Prime Number Finder - Multiple Language Implementations

This program finds and prints all prime numbers from 2 to n (inclusive).

Algorithm Overview

1. Read an integer `n` from input
2. For each number `i` from 2 to `n`:
    - Check if `i` is prime by testing divisibility from 2 to √i
    - If no divisors are found, the number is prime
3. Print all prime numbers found

C Implementation (Original)

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

int main() {
    int n;
    scanf("%d", &n);

    for (int i = 2; i <= n; i++) {
        int isPrime = 1;
        for (int j = 2; j <= sqrt(i); j++) {
            if (i % j == 0) {
                isPrime = 0;
                break;
            }
        }
        if (isPrime) {
            printf("%d\n", i);
        }
    }
    return 0;
}
```

Compilation: `gcc -o primes primes.c -lm`

C++ Implementation

cpp

```cpp
#include <iostream>
#include <cmath>

int main() {
    int n;
    std::cin >> n;

    for (int i = 2; i <= n; i++) {
        bool isPrime = true;
        for (int j = 2; j <= std::sqrt(i); j++) {
            if (i % j == 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) {
            std::cout << i << std::endl;
        }
    }
    return 0;
}
```

Compilation: `g++ -o primes primes.cpp`

C# Implementation

csharp

```
using System;

class Program {
    static void Main() {
        int n = int.Parse(Console.ReadLine());

        for (int i = 2; i <= n; i++) {
            bool isPrime = true;
            for (int j = 2; j <= Math.Sqrt(i); j++) {
                if (i % j == 0) {
                    isPrime = false;
                    break;
                }
            }
            if (isPrime) {
                Console.WriteLine(i);
            }
        }
    }
}
```

Compilation: `csc primes.cs` or `dotnet run` (if using .NET CLI)

Rust Implementation

```
rust
```

```
use std::io;

fn main() {
    let mut input = String::new();
    io::stdin().read_line(&mut input).expect("Failed to read input");
    let n: i32 = input.trim().parse().expect("Invalid number");

    for i in 2..=n {
        let mut is_prime = true;
        let sqrt_i = (i as f64).sqrt() as i32;

        for j in 2..=sqrt_i {
            if i % j == 0 {
                is_prime = false;
                break;
            }
        }

        if is_prime {
            println!("{}", i);
        }
    }
}
```

Compilation: `rustc primes.rs` or `cargo run` (if using Cargo)

Key Differences Between Languages

Feature	C	C++	C#	Rust
Input/Output	`scanf/printf`	`cin/cout`	`Console` class	`stdin/println!`
Boolean Type	`int` (0/1)	`bool`	`bool`	`bool`
Math Functions	`sqrt()` from `math.h`	`std::sqrt()`	`Math.Sqrt()`	`f64::sqrt()`
Memory Safety	Manual	Manual	Automatic (GC)	Automatic (ownership)
Compilation	`gcc -lm`	`g++`	`csc` / `dotnet`	`rustc` / `cargo`

Sample Usage

Input:

```
10
```

Output:

```
2
3
5
7
```

All implementations produce identical output for the same input.