Learn CODE: Start with C

Article is prepared by –

Sorowar Mahabub, Studies B.Sc in CSE,

Int' Islamic University Chittagong (IIUC)

Cell: 01834756433, 01521564157(whasApp available)

Email : sorowarmahabub1709vip@gmail.com

*Article Reference : Google, DataFlair_blog*

C Syntax Rules

(Learn the ABCs of Programming in C Language)

In this tutorial, we will be discussing the C syntax in detail. If you have gone through our previous tutorials you might have understood the importance of **C language.** We will now focus on the syntax and the program structure in C. We will not be discussing any complex program code at this stage. Hence, this tutorial will help you start with a basic C program in a precise manner. At the end of this tutorial, you will be able to explain each line of code.

So, let's start today's journey with C basic Syntax.
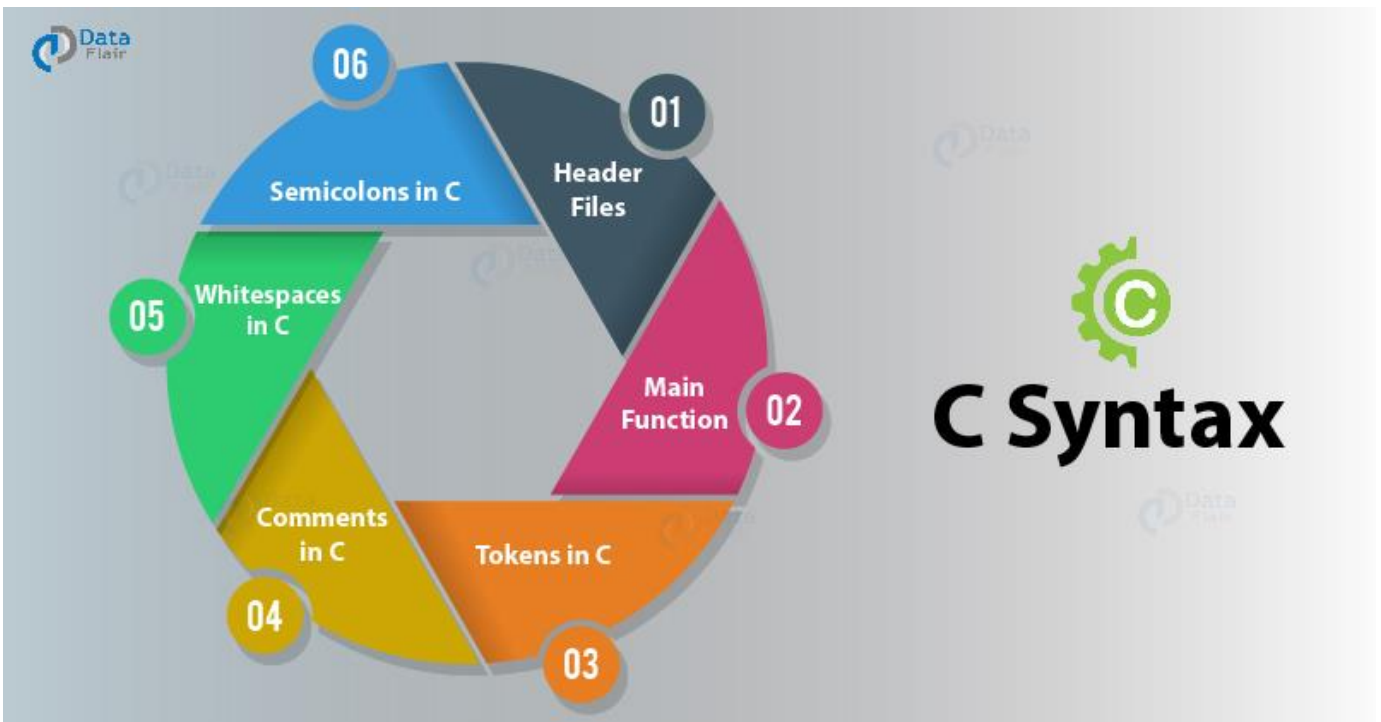
(This Picture Collected from DataFlair)

C Basic Syntax

Syntax basically refers to the protocols to be followed while writing a program. It is very necessary to follow proper syntax while coding to get the desired set of output. The C basic syntax consists of header files, main function, and program code. This is the most fundamental structure in the C program. A C program necessarily consists of the main function because the execution of the program starts from this line. Without the main function, the program execution does not start.

1. #include<stdio.h>
2. int **main**() // main function with integer return type
3. {
4. **printf**("Welcome to Learn CODE: Start with C!\ n"); // print statement to display output on the screen
5. return 0; // Indicates that the main function returns null value
6. }

From the above C syntax,

We understood the placing of these elements. Now, let's discuss each element of the above code in depth.

1. Header Files

In the above code, we have used two header files.

#include <stdio.h>
#include<conio.h>
C has a series of predefined functions embedded in the header files in the C library. We use these functions to perform a specific task with the help of a header file.

What are header files?
Header files contain a bunch of files that help the user to include the predefined functions according to his requirements. You can add header files using the preprocessor directive **#include.**

2. Main Function

When your operating system runs a program in C, it gives the control of the computer to the C program. If a program consists of only one line of code in the main function, then the program can run successfully.

As the program execution starts from this line, it becomes compulsory for every C program to have the main function.

- **clrscr**() – This function is predefined in the header file **<conio.h>**. clrscr() helps to clear the screen.
- **printf**() – This is another function that helps you to display the message on the output screen. This function is defined in **<stdio.h>.** In the above program, it is pretty clear that the printf() function helps you display the output message: "Welcome to DataFlair".
- **getch**() – This function is defined in conio.h and is used to hold the output screen until we hit the next key after we run the program.

This was a line to line description of the simplest code that can be built in C. With this, you might be clear with the flow of a C syntax. We will further discuss more complex programs in the simplest form of our upcoming C tutorials.

3. Tokens in C

A C syntax/program consists of a series of tokens. They can be identifiers, variables, symbols. We can say that they are *the smallest units of a program that convey a specific meaning to the compiler*. There are 5 *types of tokens in the C language*. **They** are :

1. Keywords
2. Identifiers/Variables
3. Constants
4. Punctuators
5. Operators
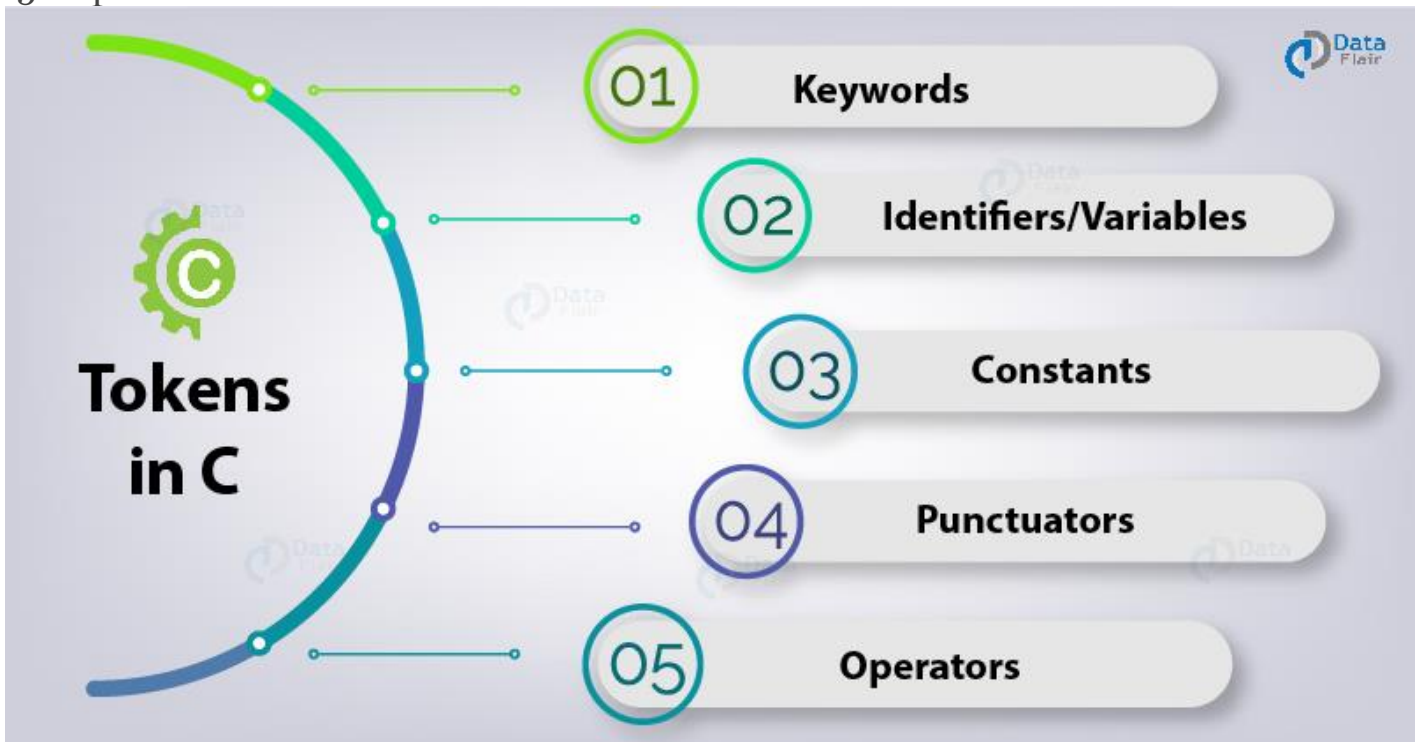
3.1 Keywords in C

Keywords are a set of words used for a special purpose in the program. They are reserved words. Some examples are for, while, switch, break, goto, etc.

```c
1. #include<stdio.h>
2. int main()
3. {
4. for(int i=0;i<3;i++)
5. {
6. printf("Welcome to Learn CODE: Start with C!\n");
7. }
8. return 0;
9. };
```

In this example, we have used a keyword to print a message using the **printf()** *function in C.*

3.2 Variables in C

It is a *data item in the program whose value can be changed* and thus, it is called variable. They are a combination of letters and digits.
Rules for naming identifiers or variables
- No special character can be used to name a variable except underscore(_).
- The variable name cannot start with a digit, it can be a letter or an underscore.
- No keyword can be used as an identifier.

3.3 Constants in C

*Constants are those values that cannot be changed during program executio*n.

3.4 Punctuators in C

Punctuators are used for formatting the C program statements and expressions.
Examples for punctuators are : [ ] { } ( ) , ; : * =

Let us talk about one of the punctuators i.e., semicolons

A semicolon is used to show the ending of a statement. It acts as a separator between two statements. If you miss this statement at the end of any line, the compiler would join the statement with the next line of code and will give a **syntax error**.

3.5 Operators in C

There are various **operators in C** that we use in order to *perform different mathematical operations*. They are:

- Arithmetic Operators : +, -, *, /
- Increment/Decrement Operators : ++,−
- Relational Operators: ==, <, >, <=, >=, !=
- Logical operators : &&, ||, !
- Conditional Operators : ?:

4. Comments in C

A comment is a simple text that makes your code more descriptive and readable. You can add comments in your code just to explain how the code is functioning. Comments aren't a necessary part of a code, you can add them according to your convenience.

There are two types of comments:

1. Single line comments
2. Multi-line comment

For single line comment just put '//' before the comment line

For multi-line comment /*——*/ , the statements of comment should be enclosed within /* and */.

```
1. #include<stdio.h>
2. int main()
3. {
4. printf("Welcome to Learn CODE: Start with C!\n"); // This is a single line comment
5. /* This is a multi-line comment */
6. };
```

5. Whitespaces in C

In a C program, whitespace is generally a blank line. The C programming language uses whitespaces to describe blanks, newline characters, and comments.

Whitespace separates one statement from another statement for the compiler to identify. In this statement,

int data ;

If you don't provide proper whitespace within the statements where it is needed, then the compilation will not be correct and you will not be able to get the desired output.

For example: If you have not provided whitespace between int and data, then it would create a compilation error.

6. Semicolons in C

Semicolons are *provided to terminate the statement*. A Semicolon indicates the start and end of statement execution.
This tells the compiler that the statement is completed. If a semicolon is not placed at the end of a statement, then the program will not compile generating a compilation error message.

In the above code, all the statements within the function terminate with a semicolon. If the semicolon is not provided, it will generate "Compilation Error".

Rules for a Program in C

1. C is a case sensitive language. Most of the program statements are in lower case.
2. All statements must necessarily terminate with a semicolon.
3. Whitespaces should be provided only between variables and keywords.

Summary

We hope you understood the basic syntax of the C programming language with the program structure. Now, you can easily apply tokens, keywords, comments and other essential elements of C language in your program. These elements of the C syntax play a very important role in programming in C and makes you proficient in the C programming language.

Bismillah Hir Rahmanir Rahim

"Let's appear in thy smiling"

Md. Sorowar Mahabub Rabby

01834-756433
01756-043655

Studies B. Sc in CSE(Computer Science & Engineering),
Int' Islamic University Chittagong (IIUC).

Baroiyerhat, Mirsharai, Chittagong.

SOrowar Mahabub   Sorowar Mahabub

sorowarmahabub1709vip@gmail.com