# Programming with C and C++

CSC-101 (Lecture 6)

**Dr. R. Balasubramanian**

**Professor**
**Department of Computer Science and Engineering**
**Mehta Family School of Data Science and Artificial Intelligence**
**Indian Institute of Technology Roorkee**
**Roorkee 247 667**

bala@cs.iitr.ac.in
https://faculty.iitr.ac.in/cs/bala/

# Conditional Logic

## Introduction to Conditional Logic

- At the core of programming, we often need to make decisions based on certain conditions.
- "If it’s raining, then I’ll take an umbrella."
- "If it’s an exam day, then I’ll study the night before."
- "If my experiment fails, then I have successfully found one way it doesn’t work."

- In C, we can use the "if" statement to write conditional code.
- This allows us to execute specific code only if a certain condition is met.
- Up next: How does an "if" statement look in C? Let’s find out!

## The "if" Statement in C

- Syntax:

```
if (condition) {
// Code to execute if condition is true
}
```

- The condition must evaluate to true or false.
- If true, the code inside the braces is executed.
- If false, the code inside the braces is skipped.
- Example:

```
if (age >= 18) {
printf("You are eligible to vote.\n");
}
```

## A Simple Program Using "if" Statement

```c
#include <stdio.h>

int main() {
int age;
printf("Enter your age: ");
scanf("%d", &age);

if (age >= 18) {
    printf("You are eligible to vote.\n");
}

printf("Thank you for using our program.\n");
return 0;
}
```

- This program checks if the user is eligible to vote based on their age.
- Let's dissect the program to understand how it works.

## Taking User Input

```
int age;
printf("Enter your age: ");
scanf("%d", &age);
```

- Here, we declare an integer variable ‘age’.
- We then prompt the user to enter their age and store the input in ‘age’.

## The Conditional Check

```
if (age >= 18) {
printf("You are eligible to vote.\n");
}
```

- We use the "if" statement to check if the age is 18 or greater.
- If the condition is true, the message is printed.
- If false, this part of the code is skipped.

## Closing the Program

```
printf("Thank you for using our program.\n");
return 0;
```

- We thank the user for using the program.
- ‘return 0;’ signifies successful execution and closes the program.

## Introduction to "if-else" Syntax

- Sometimes we want to make a decision between two possibilities.
- "if-else" structure allows us to define what to do when a condition is true, and what to do if it’s false.

```
if (condition) {
    // Code to run if condition is true
} else {
    // Code to run if condition is false
}
```

## Introduction to "else if" Syntax

- What if we have more than two possibilities?
- The "else if" clause allows us to test multiple conditions in a specific order.
- If one condition is met, the associated code block is executed, and the rest are skipped.

## Introduction to "else if" Syntax

```
if (condition1) {
// Code to run if condition1 is true
} else if (condition2) {
// Code to run if condition2 is true
} else {
// Code to run if no conditions are true
}
```

- Here, "condition1" is tested first. If true, its code block is executed.
- If "condition1" is false, "condition2" is tested. If true, its code block is executed.
- If neither "condition1" nor "condition2" are true, the code in the "else" block is executed.
- You can have as many "else if" clauses as needed, and they are checked in order.

## Solving a Quadratic Equation: Introduction

**Task:** Solve a quadratic equation $ax^2 + bx + c = 0$ by calculating its roots.

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

int main() {
  float a, b, c, discriminant, root1, root2;
  printf("Enter coefficients a, b, and c: ");
  scanf("%f %f %f", &a, &b, &c);
  discriminant = b * b - 4 * a * c;
```

- We start by declaring the coefficients and roots as floating-point variables.
- The user is prompted to enter the values for $a$, $b$, and $c$.
- We calculate the discriminant to determine the nature of the roots.

## Solving a Quadratic Equation: Roots Calculation

```
if (discriminant > 0) {
  root1 = (-b + sqrt(discriminant)) / (2 * a)
     ;
  root2 = (-b - sqrt(discriminant)) / (2 * a)
     ;
  printf("Two real roots: %f and %f\n", root1
     , root2);
} else if (discriminant == 0) {
  root1 = -b / (2 * a);
  printf("One real root: %f\n", root1);
} else {
  printf("No real roots.\n");
}
return 0;
}
```

- We use the "if-else" structure to determine the correct scenario and calculate the roots accordingly.

## Input and Discriminant

```
float a, b, c, discriminant;
printf("Enter coefficients a, b, and c: ");
scanf("%f %f %f", &a, &b, &c);
discriminant = b * b - 4 * a * c;
```

- This part captures the coefficients of the quadratic equation.
- The discriminant determines the nature of the roots.

## Using if-else to Find Roots

```
if (discriminant > 0) {
// Find two real roots
} else if (discriminant == 0) {
// Find one real root
} else {
// No real roots
}
```

- The program uses ‘if-else’ statements to decide how to calculate the roots based on the discriminant.

## Identifying the Nature of a Triangle

**Task:** Identify whether a triangle is equilateral, isosceles, scalene, or not valid based on its side lengths.

```
#include <stdio.h>

int main() {
    float side1, side2, side3;
    printf("Enter the lengths of the three
       sides of the triangle: ");
    scanf("%f %f %f", &side1, &side2, &side3)
       ;
```

- User inputs the lengths of the three sides of the triangle.
- We will use "if-else" statements to determine the type of triangle.

## Classification

```
if (side1 + side2 <= side3 || side1 + side3
    <= side2 || side2 + side3 <= side1) {
printf("Not a valid triangle.\n");
} else if (side1 == side2 && side2 == side3)
    {
printf("Equilateral triangle.\n");
} else if (side1 == side2 || side2 == side3
    || side1 == side3) {
printf("Isosceles triangle.\n");
} else {
printf("Scalene triangle.\n");
}
return 0;
}
```

- The program first checks if the sides form a valid triangle using the triangle inequality theorem. Then, it classifies the triangle based on the equality of its sides.

## Input and Basic Check

```
float side1, side2, side3;
printf("Enter the lengths of the three sides
   of the triangle: ");
scanf("%f %f %f", &side1, &side2, &side3);
```

- This part captures the sides of the triangle.
- It lays the groundwork for identifying the type of triangle.

## Using if-else to Determine Triangle Type

```
if (side1 + side2 <= side3 || side1 + side3
    <= side2 || side2 + side3 <= side1) {
// Not a valid triangle
} else if (side1 == side2 && side2 == side3)
    {
// Equilateral triangle
} else if (side1 == side2 || side2 == side3
    || side1 == side3) {
// Isosceles triangle
} else {
// Scalene triangle
}
```

- The program first checks if the sides form a valid triangle.
- Then, it classifies the triangle based on the equality of its sides.

## Grading System: Taking User Input

```
#include <stdio.h>

int main() {
    int marks;
    printf("Enter the total marks (out of
       100): ");
    scanf("%d", &marks);
    // The grading logic will go here
    return 0;
}
```

- The user is asked to enter their total marks, which must be an integer.
- This value will be used to determine the grade.

## Grading System: Implementing the Grading Logic

```
if (marks >= 90) {
    printf("Grade: A\n");
} else if (marks >= 80) {
    printf("Grade: B\n");
} else if (marks >= 70) {
    printf("Grade: C\n");
} else if (marks >= 60) {
    printf("Grade: D\n");
} else {
    printf("Grade: F\n");
}
```

- Uses a series of "if-else" statements to determine the grade.
- Each condition checks for a specific range of marks, and the corresponding grade is printed.

## Input and Basic Validation

```
int marks;
printf("Enter the total marks (out of 100):
    ");
scanf("%d", &marks);
```

- Asks the user to enter their total marks.
- You might add a validation check to ensure the marks are between 0 and 100.

## Using if-else to Determine the Grade

```
if (marks >= 90) {
// Grade A
} else if (marks >= 80) {
// Grade B
} else if (marks >= 70) {
// Grade C
} else if (marks >= 60) {
// Grade D
} else {
// Grade F
}
```

- The series of "if-else" statements assigns a grade based on the marks.
- This demonstrates how multiple conditions can be checked in sequence.

## If-Then-Else Exercises

**Problem 1: Leap Year Determination**

- Write a program to determine if a given year is a leap year.
- A leap year is divisible by 4, but years divisible by 100 are not leap years, unless they are also divisible by 400.
- Prompt the user to input the year.

**Problem 2: Sorting Three Numbers**

- Write a program that accepts three numbers and sorts them in ascending order without using array or loop constructs.
- Consider how you would handle cases where some or all of the numbers are equal.
- Prompt the user to input the three numbers.

## Questions to Ponder

1. How does the structure of an "if-else" statement differ from using multiple "if" statements? When would you prefer one over the other?
2. In a series of "if-else if" clauses, what would happen if two or more conditions are true? How does the order of conditions affect the outcome?

## Thank You and Keep Coding!

"The simplicity of an ‘if’ belies the complexity of choices." - Anonymous

Keep practicing, stay curious, and never hesitate to ask questions!