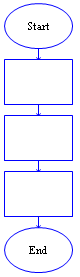
Lab 04: Decisions in C Language

*Familiarization with if statement, To understand the concept of using decisions in C instructions, To understand the concept of performing actions using decisions, To manipulate data using conditions.*

# Introduction

By default, the instructions in a program are executed sequentially. By sequential we mean "in sequence," one-after-the-other. Sequential logic is the easiest to construct and follow. Essentially, you place each statement in the order you want them executed and the program executes them in sequence from the Start statement to the End statement. As you can see by the example program to the right, the arrows linking the statements depict the execution flow. If your program included 20 basic commands, then it would execute those 20 statements in order and then quit.

When you are solving a problem as a programmer, you must determine what statements are needed to create a solution to the problem and the order in which those statements must be executed. Writing the correct statements is one task. Determining where to place those statements in your program is equally important. For example, when you want to get and process data from the user you have to GET the data before you can use it. Switching the order of these statements would produce an invalid program.



**Statement 1**

**Statement 2**

**Statement 3**

Sequential control is the "default" control in the sense that every statement automatically points to the next statement in the flowchart diagram. You do not need to do any extra work to make sequential control happen. However, using sequential control alone will not allow the development of solutions for most real-world problems. Most real world problems include "conditions" that determine what should be done next. For example, "If it is raining, then take your umbrella," requires a decision to be made based on the weather. The "condition" (i.e., the current weather) determines whether the action should be executed or not executed. This is called "selection control".

**Statement 1**

**Statement 2**

**Statement 3**

Hence, in serious programming situations, seldom do we want the instructions to be executed sequentially. Many a times, we want a set of instructions to be executed in one situation, and an entirely different set of instructions to be executed in another situation. This kind of situation is dealt in C programs using a decision control instruction. As mentioned earlier, a decision control instruction can be implemented in C using the

1. if statement
2. if-else statement
3. Conditional operators (Ternary Operator)

Figure 1 Sequential statements

# The IF statement

The ability to control the flow of your program, letting it make decisions on what code to execute, is valuable to the programmer. The ***if*** statement allows you to control if a program enters a section of code or not based on whether a given condition is true or false. One of the important functions of the ***if*** statement is that it allows the program to select an action based upon the user's input. For example, by using an ***if*** statement to check a user-entered password, your program can decide whethera user is allowed access to the program.

Without a conditional statement such as the if statement, programs would run almost the exact same way every time, always following the same sequence of function calls. If statements allow the flow of the program to be changed, this leads to more interesting code.

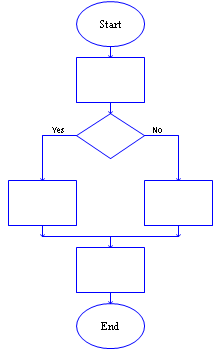
Before discussing the actual structure of if statement, let us examine the meaning of TRUE and FALSE in computer terminology. A true statement is one that evaluates to a nonzero number. A false statement evaluates to zero. When you perform comparison with the relational operators, the operator will return 1 if the comparison is true, or 0 if the comparison is false. For example, the check 0 == 2 evaluates to 0. The check “2

**Statement 1**

**Decision**

**Statement 2a Statement 2b**

**Statement 3**



**Statement 1**

**Decision**

**Statement 2a**

**Statement 2b**

**Statement 3**

== 2” evaluates to a 1. If this confuses you, try to use a printf statement to output the result of those various comparisons

(for example: printf ( "%d", 2 == 1 )

When programming, the aim of the program will often require the checking of one value stored by a variable against another value to determine whether one is larger, smaller, or equal to the other.

There are a number of operators that allow these checks. Here are the relational operators, as they are known,

Figure 2 - Decision control statements

along with examples:

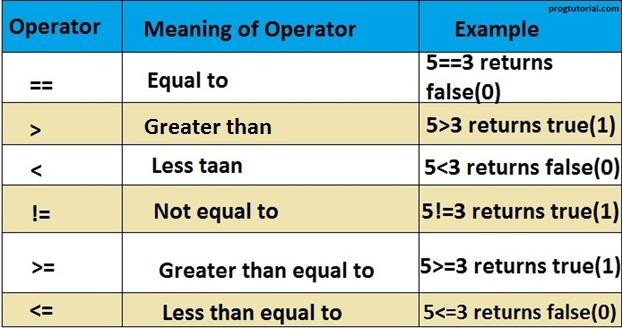


Figure 3 - Relational operators explained

It is highly probable that you have seen these before, probably with slightly different symbols. They should not present any hindrance to understanding. Now that you understand TRUE and FALSE well as the comparison operators, let us look at the actual structure of if statements.

* + 1. **Basic If Syntax**

The structure of an if statement is as follows:

## if ( statement is TRUE )

**{**

**Execute this line of code**

**}**

Here is a simple example that shows the syntax:

## if ( 5 < 10 )

**{**

**printf( "Five is now less than ten, that's a big surprise" );**

**}**

Here, we're just evaluating the statement, "is five less than ten", to see if it is true or not; with any luck, it is! If you want, you can write your own full program including stdio.h and put this in the main function and run it to test.

To have more than one statement execute after an if statement that evaluates to true, use braces, like we did with the body of the main function. Anything inside braces is called a compound statement, or a block. When using if statements, the code that depends on the if statement is called the "body" of the if statement.

## For example:

if ( TRUE )

{

/\* between the braces is the body of the if statement \*/ Execute all statements inside the body

}

It is recommended to always put braces following if statements. If you do this, you never have to remember to put them in when you want more than one statement to be executed, and you make the body of the if statement more visually clear.

# The ELSE statement

Sometimes when the condition in an if statement evaluates to false, it would be nice to execute some code instead of the code executed when the statement evaluates to true. The "else" statement effectively says that whatever code after it (whether a single line or code between brackets) is executed if the if statement is FALSE.

It can look like this:

## if ( TRUE )

**{**

**/\*Execute these statements if TRUE\*/**

**}**

**else**

**{**

**/\*Execute these statements if FALSE\*/**

**}**

**Student Exercises and Tasks:**

**Operators Revisited:**

**Task 01:** Aslam**’s** basic salary is input through the keyboard. His medical allowance is 40% of basic salary, and house rent allowance is 20% of basic salary. Write a program to calculate his gross salary.

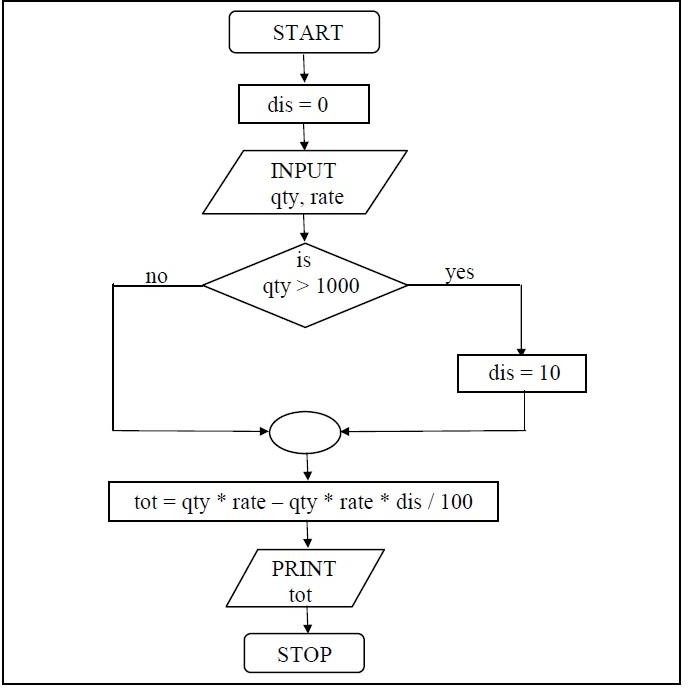
**Task 02:** If a five-digit number is input through the keyboard, write a program to print in reverse order. (Hint: Use the modulus operator ‘%’)

**Task 03:** In a town, the percentage of men is 52. The percentage of total literacy is 48. If total percentage of literate men is 35 of the total population, write a program to find the total number of illiterate men and women if the population of the town is 80,000.

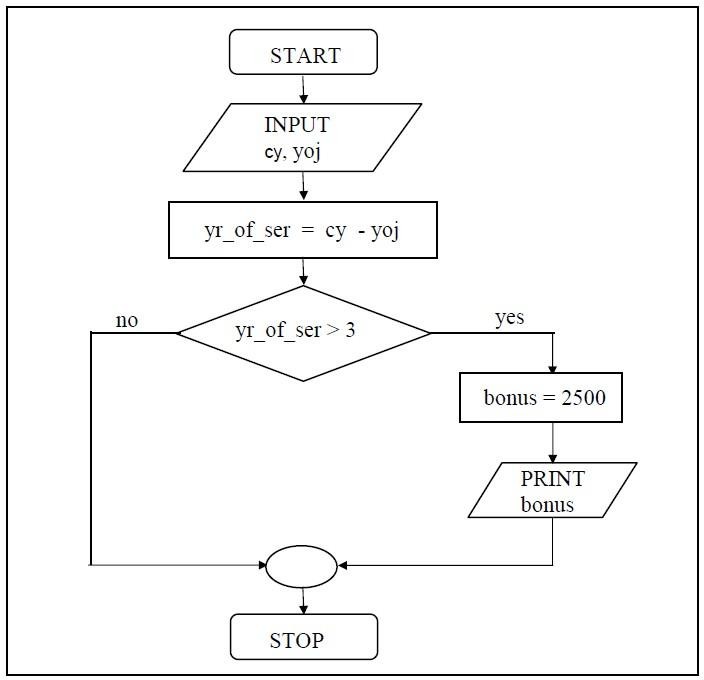
**Task 04:** A cashier has currency notes of denominations 10, 50 and 100. If the amount to be withdrawn is input through the keyboard in hundreds, find the total number of currency notes of each denomination the cashier will have to give to the withdrawer**. *i.e A user input 480 then the answer will be 4 notes of Rs. 100, one note of Rs. 50 and 3 notes of Rs. 10.***

## Conditional Decision Statements:

**Task 05:** While purchasing certain items, a discount of 10% is offered if the quantity purchased is more than1000. If quantity and price per item are input through the keyboard, write a program to calculate the total expenses.



**Task 06:**



**Task 07:** If cost price and selling price of an item is input through the keyboard, write a program to determine whether the seller has made profit or incurred loss. Also determine how much profit he made or loss he incurred.

**Task 08:** Any integer is input through the keyboard. Write a program to find out whether it is an odd number or even number. (Hint: use modulo operator.)