CSE 06131223 ♦ CSE 06131224

Structured Programming

Lecture 8

Decision Making and Branching in C (1)

Prepared by

Md. Mijanur Rahman, Prof. Dr.
Dept. of Computer Science and Engineering
Jatiya Kabi Kazi Nazrul Islam University, Bangladesh
www.mijanrahman.com

Contents

DECISION MAKING AND BRANCHING IN C

Conditional Control Structures

- **Control Structures** are statements that change the flow of a program to a different code segment based on certain conditions.
- The control structures are categorized into three major Conditional types; they are:
  1. Decision making and branching statements
     a) Selection statements
     b) Jump statements
  2. Decision making and looping (Iteration)

Conditional Control Structures

- **Conditional Control Structures statements in C:**

1. Selection Statements:

- If statement
- If Else Statement
- Else If statement
- Nested If statement
- Switch statement

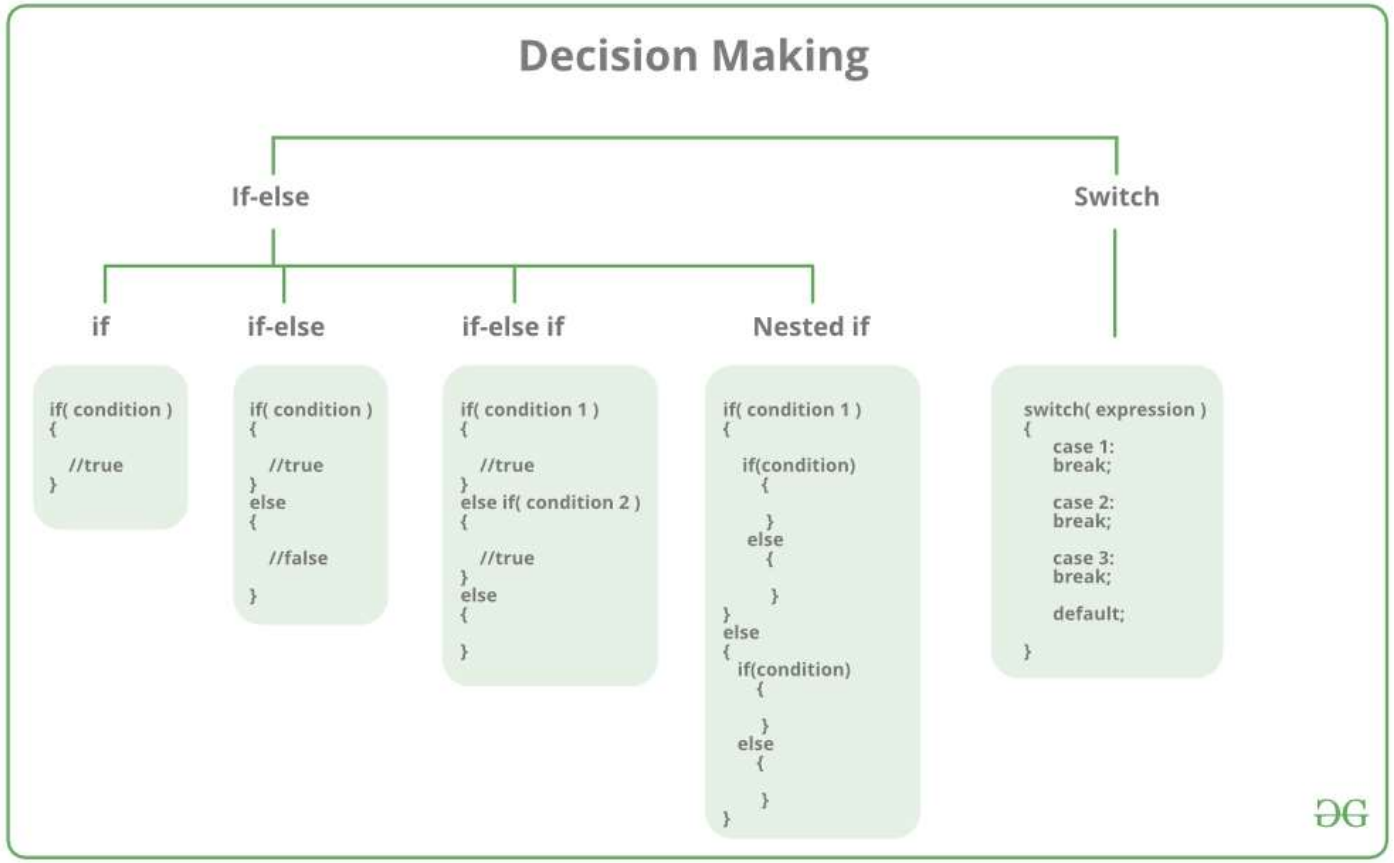
2. Iteration Statements:

- For loop
- While loop
- do while loop

3. Jump Statements:

- return
- goto
- exit()
- break
- continue

Selection Statements in C



If statement in C

- if statement is the most simple decision-making statement. It is used to decide whether a certain statement or block of statements will be executed or not; i.e., if a certain condition is true then a block of statement is executed otherwise not.

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

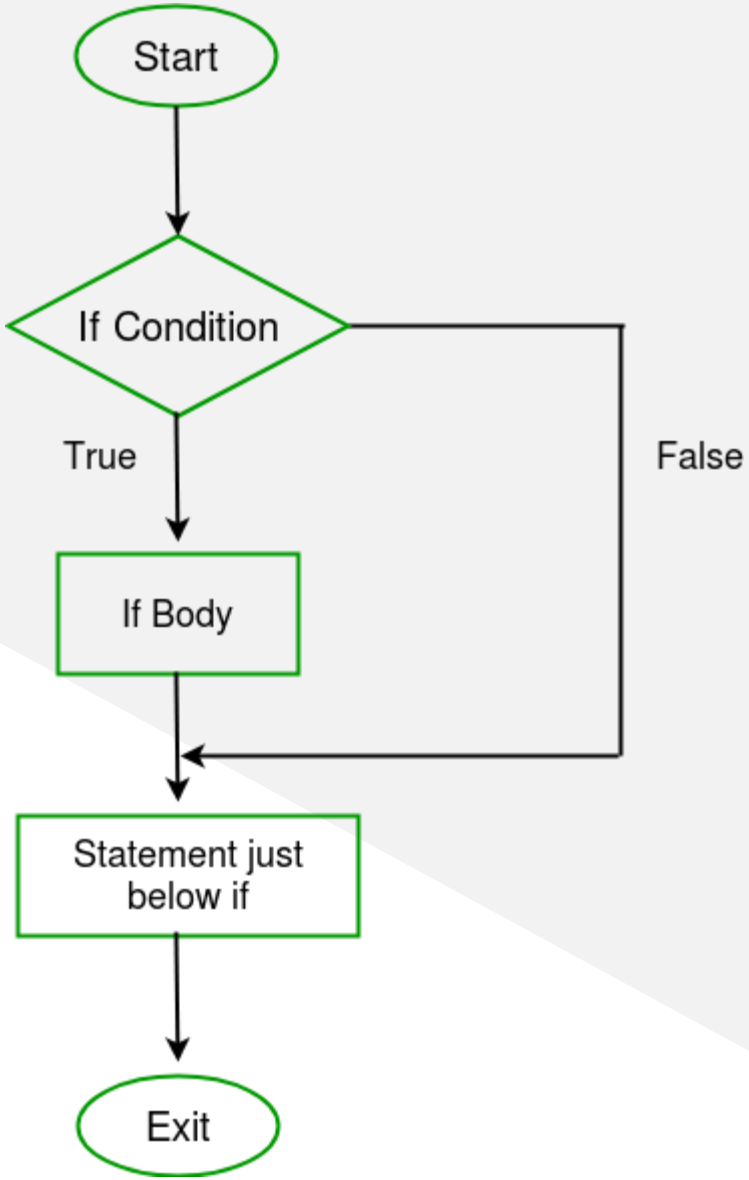
- Here, the **condition** after evaluation will be either true or false. C if statement accepts boolean values – if the value is true then it will execute the block of statements below it otherwise not. If we do not provide the curly braces ‘{‘ and ‘}’ after if(condition) then by default if statement will consider the first immediately below statement to be inside its block.

If Statement in C

- Example:

```
if(condition)
   statement1;
   statement2;

// Here if the condition is true, if block
// will consider only statement1 to be inside
// its block.
```



Flowchart of IF statement

If Statement in C

- **Example:**

```c
// C program to illustrate If statement
#include <stdio.h>

int main() {
    int i = 10;

    if (i > 15)
    {
        printf("10 is less than 15");
    }

    printf("I am Not in if");
}
```

If-else Statement in C

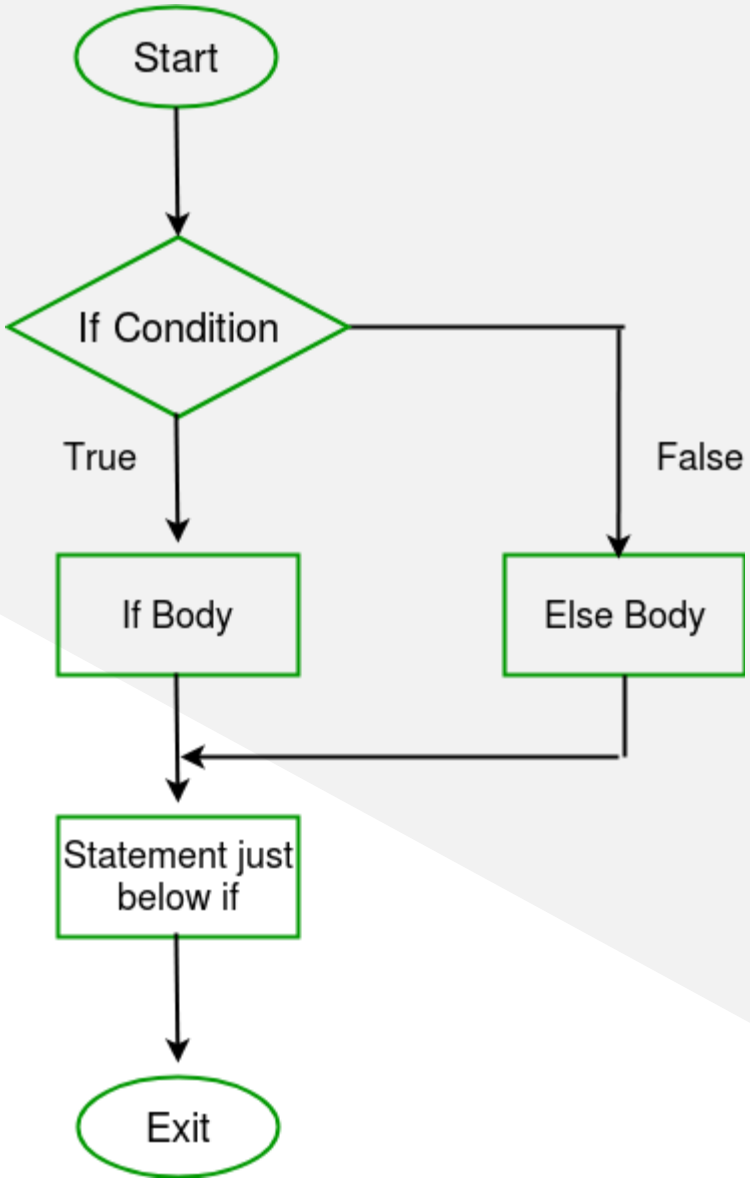
- The *if* statement alone tells us that if a condition is true it will execute a block of statements and if the condition is false it won't. But what if we want to do something else if the condition is false. Here comes the C *else* statement.
- We can use the *else* statement with *if* statement to execute a block of code when the condition is false.

Syntax:

```
if (condition)
{
    // Executes this block if
    // condition is true
}
else
{
    // Executes this block if
    // condition is false
}
```

If-else Statement in C

- **Flowchart of IF-ELSE statement:**



If-else Statement in C

- **Example:**

```
1. // C program to illustrate If statement
2. #include <stdio.h>
3.
4. int main() {
5.     int i = 20;
6.
7.     if (i < 15){
8.
9.         printf("i is smaller than 15");
10.    }
11.    else{
12.
13.        printf("i is greater than 15");
14.    }
15.    return 0;
16. }
```

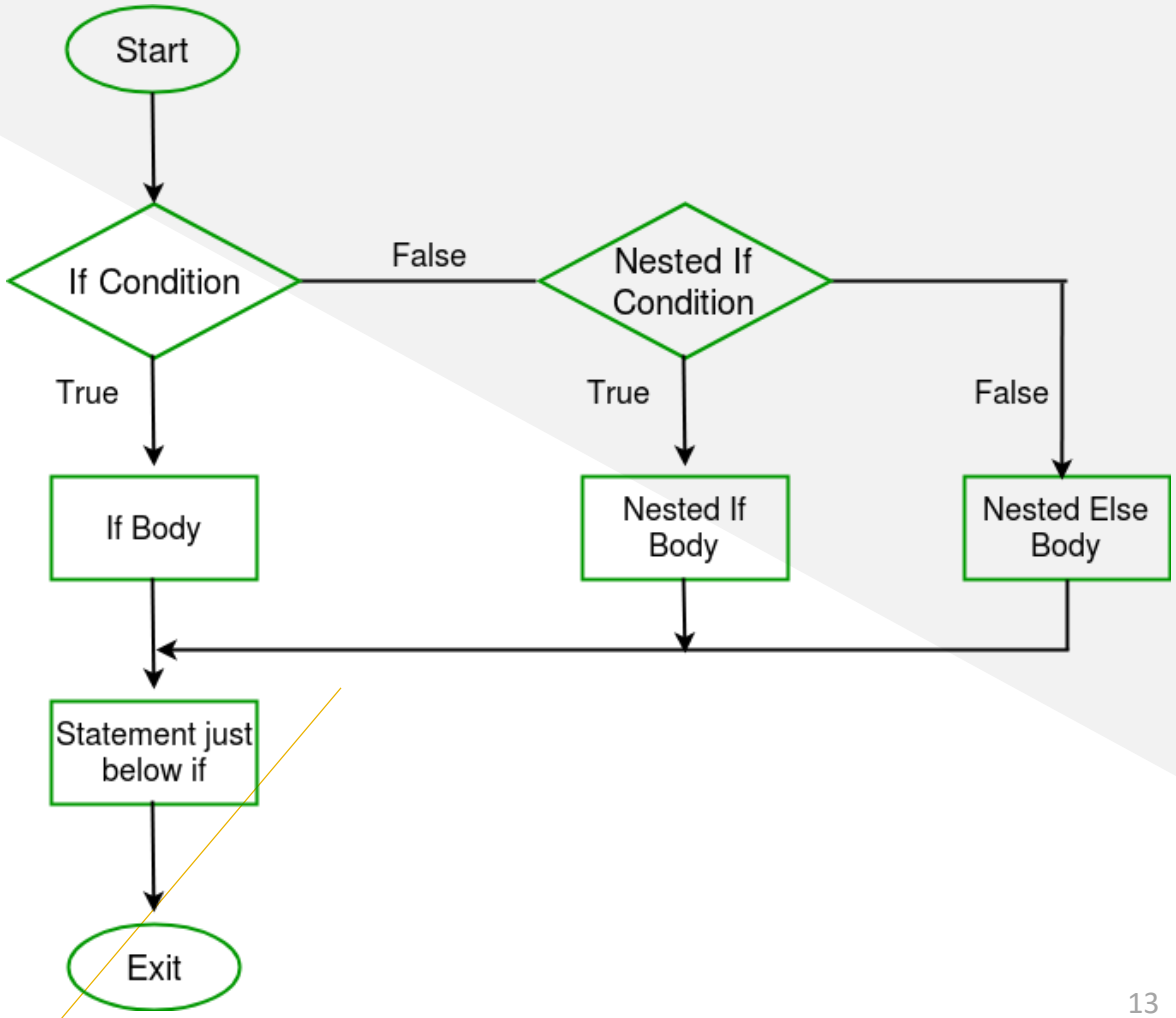
Nested If-else statement in C

- A nested if in C is an if statement that is the target of another if statement. Nested if statements mean an if statement inside another if statement. Yes, both C and C++ allow us to nested if statements within if statements, i.e., we can place an if statement inside another if statement.
- **Syntax:**

```
if (condition1)
{
    // Executes when condition1 is true
    if (condition2)
    {
        // Executes when condition2 is true
    }
}
```

Nested If-else statement in C

- **Flowchart of Nested IF-ELSE statement:**



Nested If-else statement in C

- **Example:**

```c
// C program to illustrate nested-if statement
#include <stdio.h>

int main() {
    int i = 10;

    if (i == 10)
    {
        // First if statement
        if (i < 15)
           printf("i is smaller than 15\n");

        // Nested - if statement
        // Will only be executed if statement above
        // is true
        if (i < 12)
            printf("i is smaller than 12 too\n");
        else
            printf("i is greater than 15");
    }

    return 0;
}
```

Nested If-else statement in C

- **Example: Find the Largest Number Among Three Numbers**

```c
#include <stdio.h>

int main() {

  double n1, n2, n3;

  printf("Enter three numbers: ");
  scanf("%lf %lf %lf", &n1, &n2, &n3);

  // outer if statement
  if (n1 >= n2) {

    // inner if...else
    if (n1 >= n3)
      printf("%.2lf is the largest number.", n1);
    else
      printf("%.2lf is the largest number.", n3);
  }
```

```c
  // outer else statement
  else {

    // inner if...else
    if (n2 >= n3)
      printf("%.2lf is the largest number.", n2);
    else
      printf("%.2lf is the largest number.", n3);
  }

  return 0;
}
```

```
Enter three numbers: -4.5
3.9
5.6
5.60 is the largest number.
```

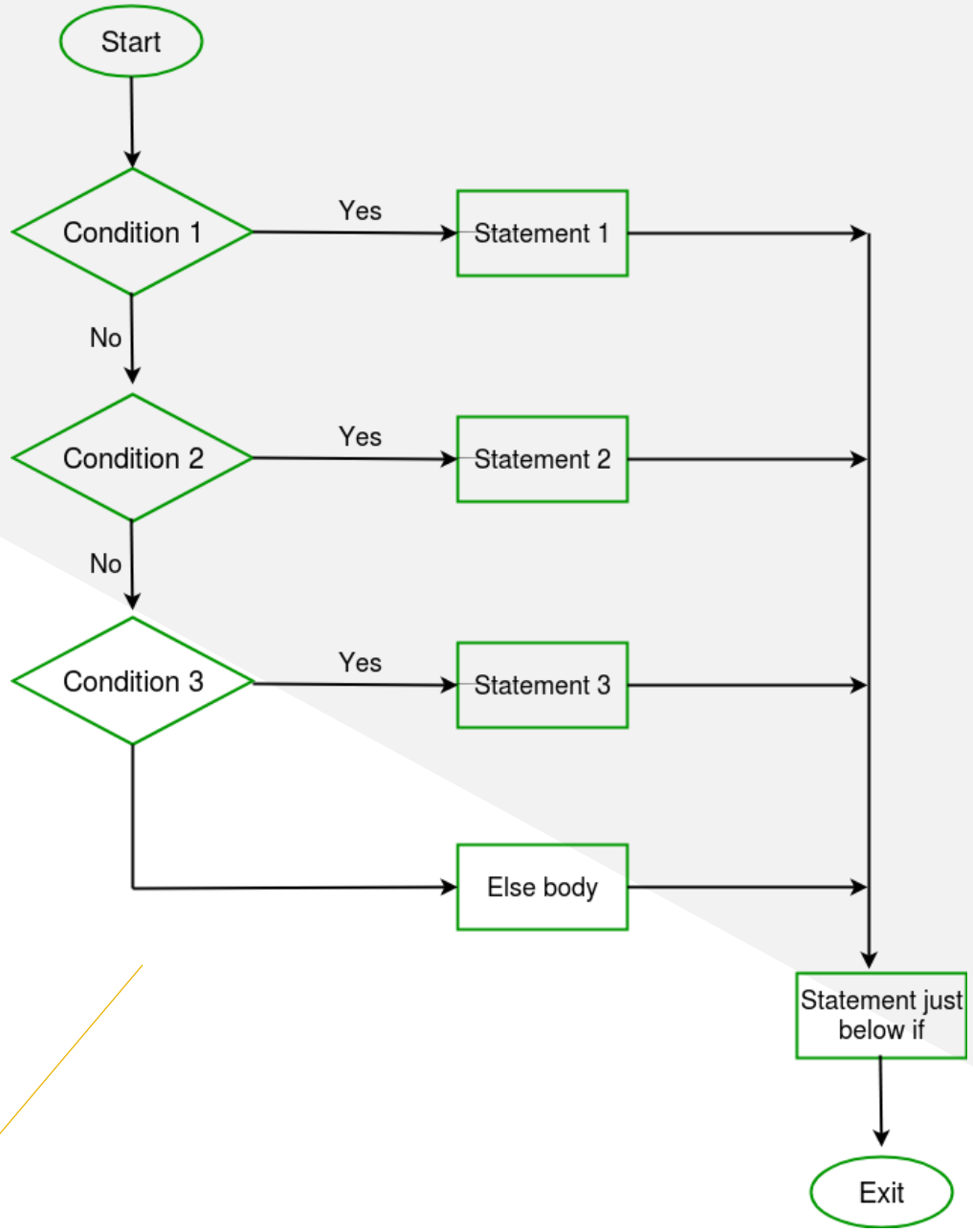
Else-If Ladder Statement

- The **else if statement** is an extension of the "if else" conditional branching statement. When the expression in the "if" condition is "false" another "if else" construct is used to execute a set statements based on an expression.
- This control structure statement also known as **else if ladder** statement.
- **Syntax:**

```
if (condition)
    statement;
else if (condition)
    statement;
.
.
else
    statement;
```

Else-If Ladder Statement

- **Flowchart:**



Else-If Ladder Statement

- **Example:**

```
int main()
{
  int mark;
  printf("Enter mark: ");
  scanf("%d", &mark);
  if (mark >=60 )
  {
    printf("First Division\n");
  }
  else if (mark >=45)
  {
    printf("Second Division\n");
  }
  else if (mark >=33 )
  {
    printf("Third Division\n");
  }
  else
  {
    printf("Fail\n");
  }
  return 0;
}
```

Output:

```
Enter mark:65
First Division
```

THE END