History of C programming Language

- C is a programming language which born at "AT & T's Bell Laboratory" of USA in 1972.
- C was written by Dennis_Ritchie.
- C language was created for a specific purpose i.e designing the UNIX operating system.

Why name C was given to Language:

- Many of C's principles and ideas were derived from the earlier language B. (Ken Thompson was the developer of B Language.)
- BCPL and CPL are the earlier ancestors of B Language
- CPL is common Programming Language. In 1967, BCPL Language (Basic CPL) was created as a scaled down version of CPL
- As many of the **features were derived from "B" Language thats why it was named as "C"**.

Summary of C programming Language History:

Summary –		
1	B Language Developed By	Ken Thompson
2	Operating System Developed in C	UNIX
3	Developed at	AT & T Bell Laboratory
4	Creator of Traditional C	Dennis Ritchie
5	Year	**1972**

C Language Compiler

compiler is translator software which converts HLL program into a object program understandable by a computer.

To compile the program on **MS Window** operating system we use the compile option from the compile menu or ALT+F9 key combination and to execute the program CTR+F9 key combinations. It will report us about errors , if no error is there then we shall see successful compilation message.

C - Program Structure

All the C programs are written into text files with extension ".c", known as **program file** of C, for example *sum.c.*

A C program basically has the following

1. Preprocessor Commands
2. Functions
3. Variables
4. Statements & Expressions
5. Comments

Following is an example of C program for addition of two numbers:

```
/* Program for addition of two numbers*/
#include<stdio.h>
#include<conio.h>
void main()
{
        int n1,n2,R;
        clrscr();
        printf("Enter two numbers:");
        scanf("%d%d",&n1,&n2);
        R=n1+n2;
        printf("\nSum =%d",R);
     getch();
}
```

Let us take a look at the **various parts** of the above program –

Preprocessor Command (Directive):

It is a directions to the compiler to process the line following the symbol # before processing main () function of C program. For example *#include <stdio.h>* line tells a C compiler to include stdio.h file from standard library before processing main () function of C program.

Functions:

are main building blocks of any C Program. Every C Program will have one or more functions and there is one mandatory function which is called *main()* function. The C Programming language provides a set of built-in functions such as *printf(), clrscr(), getch() etc.*

Variables:

are used to hold data such as characters, numbers, strings etc. for data manipulation e.g. n1,n2 R etc.

Statements & Expressions:

Statements are expressions, assignments, function calls, or control flow statements which make up C programs. Expressions combine variables and constants to create new values.

Comments:

comments are used to give additional useful information inside a C Program. Comments can be used to describing function behavior, variable usage, and algorithms etc. Comments are ignored by C compilers. The comments are of two types (1) multiline comments (/*……..*/) (2) single line comments (//….).

A multiline comment can span through multiple lines. All the comments will be put inside /*...*/ symbols.

Single line comment will be put after // symbol

Character Set

The character set in C Language can be grouped into the following categories

1. Letters 2. Digits 3. Special Characters 4. White Spaces

Letters	Digits
Upper Case A to Z	0 to 9
Lower Case a to z	.

Special Characters

,	Comma	&	Ampersand
.	Period	^	Caret
;	Semicolon	*	Asterisk
:	Colon	-	Minus Sign
?	Question Mark	+	Plus Sign
'	Apostrophe	<	Opening Angle (Less than sign)
"	Quotation Marks	>	Closing Angle (Greater than sign)

White Space

1. Blank Space
2. Horizontal Tab
3. Carriage Return
4. New Line
5. Form Feed

C Keywords

C keywords are the words that convey a special meaning to the c compiler. The keywords cannot be used as variable names because by doing so, we are trying to assign a new meaning to the keyword which is not allowed.

The list of C keywords is given below:

auto	**break**	**case**	**char**	**const**
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct

switch	typedef	union	unsigned	void
volatile	while			

Note:

While naming the functions and variables keywords can not be used.

C Tokens

In a passage of text, individual words and punctuation marks are called tokens or lexical units. Similarly, the smallest individual unit in a c program is known as a token or a lexical unit. C tokens can be classified as follows:

1. Keywords
2. Identifiers
3. Constants
4. Strings
5. Special Symbols
6. Operators

C Identifiers

Identifiers are used as the general terminology for the names of variables, functions and arrays. These are user defined names consisting of arbitrarily long sequence of letters and digits with either a letter or the underscore (_) as a first character.

There are certain rules that should be followed while naming c identifiers:

- They must begin with a letter or underscore (_).
- They must consist of only letters, digits, or underscore. No other special character is allowed.
- It should not be a keyword.
- It must not contain white space.
- It should be up to 31 characters long as only first 31 characters are significant.

Some examples of c identifiers:

Name	Remark
A92	Valid
mp.exe	Invalid as it contains special character other than the underscore
void	Invalid as it is a keyword

C Constants

C constants refer to the data items that do not change their value during the program execution.

Several types of C constants that are allowed in C are:

1. Integer Constants

Integer constants are whole numbers without any fractional part. It must have at least one digit and may contain either + or – sign. A number with no sign is assumed to be positive.

There are **three** types of integer constants:

Decimal Integer Constants

Integer constants consisting of a set of digits, 0 through 9, preceded by an optional – or + sign.

Example

341, -341, 0, 8972

Octal Integer Constants

Integer constants consisting of sequence of digits from the set 0 through 7 starting with 0 is said to be octal integer constants.

Example

010, 0424, 0, 0540

Hexadecimal Integer Constants

Hexadecimal integer constants are integer constants having sequence of digits preceded by 0x or 0X. They may also include alphabets from A to F representing numbers 10 to 15.

Example

0xD, 0X8d, 0X, 0xbD

2. Real Constants

The numbers having fractional parts are called real or floating point constants. These may be represented in one of the two forms called **fractional form** or the **exponent form** and may also have either + or – sign preceding it.

Example

0.05, -0.905, 562.05, 0.015

Representing a real constant in exponent form

The general format in which a real number may be represented in exponential or scientific form is

mantissa e exponent

The mantissa must be either an integer or a real number expressed in decimal notation.

The letter e separating the mantissa and the exponent can also be written in uppercase i.e. E and, the exponent must be an integer.

Examples

252E85, 0.15E-10, -3e+8

3. Character Constants

A character constant contains one single character enclosed within single quotes.

Examples: ‘a’ , ‘Z’, ‘5’

It should be noted that character constants have numerical values known as ASCII values, for example, the value of ‘A’ is 65 which is its ASCII value.

4.String Constants

String constants are sequence of characters enclosed within double quotes. Example:

"hello"

"abc"

"hello911"

Every sting constant is automatically terminated with a special character **'\n'** called the **null character** which represents the end of the string.

Note:

The size of the string is the total number of characters plus one for the null character.

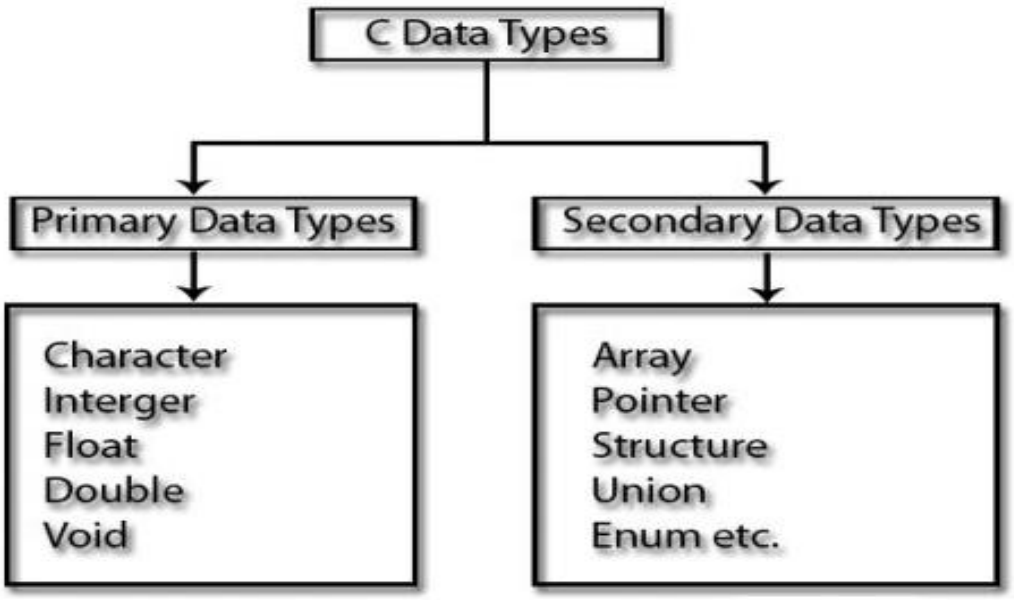
Data Types

The 'data types is used to define a variable before its use. The definition of a variable will assign storage for the variable and define the type of data that will be held in the location.

In C these are of two types

(1) primary (or built in) data types

(2) secondary (or user defined) data types.



Type	Size(bytes)	Range
char	1	-128 to 127
int	2	-32,768 to 32767
Float	4	3.4E-38 to 3.4E+38
double	8	1.7E-308 to 1.7E+308
void	0	-

Modifiers :

The data type modifiers are given below:

1. short 2. long 3. signed 4. unsigned

The modifiers define the amount of storage allocated to the variable. Following table shows this fact:

Type	Bytes	Range
short int	2	-32,768 -> +32,767
unsigned short int	2	0 -> +65,535
unsigned int	2	0 -> +65,535
int or signed int	2	-32,768 -> +32,767
long int	4	-2,147,483,648 -> +2,147,483,647
char or signed char	1	-128 -> +127
unsigned char	1	0 -> +255
float	4	3.4E-38 -> 3.4E+38
double	8	1.7E-308 -> 1.7E+308
long double	10	3.4E-4932 -> 1.1E+4932

Comments

Single Line Comment

Comments are non-executable code used to provide documentation to programmer. Single Line Comment is used to comment out just Single Line in the Code. It is used to provide One Liner Description of line.

1. Single Line Comment Can be Placed Anywhere
2. Single Line Comment Starts with '//'
3. Any Symbols written after '//' are ignored by Compiler
4. Comment cannot hide statements written before '//' and On the Successive new line

Example :

```
#include<stdio.h>
void main()
{
printf("Hello"); //Single Line Comment
printf("By");
}
```

Multi Line Comment

1. Multi line comment can be placed anywhere.
2. Multi line comment starts with `/*`.
3. Multi line comment ends with `*/`.
4. Any symbols written between `'/*'` and `'*/'` are ignored by Compiler.
5. It can be split over multiple lines

Example:

```
#include<stdio.h>
void main()
{
```

```
printf("Hello");
/* Multi
      Line
  Comment
*/
printf("Bye");
}
```