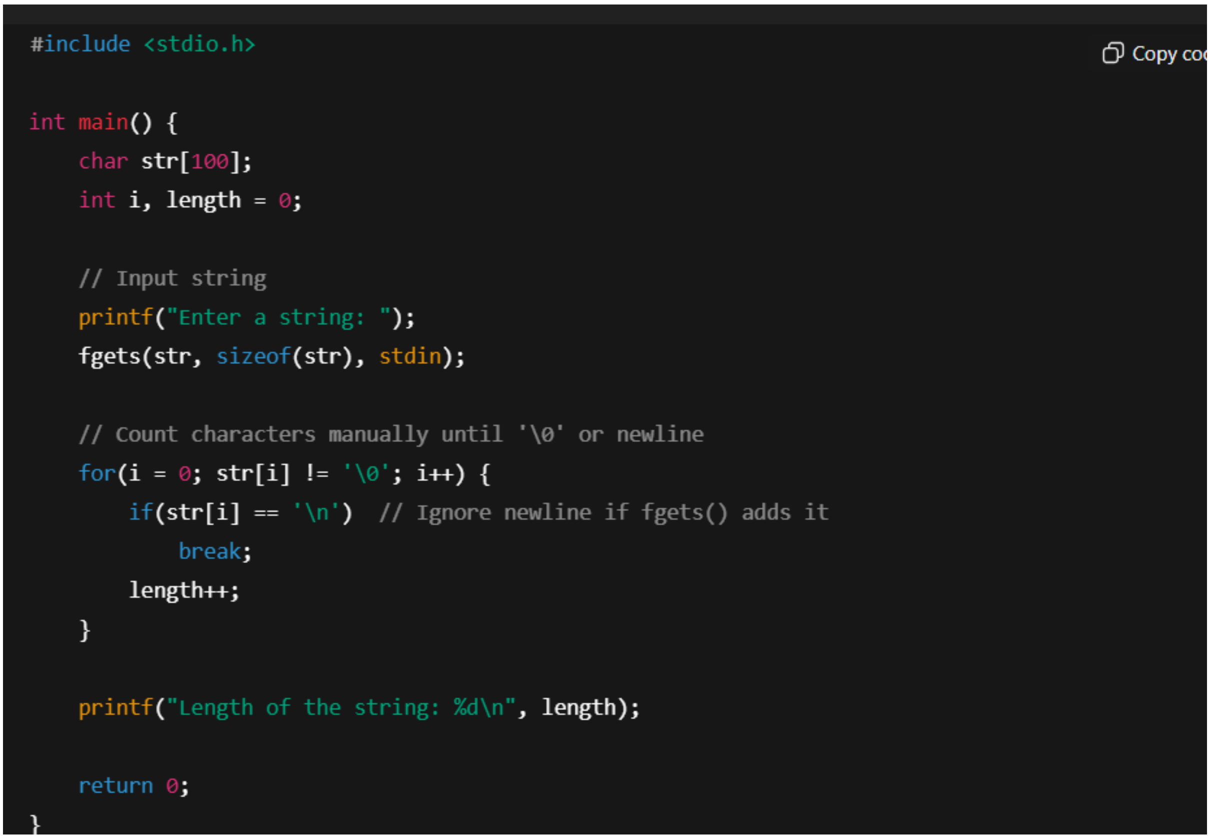
Output:



```
C:\Users\shahh\OneDrive\c.tutorials>.\a.exe
Enter first string: hello
Enter second string: world
Concatenated string: helloworld
Length of concatenated string: 10
```

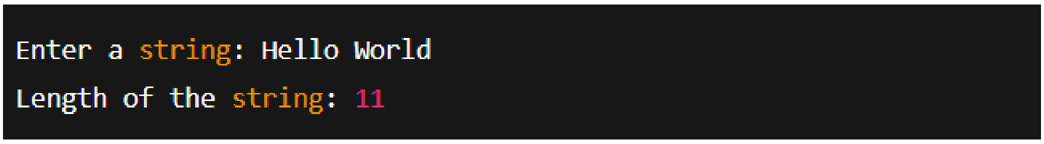
(27). Find length of string which is entered by user without using inbuilt function.

Ans:



```
#include <stdio.h>

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

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

    // Count characters manually until '\0' or newline
    for(i = 0; str[i] != '\0'; i++) {
        if(str[i] == '\n')  // Ignore newline if fgets() adds it
            break;
        length++;
    }

    printf("Length of the string: %d\n", length);

    return 0;
}
```

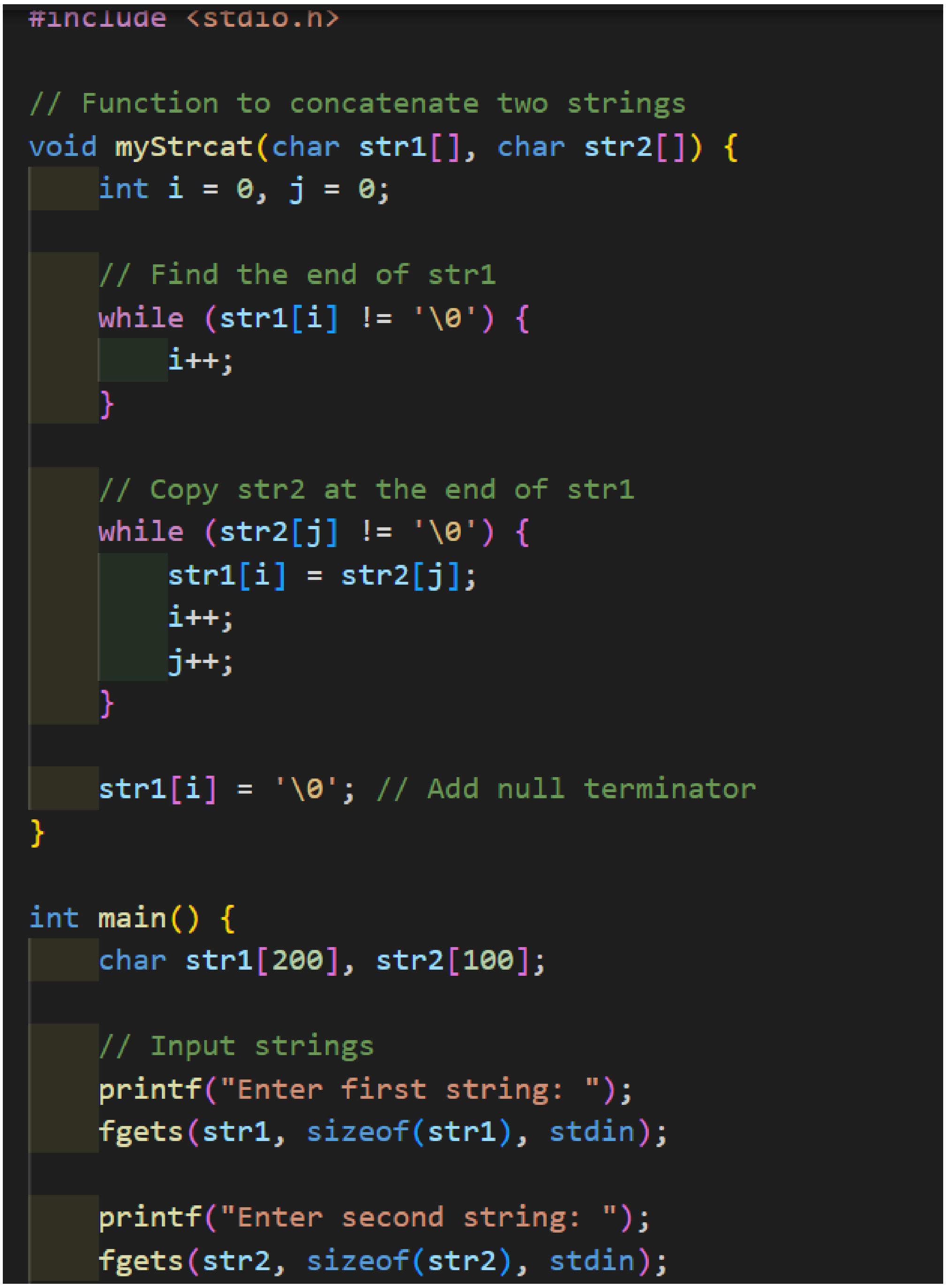
Output:



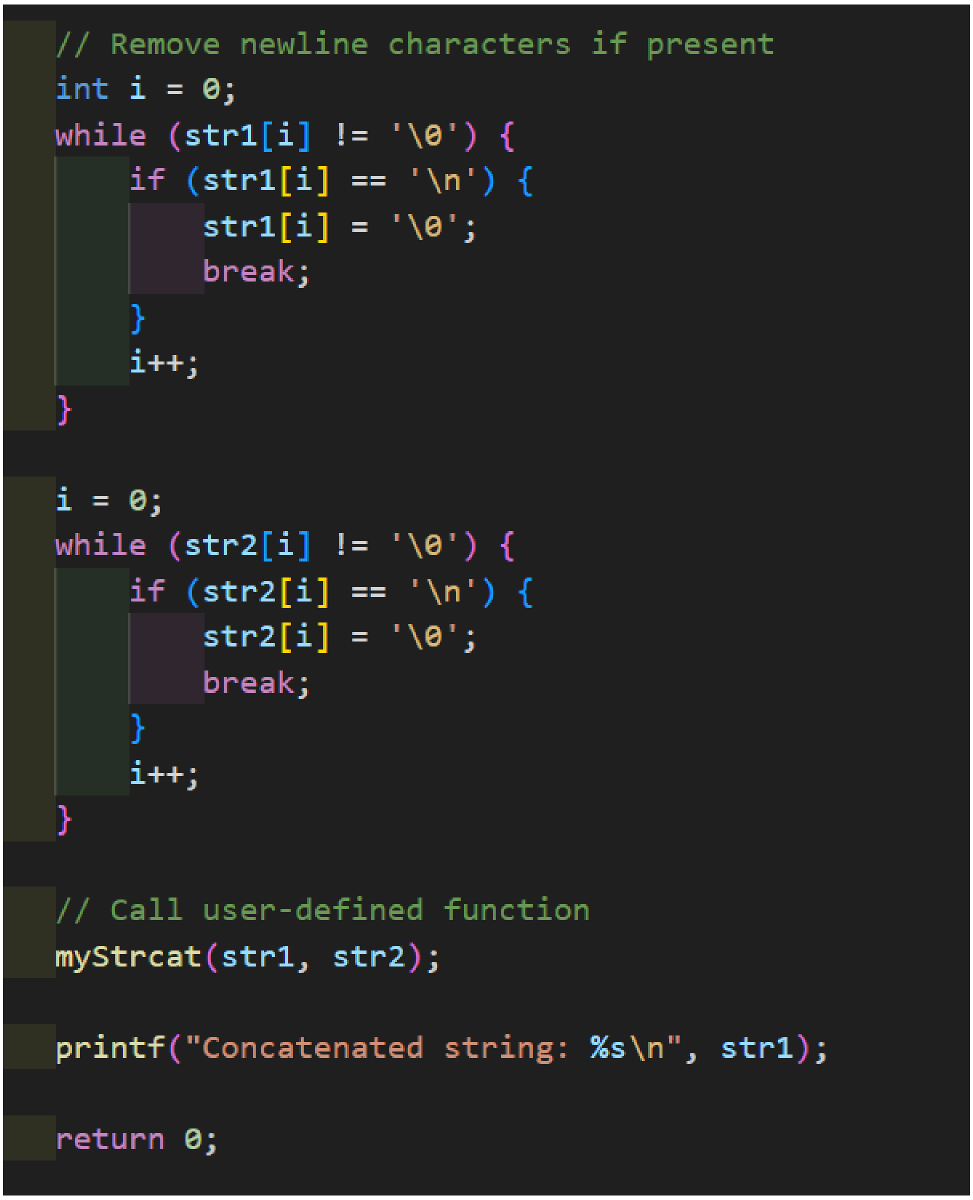
```
Enter a string: Hello World
Length of the string: 11
```

(28). Join 2 strings using of user defined function without using inbuilt function.

Ans:

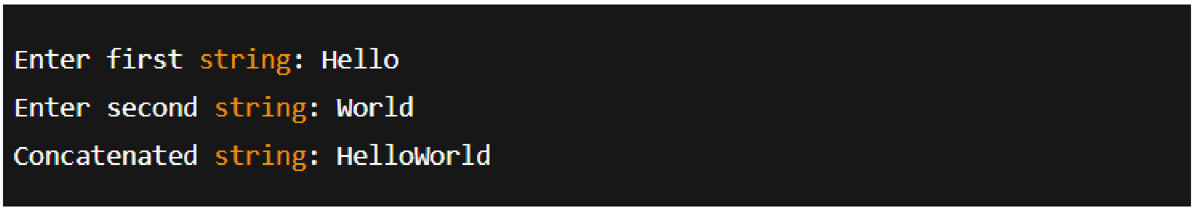


```c
#include <stdio.h>

// Function to concatenate two strings
void myStrcat(char str1[], char str2[]) {
    int i = 0, j = 0;

    // Find the end of str1
    while (str1[i] != '\0') {
        i++;
    }

    // Copy str2 at the end of str1
    while (str2[j] != '\0') {
        str1[i] = str2[j];
        i++;
        j++;
    }

    str1[i] = '\0'; // Add null terminator
}

int main() {
    char str1[200], str2[100];

    // Input strings
    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);

    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);
```



```c
// Remove newline characters if present
int i = 0;
while (str1[i] != '\0') {
    if (str1[i] == '\n') {
        str1[i] = '\0';
        break;
    }
    i++;
}

i = 0;
while (str2[i] != '\0') {
    if (str2[i] == '\n') {
        str2[i] = '\0';
        break;
    }
    i++;
}

// Call user-defined function
myStrcat(str1, str2);

printf("Concatenated string: %s\n", str1);

return 0;
```

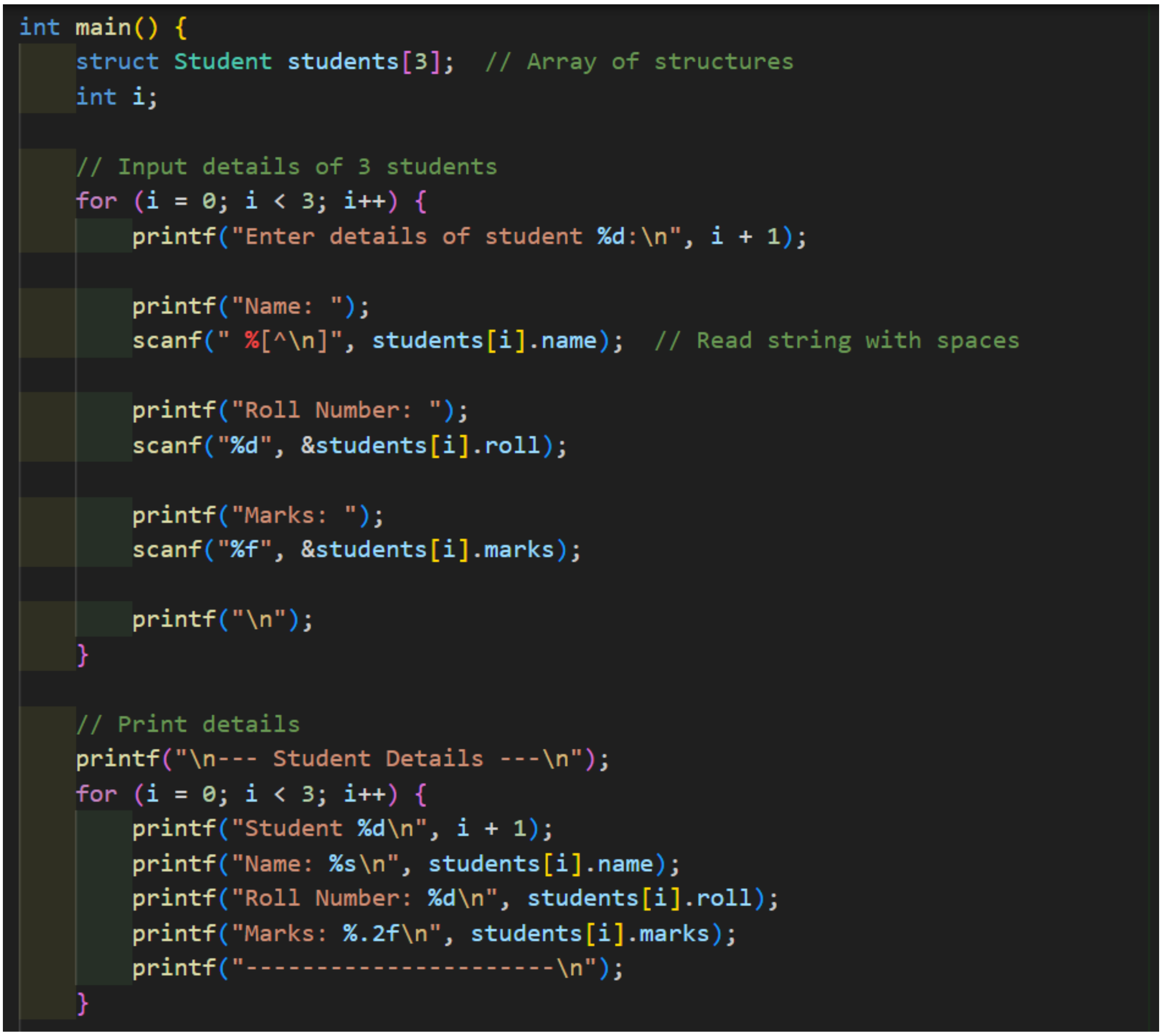
Output:



```
Enter first string: Hello
Enter second string: World
Concatenated string: HelloWorld
```

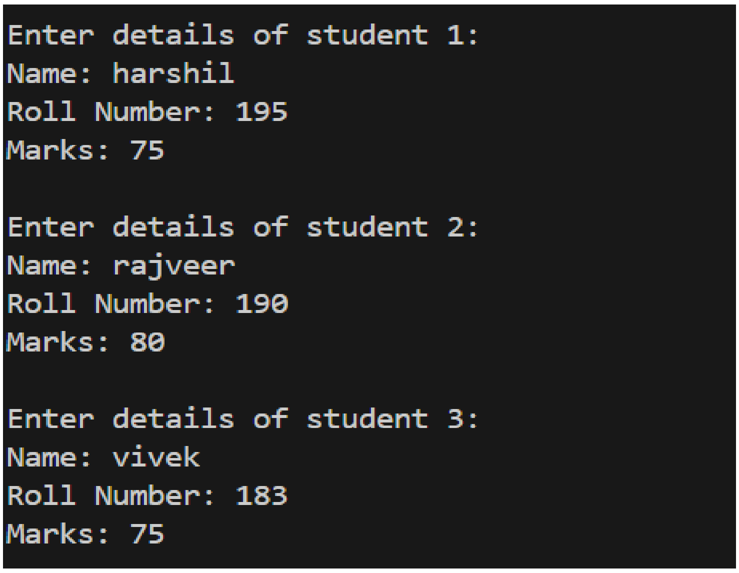
(29). Write a C program that defines a structure to store a student's details (name, roll number, and marks). Use an array of structures to store details of 3 students and print them.

Ans:

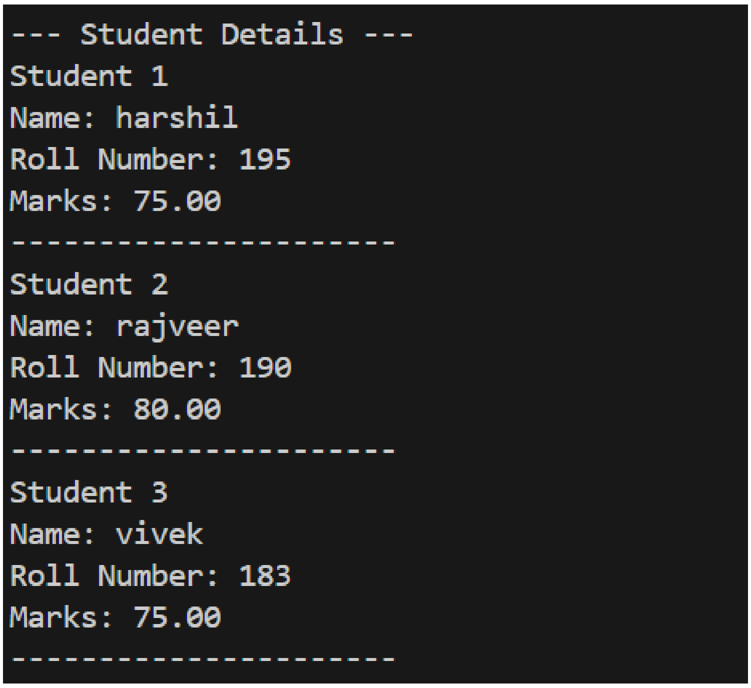


```
int main() {
    struct Student students[3];  // Array of structures
    int i;

    // Input details of 3 students
    for (i = 0; i < 3; i++) {
        printf("Enter details of student %d:\n", i + 1);

        printf("Name: ");
        scanf(" %[^\n]", students[i].name);  // Read string with spaces

        printf("Roll Number: ");
        scanf("%d", &students[i].roll);

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

        printf("\n");
    }

    // Print details
    printf("\n--- Student Details ---\n");
    for (i = 0; i < 3; i++) {
        printf("Student %d\n", i + 1);
        printf("Name: %s\n", students[i].name);
        printf("Roll Number: %d\n", students[i].roll);
        printf("Marks: %.2f\n", students[i].marks);
        printf("---------------------\n");
    }
```

Output:



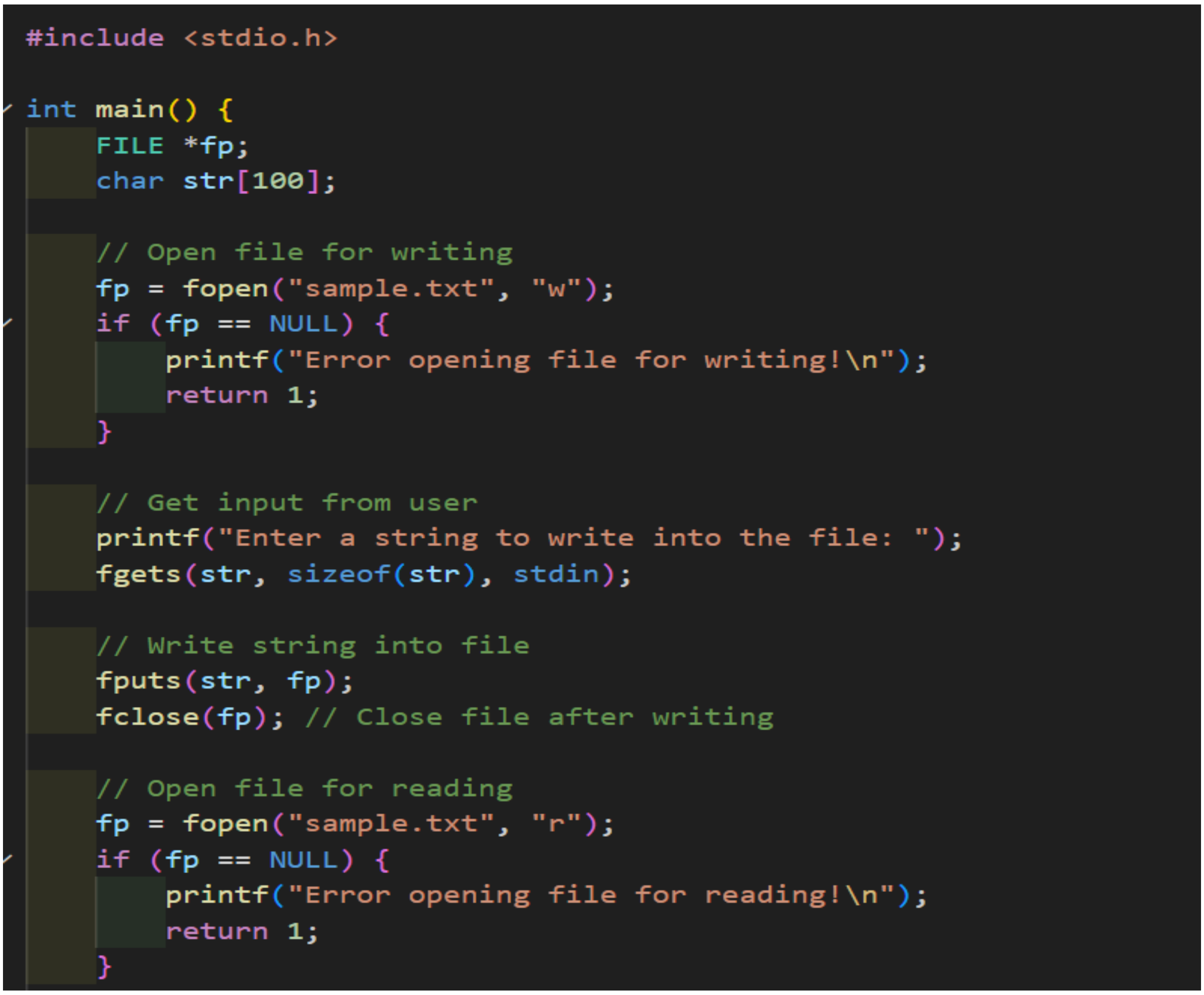
```
Enter details of student 1:
Name: harshil
Roll Number: 195
Marks: 75

Enter details of student 2:
Name: rajveer
Roll Number: 190
Marks: 80

Enter details of student 3:
Name: vivek
Roll Number: 183
Marks: 75
```



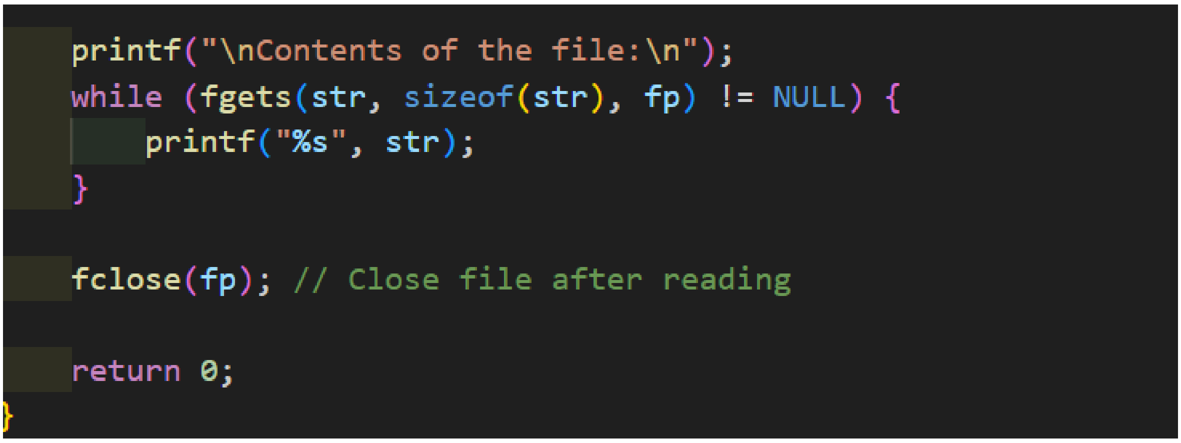
```
--- Student Details ---
Student 1
Name: harshil
Roll Number: 195
Marks: 75.00
----------------------
Student 2
Name: rajveer
Roll Number: 190
Marks: 80.00
----------------------
Student 3
Name: vivek
Roll Number: 183
Marks: 75.00
----------------------
```

(30). Write a C program to create a file, write a string into it, close the file, then open the file again to read and display its contents.

Ans:

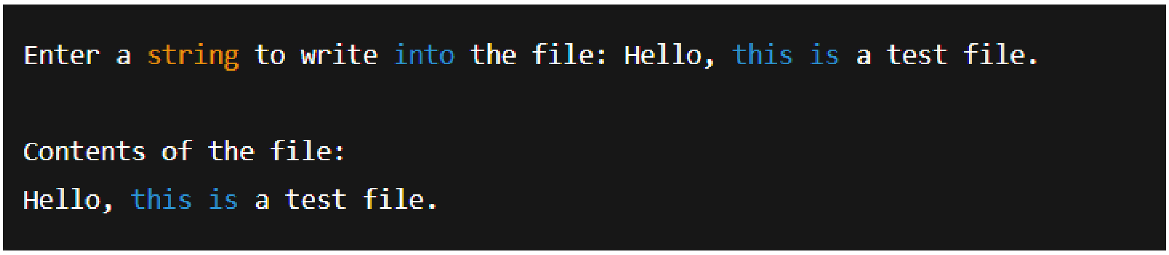


```c
#include <stdio.h>

int main() {
    FILE *fp;
    char str[100];

    // Open file for writing
    fp = fopen("sample.txt", "w");
    if (fp == NULL) {
        printf("Error opening file for writing!\n");
        return 1;
    }

    // Get input from user
    printf("Enter a string to write into the file: ");
    fgets(str, sizeof(str), stdin);

    // Write string into file
    fputs(str, fp);
    fclose(fp); // Close file after writing

    // Open file for reading
    fp = fopen("sample.txt", "r");
    if (fp == NULL) {
        printf("Error opening file for reading!\n");
        return 1;
    }
```



```
    printf("\nContents of the file:\n");
    while (fgets(str, sizeof(str), fp) != NULL) {
        printf("%s", str);
    }

    fclose(fp); // Close file after reading

    return 0;
}
```

Output:



```
Enter a string to write into the file: Hello, this is a test file.

Contents of the file:
Hello, this is a test file.
```