Tokens

C tokens are the basic buildings blocks in C language which are constructed together to write a C program. Each and every smallest individual units in a C program are known as C tokens.

In other words token are the basic building blocks of C programming.

C tokens are of six types. They are,

1. Keywords (eg: int, while),
2. Identifiers (eg: main, total),
3. Constants (eg: 10, 20),
4. Strings (eg: “total”, “hello”),
5. Special symbols (eg: (), {}),
6. Operators (eg: +, /,-,*)

Token	Meaning
Keyword	A variable is a meaningful name of data storage location in computer memory. When using a variable you refer to memory address of computer
Constant	Constants are expressions with a fixed value
Identifier	The term identifier is usually used for variable names
String	Sequence of characters
Special Symbol	Symbols other than the Alphabets and Digits and white-spaces
Operators	A symbol that represent a specific mathematical or non mathematical action

C tokens example program:

```
int main()
{
        int x, y, total;
        scanf(“%d%d”,&y,&y);
        total = x + y;
        printf (“Total = %d \n”, total);
}
```

where,

- main – identifier
- {,}, (,) – delimiter

- int – keyword
- x, y, total – identifier

Keywords: Keywords are reserved words by compiler. Keywords are assigned with fixed meaning and they cannot be used as variable name. No header file is needed to include the keywords.

There are 32 keywords

Eg : auto, break, double, int, float

Identifiers: These are the names of variables, functions and arrays, these are the user defined names

Constants: constants in "C" are applicable to the values which not change during the execution of a program.

Variable: It is a data name that may be used to store a data value. A variable may take different values at different times of execution and may be chosen by the programmer in meaningful way. It may consist of letters, digits and underscore character.

Eg: 1) Average
2) Height

Operators

An operator is a symbol that tells the Computer to perform certain mathematical or logical manipulations.

Expression: An expression is a sequence of operands and operators that reduces to single value Eg: 10+25 is an expression whose value is 35 C operators can be classified into a no. of categories.

They include:

1. Arithmetic
2. Relational
3. Logical
4. Assignment
5. Increment and Decrement
6. Conditional
7. Bitwise
8. Special

Arithmetic Operators: C provides all the basic arithmetic operators, they are +, -, *, /, % Integer division truncates any fractional part. Eg: a + b a – b a * b -a * b a / b a % b

Relational Operator: Comparisons can be done with the help of relational operators. The expression containing a relational operator is termed as a relational expression. The value of a relational expression is either one or zero.

1) < (is less than)
2) <= (is less than or equal to)
3) > (is greater than)
4) = (is greater than or equal to)
5) = = (is equal to)
6) != (is not equal to)

Logical Operator: Logical Operators are used when we want to test more than one condition and make decisions. Here the operands can be constants, variables and expressions

C has the following three logical operators.

&& (logical **AND**)
|| (logical **OR**)
! (logical **NOT**)

Eg: 1) if(age>55 && sal<1000)
2) if(number<0 || number>100)

Assignment Operator: Used to assign the result of an expression to a variable. „= „is the assignment operator. In addition C has a set of „short hand" assignment operators of the form

Var = Exp :

Increment And Decrement Operators: C has two very useful operators that are not generally found in other languages. These are the *increment* and *decrement* operator:

++ and --

The operator ++ adds 1 to the operands while – subtracts 1.It takes the following form:

++m; or m++
--m; or m—

Conditional Operator: is used to check a condition and Select a Value depending on the Value of the condition. Variable = (condition)? Value 1 : Value 2:

Bitwise Operator: are used to perform operations at binary level i. e. bitwise. These operators are used for testing the bits, or Shifting them right or left . These operators are not applicable to float or double. Following are the Bitwise operators with their meanings.

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
<<	Shift left
>>	Shift right
~	One's complement

SPECIAL OPERATORS

C supports some special operators such as

- Comma operator
- Size of operator
- Pointer operators(& and *) and
- Member selection operators(. and ->)

THE COMMA OPERATOR

The comma operator can be used to link the related expressions together. A comma-linked list of expressions are evaluated *left* to *right* and the value of *right-most* expression is the value of the combined expression.

Eg: value = (x = 10, y = 5, x + y);

This statement first assigns the value 10 to **x**, then assigns 5 to **y,** and finally assigns 15(i.e, 10+5) to **value**.

The Size of Operator

The size of is a compiler time operator and, when used with an operand, it returns the number of bytes the operand occupies.

Eg: 1) m = **sizeof**(sum);

2) n = **sizeof(long int)**

3) k = **sizeof**(235L)