Chapter: 2

**C Control Statements**

General structures of : if, else-if, do, while, for & switch statements

**2.1 The Selection statements**

selection statements make decisions based upon the outcome of some condition. There are three selection statements in C : if, else-if and switch

/\* function definition to swap the values \*/

Every C program consists of one or more modules called functions. One of the functions must be called main. The program will always begin by executing the main function, which may access other functions. Any other function definitions must be defined separately, either *ahead of or after* main .

Arguments :tgvrygbh4yn5um,698kl6

functions

**2.2 The "if" statement**

In its simplest form, the if statement allows your program to conditionally execute a statement. This form of the if is shown here:

^{expression) statement;

The expression may be any valid C expression. If the expression evaluates as true, the statement will be executed. If it does not. the statement is bypassed, and the line of code following the if is executed. In C, an expression is true if it evaluates to any nonzero value II it evaluates to zero, it is false. The statement that follows an if is usually referred to as the target of the if statement.

Commonly, the expression inside the if compares one value with another using a"relational operator Although you will learn about all the relational operators later in this chapter, three are introduced here so that we can create some example programs. A relational operator tests how one value relates to another. For example, to see if one value is greater than another. C uses the > relational operator. The outcome of this comparison is either true or false. For example. 10 > 9 is true, but 9 > 10 is false. Therefore, the following if will cause the message true to be displayed.

if(10 > 9) printf("true");

However, because the expression in the following statement is false, the if does not execute its target statement.

if(5 > 9) printf(“this will not print");

C uses < as its less than operator. For example, 10 < 11 is true. To test for equality, C provides the == operator. (There can be no space between the two equal signs.) Therefore, 10 =»=» 10 is true, but 10 = = 11 is not.

Of course, the expression inside the if may involve variables. For example, the following program tells whether an integer entered from the keyboard is negative or non-negative.

iinclude <stdio.h>

int main(void)

{

int num;

printf1"Enter an integer: "); scan£l"%d", inum);

if(num c C) printf("Number is negative."); if(num > -1' printf("Number is non-negative.");

return 0;

\*

Remember, in C, true is apv nonzero value-and false is zero. Therefore, it is perfectly valid to have an if statement such as the one shown here:

if(count+1) printf("Not Zero");