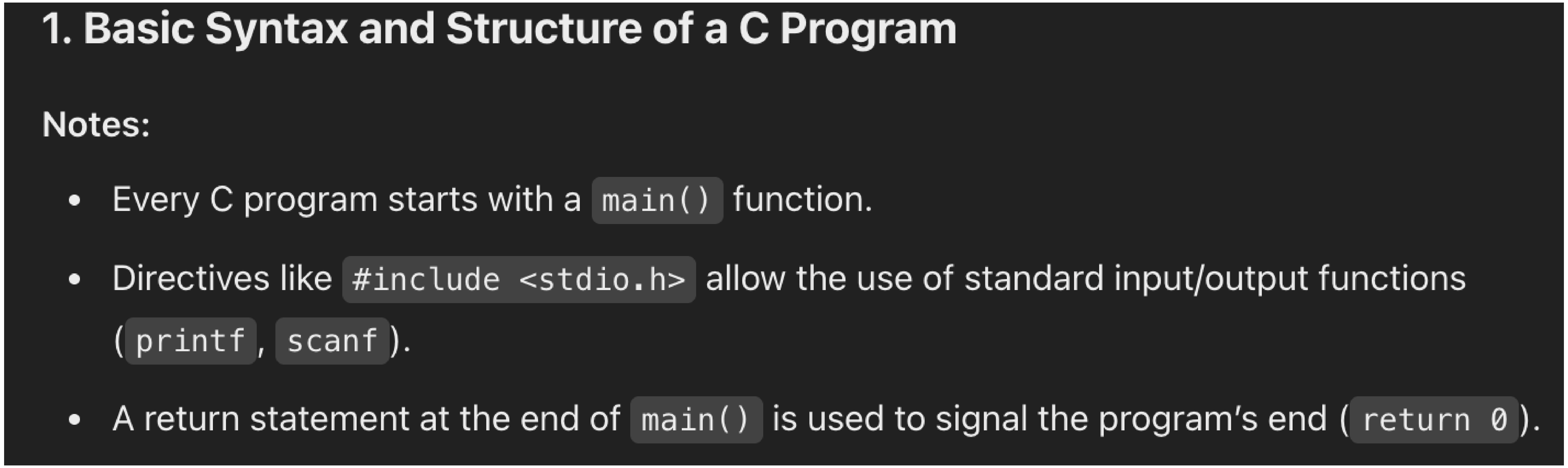
C PROGRAMMING

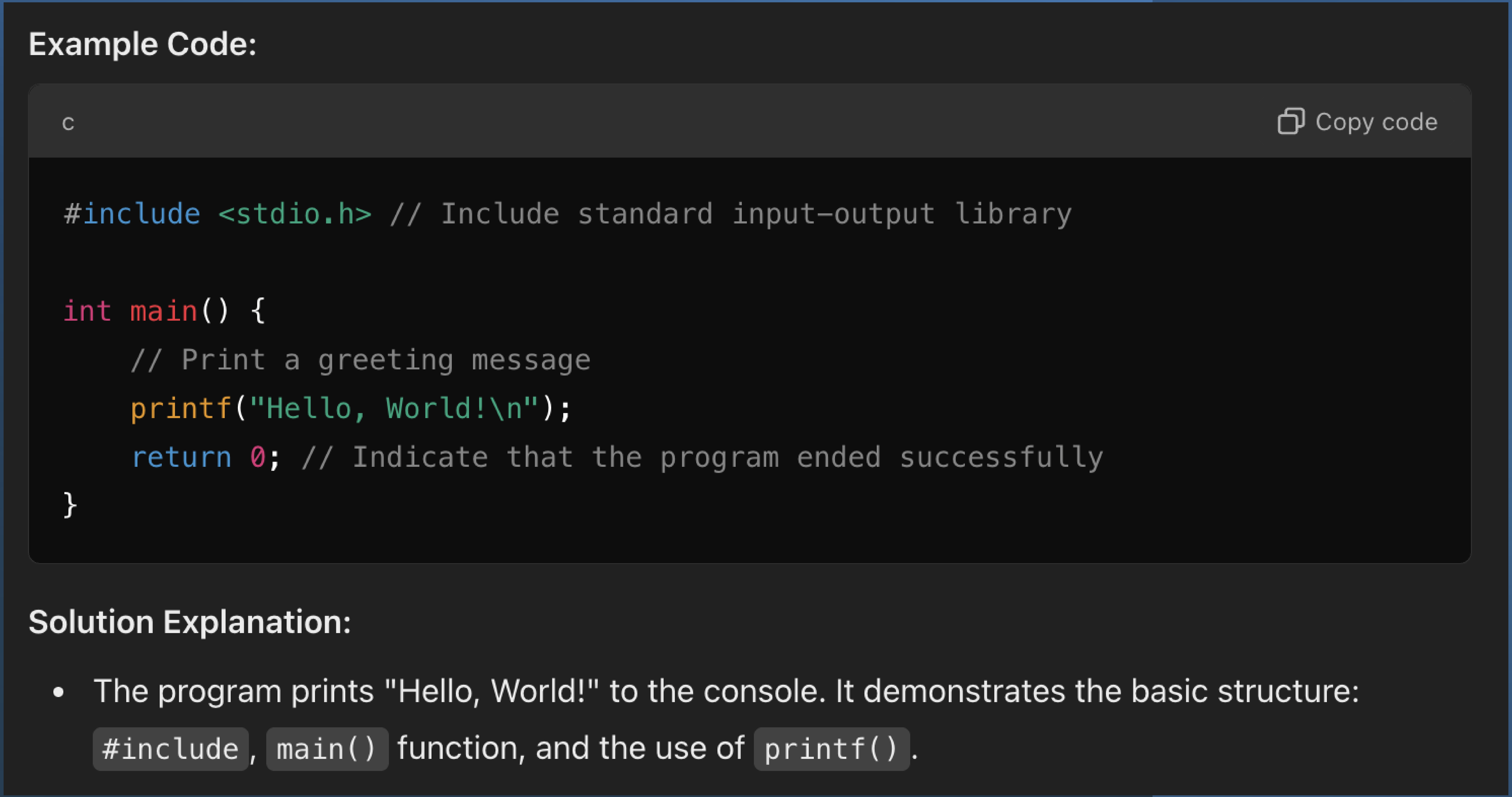
By Lungsom Lamnio



1. Basic Syntax and Structure of a C Program

Notes:

- Every C program starts with a `main()` function.
- Directives like `#include <stdio.h>` allow the use of standard input/output functions (`printf`, `scanf`).
- A return statement at the end of `main()` is used to signal the program's end (`return 0`).

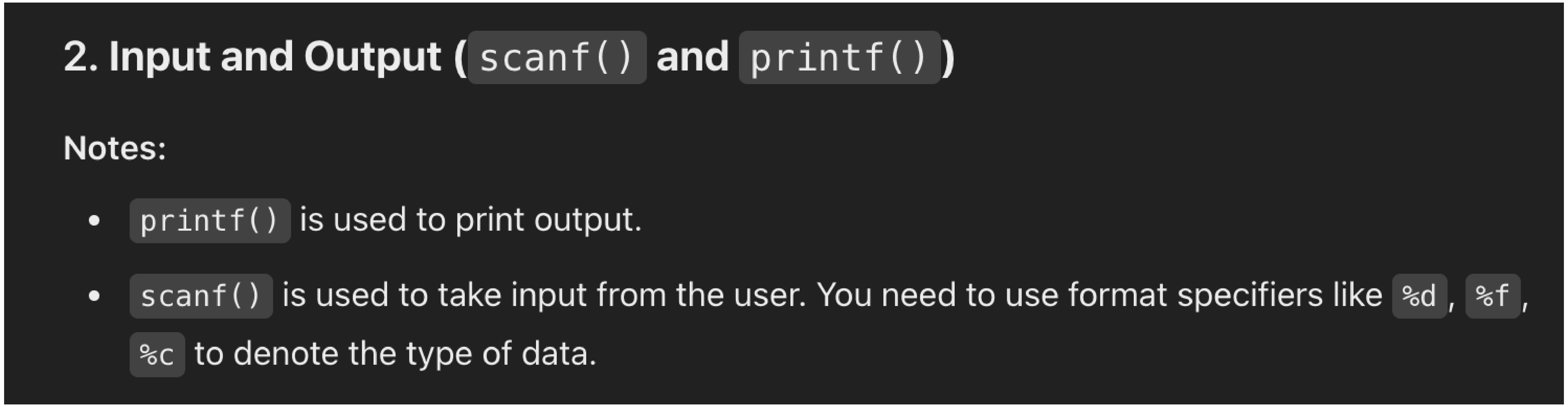


Example Code:

```c
#include <stdio.h> // Include standard input-output library

int main() {
    // Print a greeting message
    printf("Hello, World!\n");
    return 0; // Indicate that the program ended successfully
}
```

Solution Explanation:

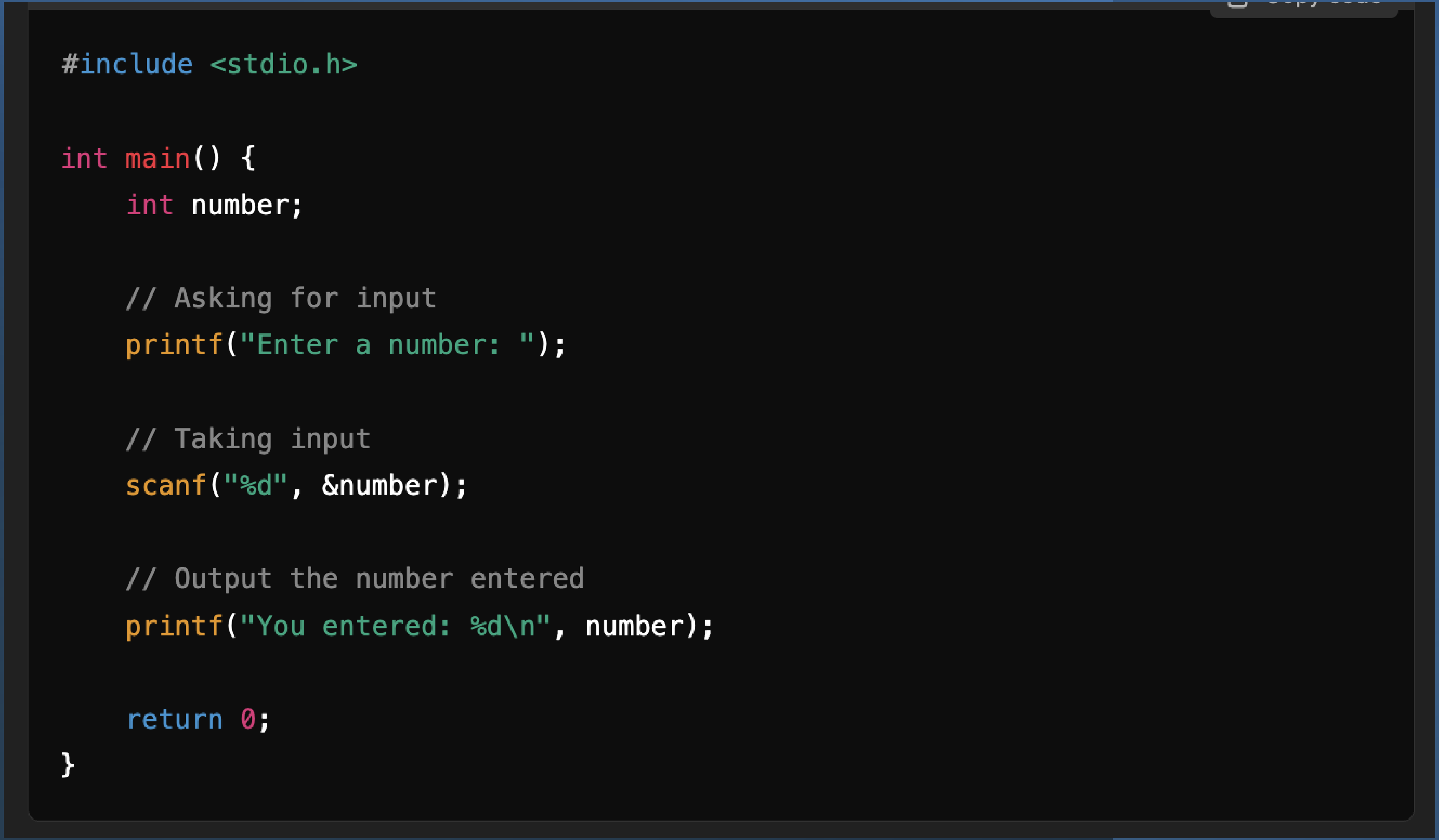
- The program prints "Hello, World!" to the console. It demonstrates the basic structure: `#include`, `main()` function, and the use of `printf()`.



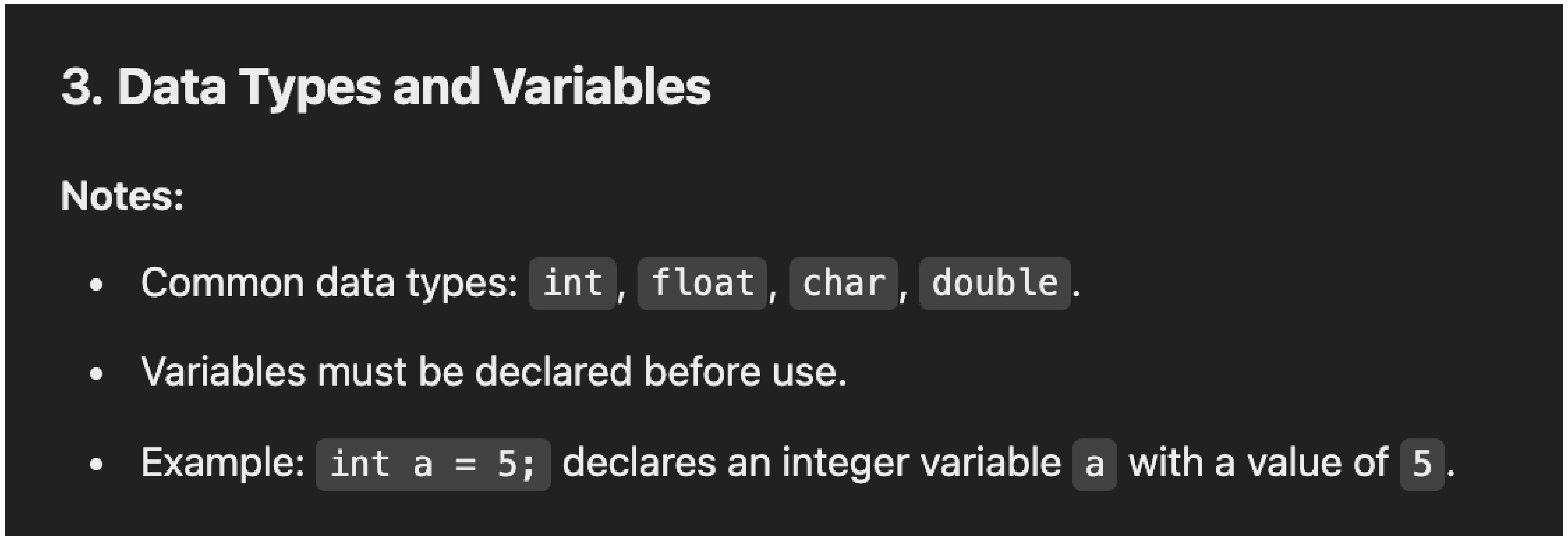
2. Input and Output (`scanf()` and `printf()`)

Notes:

- `printf()` is used to print output.
- `scanf()` is used to take input from the user. You need to use format specifiers like `%d`, `%f`, `%c` to denote the type of data.



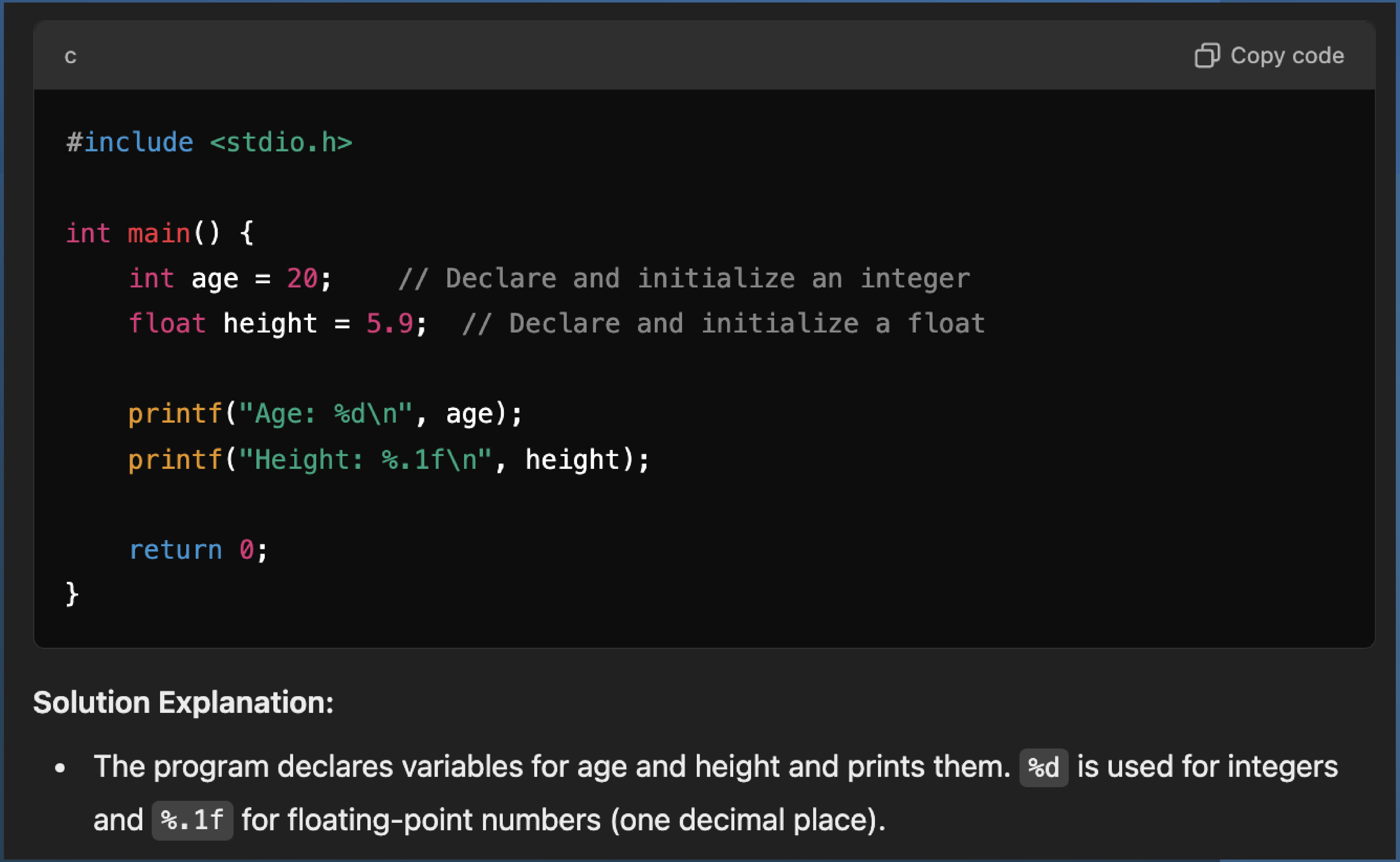
```c
#include <stdio.h>

int main() {
    int number;

    // Asking for input
    printf("Enter a number: ");

    // Taking input
    scanf("%d", &number);

    // Output the number entered
    printf("You entered: %d\n", number);

    return 0;
}
```



3. Data Types and Variables

Notes:

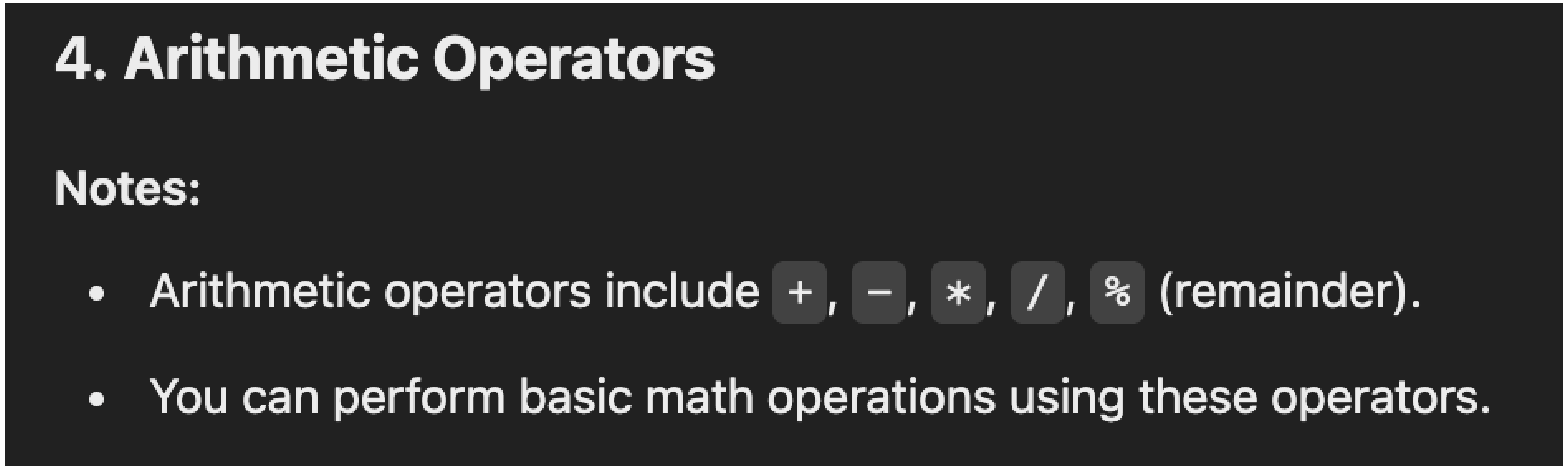
- Common data types: `int`, `float`, `char`, `double`.
- Variables must be declared before use.
- Example: `int a = 5;` declares an integer variable `a` with a value of `5`.



```c
#include <stdio.h>

int main() {
    int age = 20;    // Declare and initialize an integer
    float height = 5.9;  // Declare and initialize a float

    printf("Age: %d\n", age);
    printf("Height: %.1f\n", height);

    return 0;
}
```

Solution Explanation:

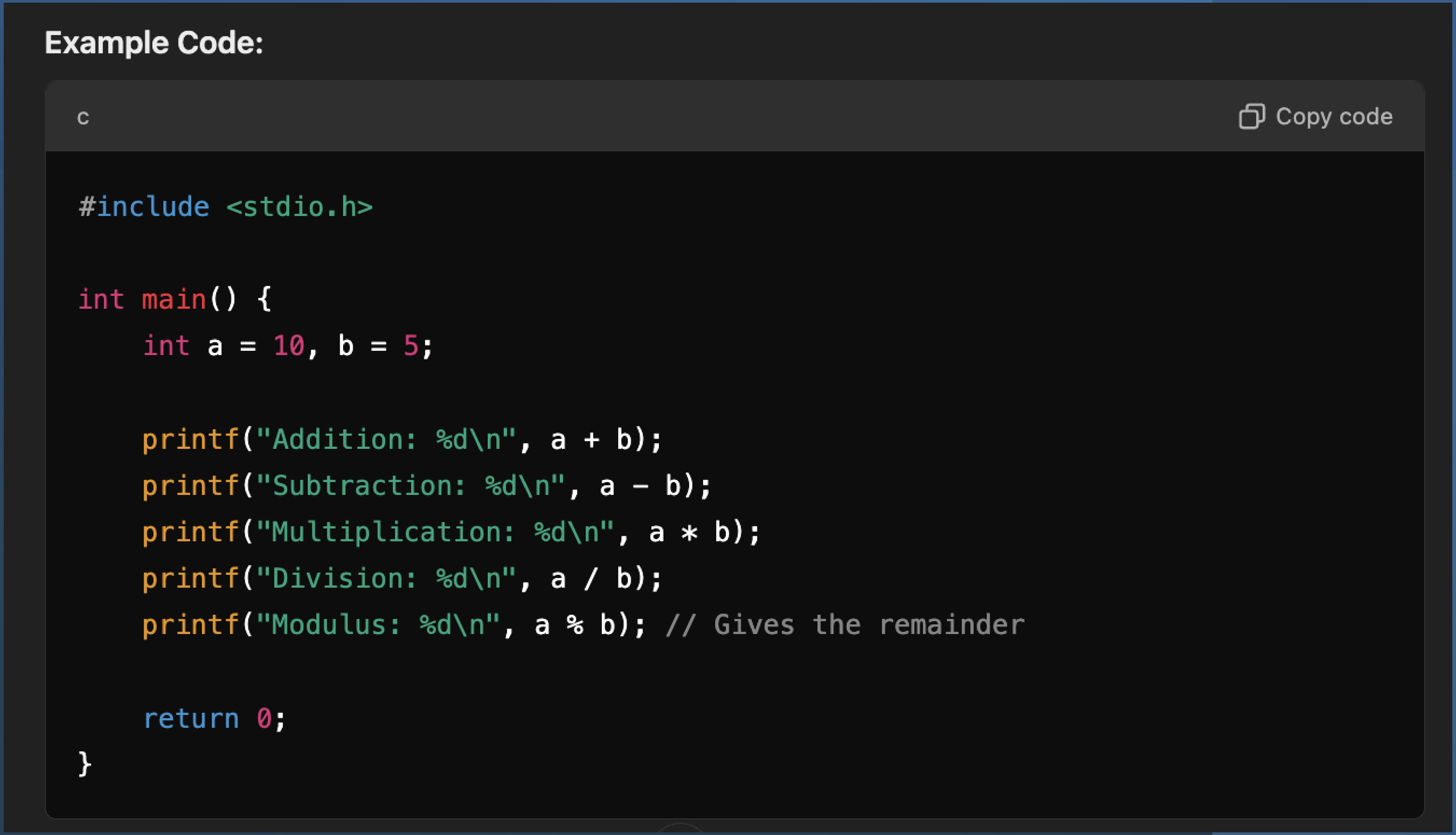
- The program declares variables for age and height and prints them. `%d` is used for integers and `%.1f` for floating-point numbers (one decimal place).



4. Arithmetic Operators

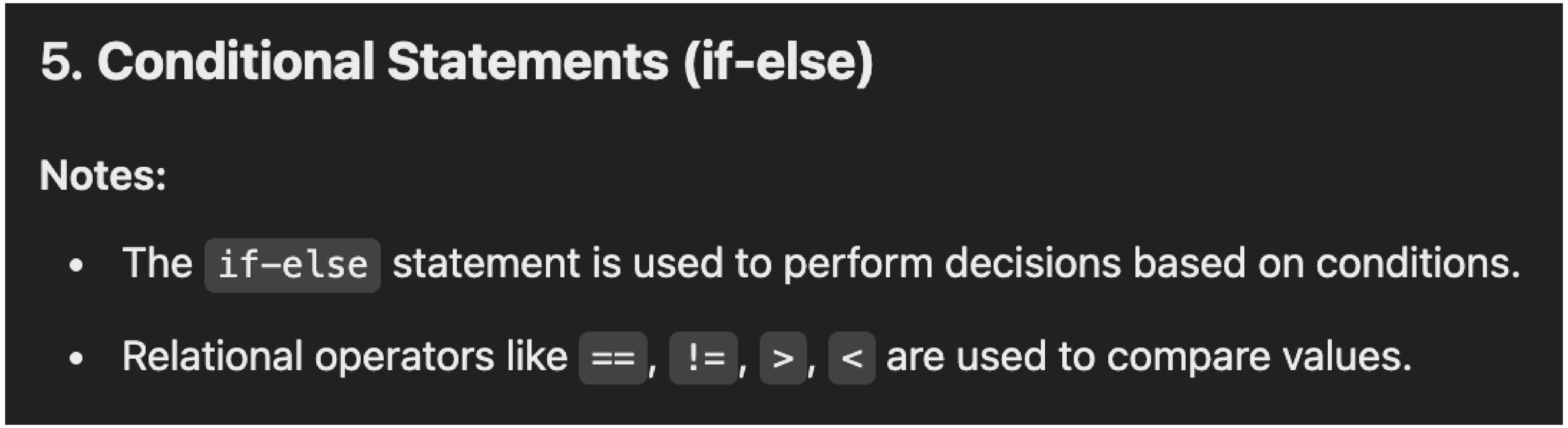
Notes:

- Arithmetic operators include `+`, `-`, `*`, `/`, `%` (remainder).
- You can perform basic math operations using these operators.



Example Code:

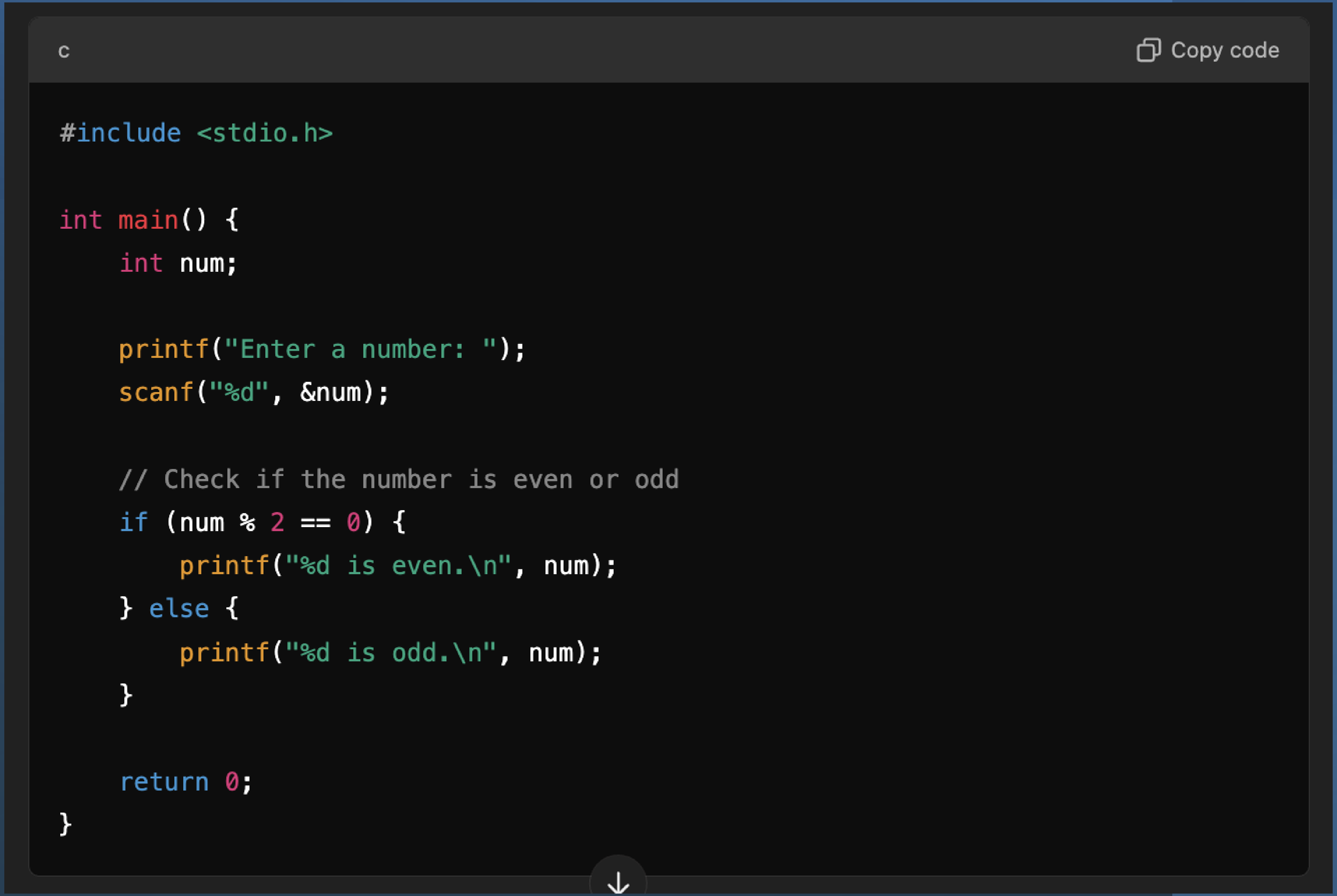
```c
#include <stdio.h>

int main() {
    int a = 10, b = 5;

    printf("Addition: %d\n", a + b);
    printf("Subtraction: %d\n", a - b);
    printf("Multiplication: %d\n", a * b);
    printf("Division: %d\n", a / b);
    printf("Modulus: %d\n", a % b); // Gives the remainder

    return 0;
}
```



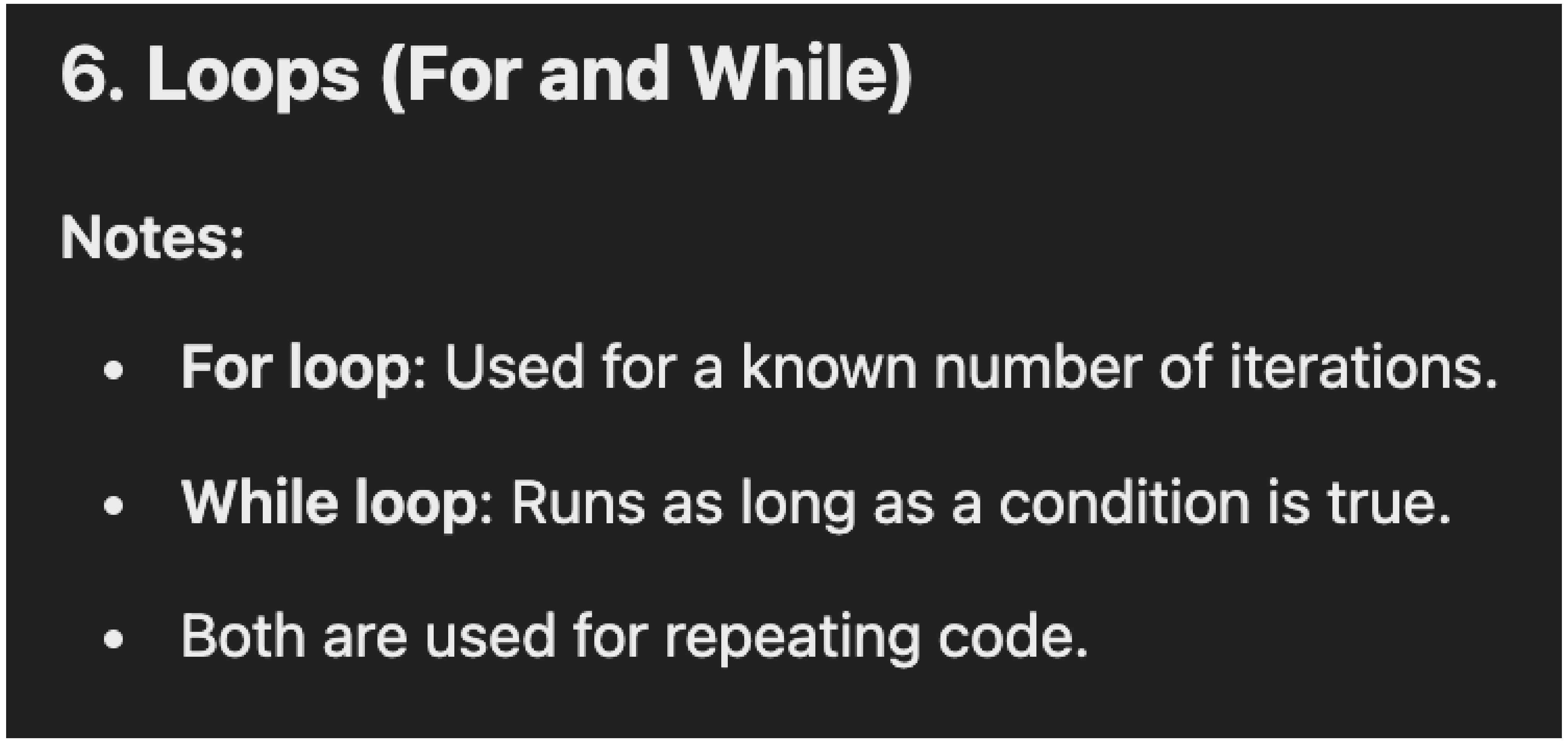
5. Conditional Statements (if-else)

Notes:

- The `if-else` statement is used to perform decisions based on conditions.
- Relational operators like `==`, `!=`, `>`, `<` are used to compare values.



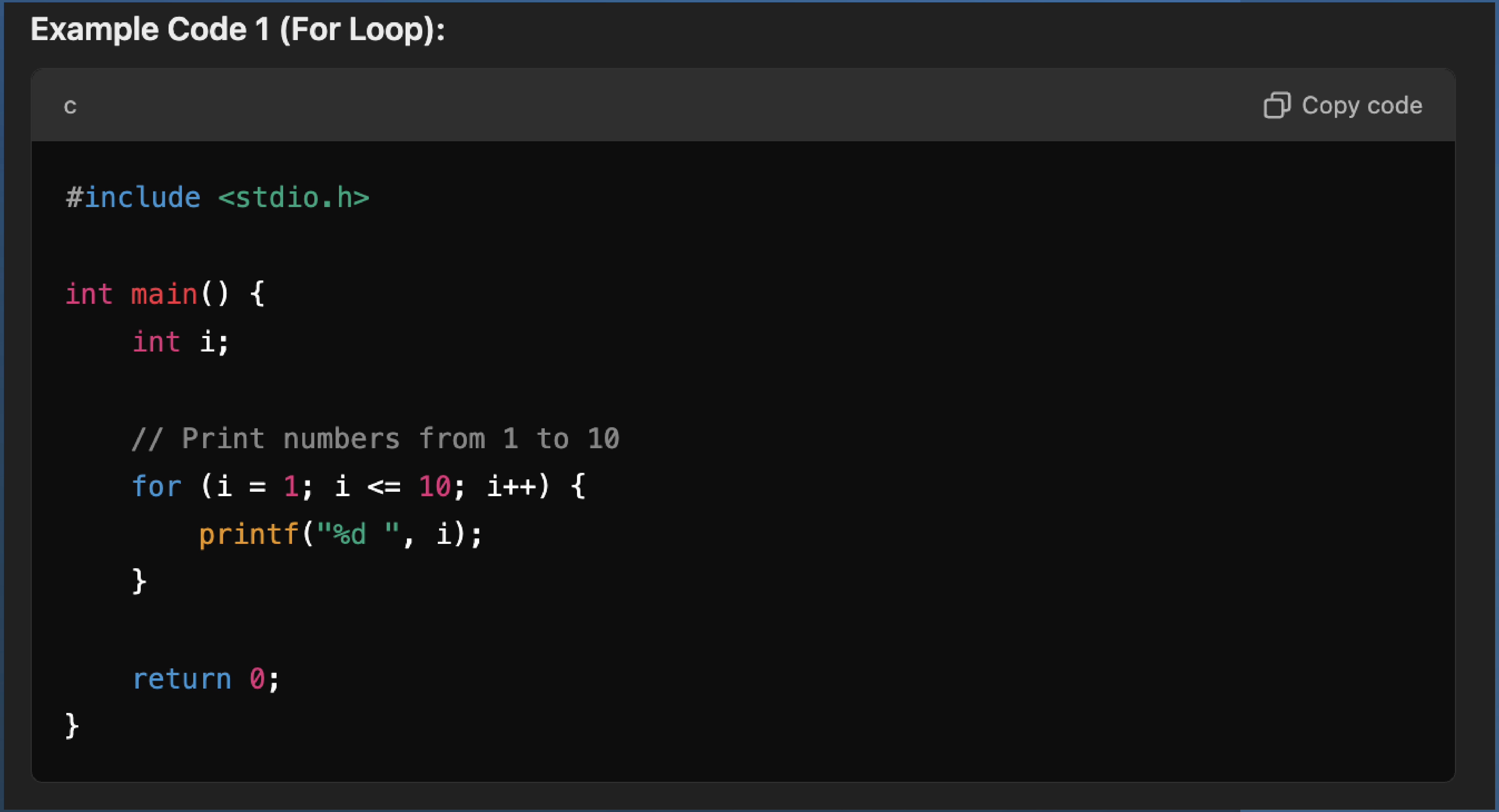
```c
#include <stdio.h>

int main() {
    int num;

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

    // Check if the number is even or odd
    if (num % 2 == 0) {
        printf("%d is even.\n", num);
    } else {
        printf("%d is odd.\n", num);
    }

    return 0;
}
```



6. Loops (For and While)

Notes:

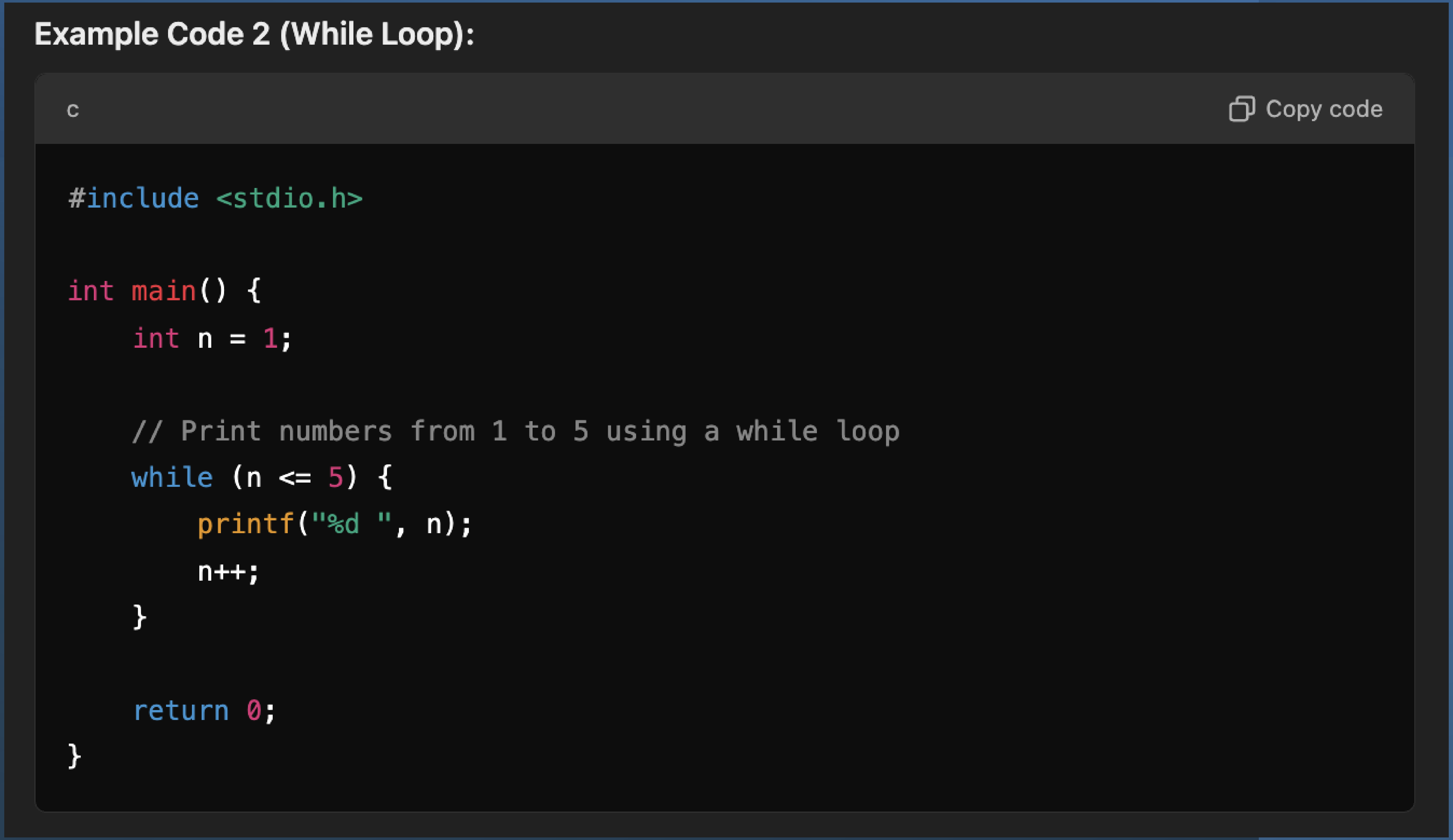
- **For loop**: Used for a known number of iterations.
- **While loop**: Runs as long as a condition is true.
- Both are used for repeating code.


Example Code 1 (For Loop):

c

Copy code

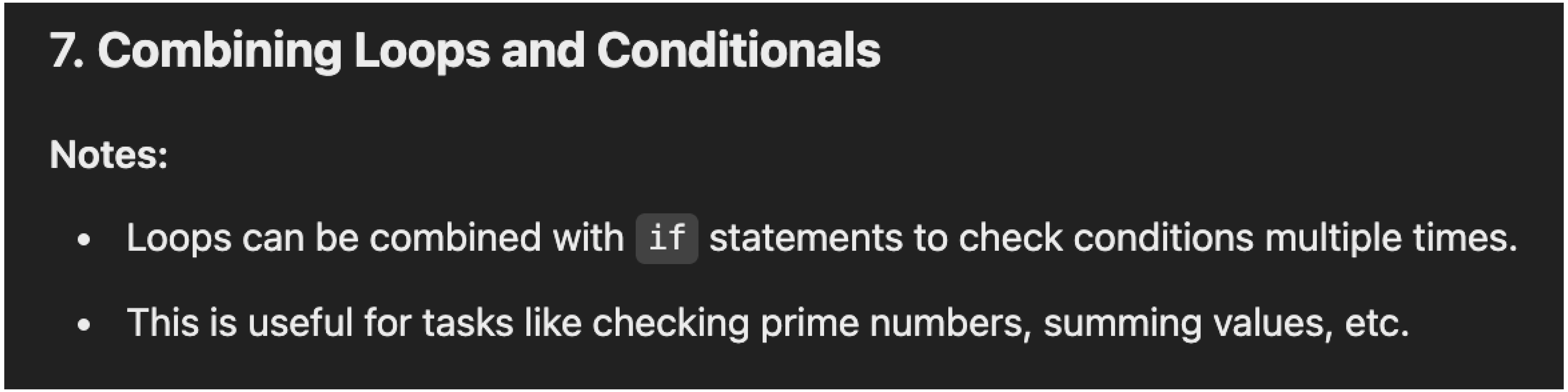
```c
#include <stdio.h>

int main() {
    int i;

    // Print numbers from 1 to 10
    for (i = 1; i <= 10; i++) {
        printf("%d ", i);
    }

    return 0;
}
```



Example Code 2 (While Loop):

c

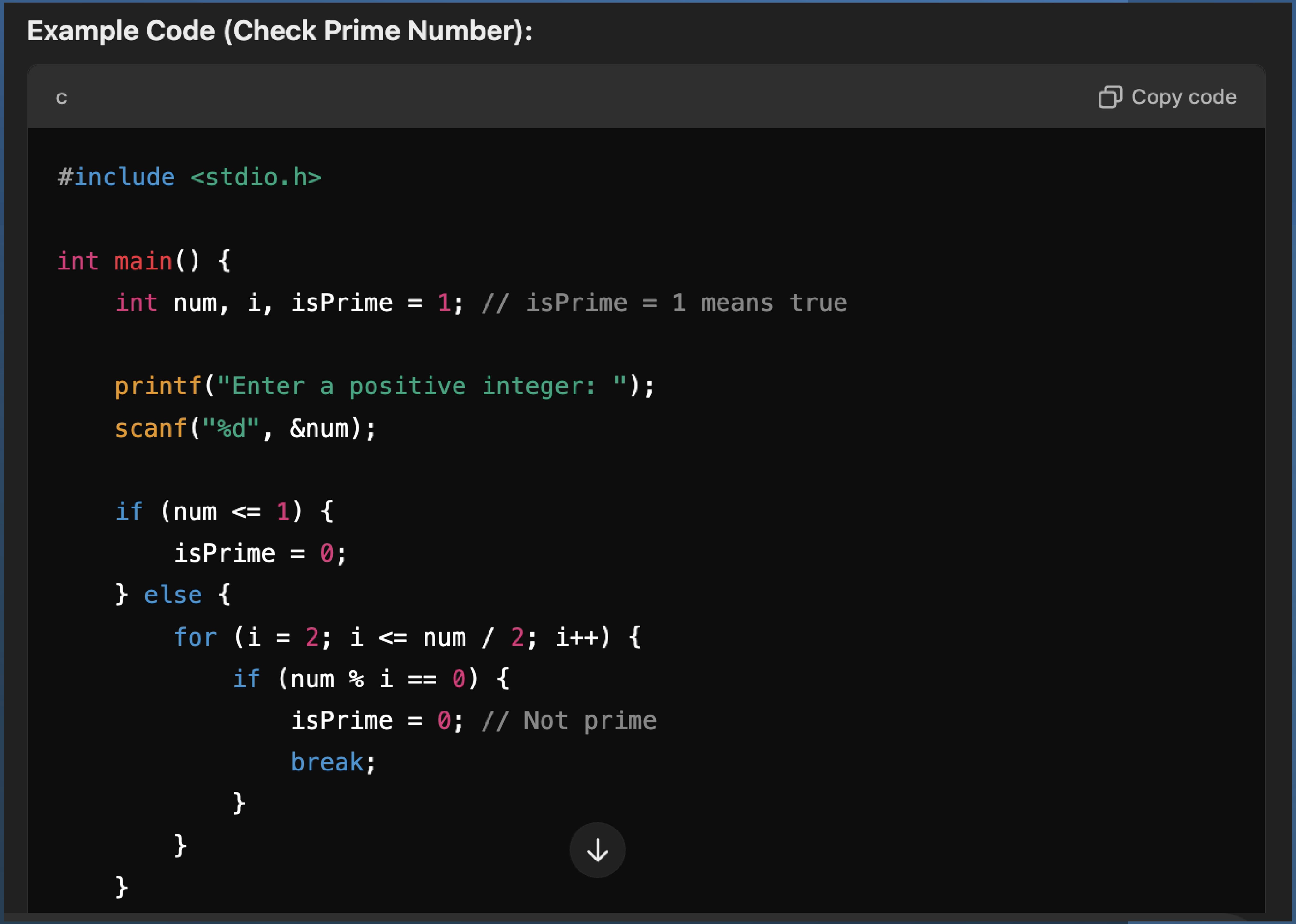
```c
#include <stdio.h>

int main() {
    int n = 1;

    // Print numbers from 1 to 5 using a while loop
    while (n <= 5) {
        printf("%d ", n);
        n++;
    }

    return 0;
}
```



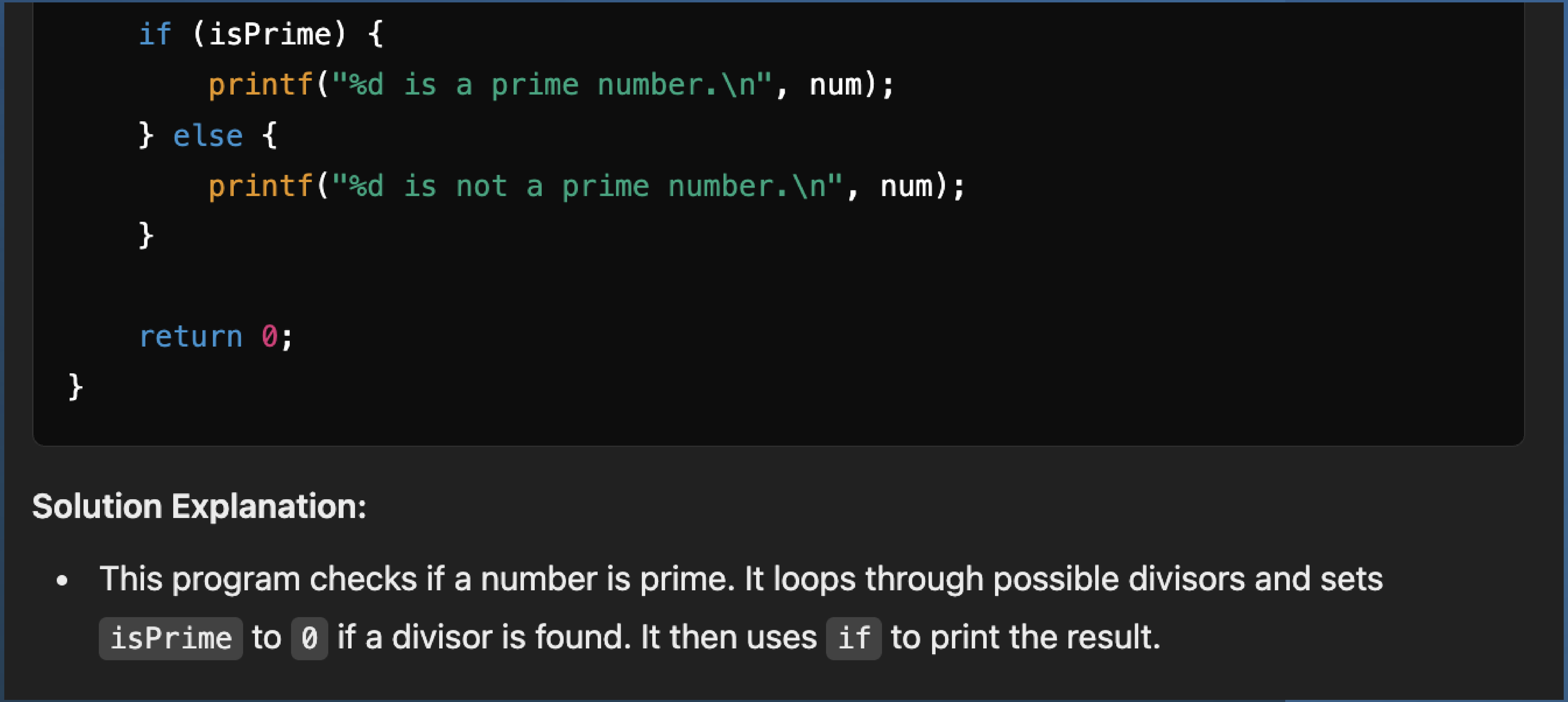
7. Combining Loops and Conditionals

Notes:

- Loops can be combined with `if` statements to check conditions multiple times.
- This is useful for tasks like checking prime numbers, summing values, etc.


Example Code (Check Prime Number):

```c
#include <stdio.h>

int main() {
    int num, i, isPrime = 1; // isPrime = 1 means true

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

    if (num <= 1) {
        isPrime = 0;
    } else {
        for (i = 2; i <= num / 2; i++) {
            if (num % i == 0) {
                isPrime = 0; // Not prime
                break;
            }
        }
    }
```



```c
    if (isPrime) {
        printf("%d is a prime number.\n", num);
    } else {
        printf("%d is not a prime number.\n", num);
    }

    return 0;
}
```

Solution Explanation:

- This program checks if a number is prime. It loops through possible divisors and sets `isPrime` to `0` if a divisor is found. It then uses `if` to print the result.

Assignment Questions

1. Write a program that asks the user to input a year and checks if it's a leap year.
2. Write a program to calculate the grade of a student based on their marks. Use conditions:

   Marks >= 90: Grade A

   - 80 <= Marks < 90: Grade B
   - 70 <= Marks < 80: Grade C
   - 60 <= Marks < 70: Grade D
   - Marks < 60: Fail

3. Write a program to find the factorial of a number using a for loop.
4. Write a program to print all prime numbers between 1 and n, where n is entered by the user.
5. Write a program to swap the values of two variables without using a third variable (using arithmetic operations).

Steps to submit the assignments

Step 1: Make a GitHub repo with the name as YourMentor.
Step 2: Inside the repo YourMentor, make a folder with the name as C_BootCamp.
Step 3: Submit the assignment with the day number (eg: Day1_Assignment.pdf).
Step 4: DM me the link of repo.

NOTE:

1. Active members will get the chance to make a real-life project with me.
2. Advantages with Upcoming BootCamps.

THANK YOU