ARITHMETIC EXPRESSIONS IN C PROGRAMMING - II

Ternary operator

In C the ternary operator is the conditional expression operator.This accepts three operands. Question mark (?) and colon (:) are the two symbols used.

The format of the ternary operator is:

condition ? expression1 : expression2

Example

answer = (x < 1) ? 1 : x * x;
line too long result is returned.

Consider another example which will clarify the usage of the ternary operator :

max_value = (amount > value) ? amount : value;

Modulus operator

The Modulus operator is denoted by the symbol % (percentage sign). The operation performed using this operator is different from division. The remainder left when the first operand is divided by the second operand is returned as the result of the operation.

Program 4.4

```
/* Modulus operator */

#include <stdio.h>
```

```
main()

{

  int x = 25, y=10, z;


      z = x % y;

      printf ("x %% y is %d\n", z);

}
```

Notice that in the printf statement the percentage sign (%) is used twice , this is because printf uses whatever follows the percentage sign to determine how to print the following argument. In the above program since it encounters another percentage sign, it understands that you want to display a percentage symbol and inserts one in the statement.

Logical operators

Logical operators in C are as per table 4.1

Table 4.1

&&	Logical AND
||	Logical OR
!	Unary NOT

Logical operators use true or false properties of expressions to return a true or false. True is represented by a non-zero value and false by zero. Logical operators are different from bitwise AND (&) and OR (|) operations (discussed in Chapter 11).

Consider the following expression :

result = q1 && q2; /* result is true if both q1 and q2 are non-zero */
result = q1 || q2; /* result is true if either q1 or q2 is non-zero */
result = !q1; /* result is true only if q1 is zero */

Precedence of operators

The precedence of an operator gives the order in which operators are applied in expressions: the highest precedence operator is applied first, followed by the next highest, and so on. The *associativity* of an operator gives the order in which expressions involving operators of the same precedence are evaluated. A list of operators, its precedence and associativity can be found in the Appendix.

Datatype conversions

C does implicit datatype conversion when the need arises. When a floating point value is assigned to an integer variable, the decimal portion is truncated. When a value 156.43 is assigned to an integer variable, 156 is stored and the decimal portion is discarded. If an integer 200 is assigned to a floating point variable, the value is converted to 200.000000 and stored.

Another point to remember is, whenever the two operands in an expression are integers, the operation is carried out using the rules of integer arithmetic. Hence any decimal portion resulting from a division operation will be ignored, even when the result is assigned to a floating point variable. This is demonstrated in program 4.6. If one of the operands is a floating point variable then floating point arithmetic will be followed.

Program 4.5

```
/* Implicit datatype conversions in C */

#include <stdio.h>

main()
```

```
{
float f1 = 156.43, result;
int i1, i2 = 200;

   result = i2 / 100;
   printf ("%d divided by 100 results in %f\n", i2,result);

   result = i2 / 100.0;
   printf("%d divided by 100.0 results in %f\n", i2,result);
}
```

Source : http://www.peoi.org/Courses/Coursesen/cprog/frame4.html