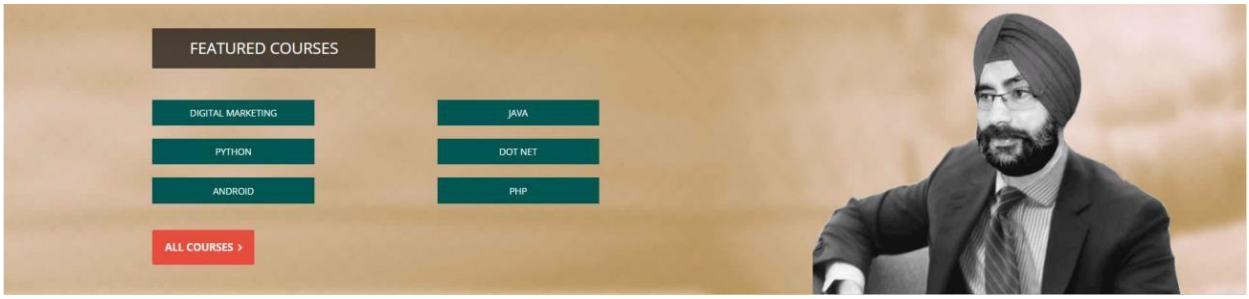
Techgueu – C Language

Mob: 9897039855 www.techgurudehradun.in



History of C Language

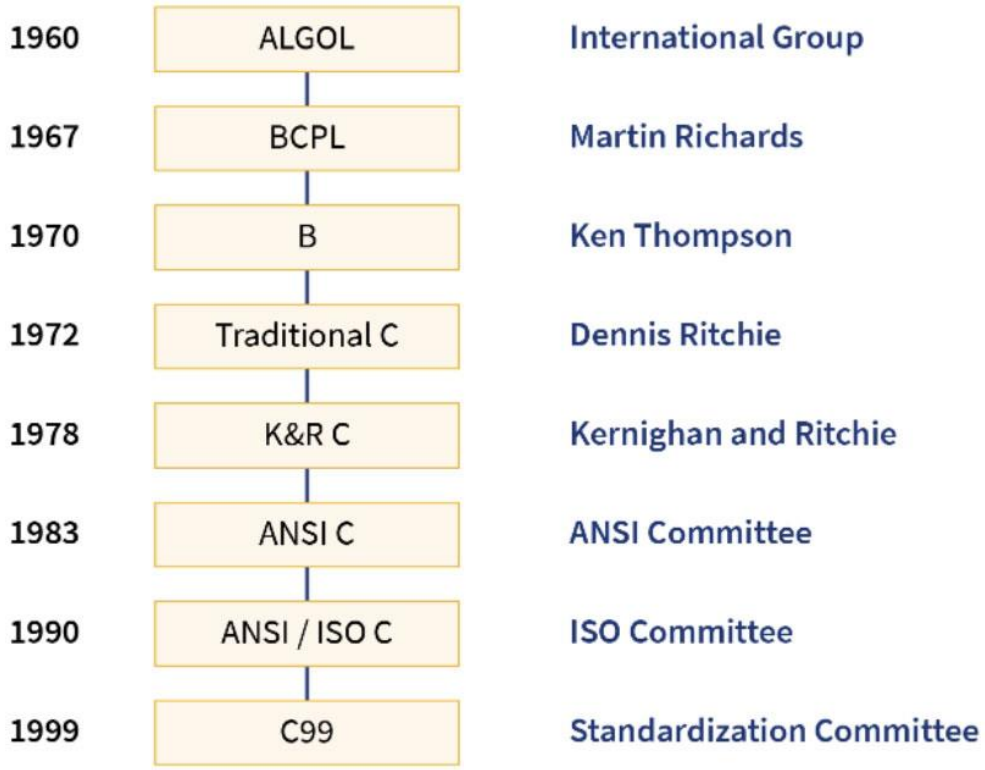
Before learning anything, it is very necessary to know the history of what you are going to learn. The history of the C language is interesting to know. In the early 1970s, the C programming language was developed as a system implementation language for the emerging Unix operating system. It evolved a type structure from the type less language BCPL; started on a small machine as a tool to better a minimal programming environment, it has become one of the most widely used languages today.

Introduction to C Programming Language:

The history of C-language is interesting to know. The C-language is a general-purpose and procedural-oriented programming language. It is a structured and machine-independent programming language. It was developed by Dennis Ritchie in 1972 at the AT&T Bell Laboratories. It was developed along with the UNIX operating system, and is strongly linked with UNIX operating system. History of C language revolves around development as a system implementation language to write an operating system. In terms of the history of C language, its main features include low-level memory access as well as high-level memory access (so it is a middle-level programming language), a handy set of keywords, and a neat and clean style, these features make C programming language suitable for system programming. C supports a wide variety of built-in functions, standard libraries and header files. It follows a top-down approach. Many languages have derived syntax directly or indirectly from the C programming language. For example, C++ is closely a superset of the C language. Also, C programming language is very popular for system-level apps.

History of C Programming Language:

To learn about the history of C language, let's first start with its root and early developments. The root of all modern languages is ALGOL (Algorithmic Language). ALGOL was the first computer programming language to use a block structure, and it was introduced in 1960.
In 1967, Martin Richards developed a language called BCPL (Basic Combined Programming Language). BCPL was derived from ALGOL.
In 1970, Ken Thompson created a language using BCPL called B. Both BCPL and B programming languages were typeless. After that, C was developed using BCPL and B by Dennis Ritchie at the Bell lab in 1972. So, in terms of history of C language, it was used in mainly academic environments, but at long last with the release of many C compilers for commercial use and the increasing popularity of UNIX, it began to gain extensive support among professionals.



Early Implementations and Language Standard:

As discussed, in the history of C language, the development of C was intended to serve as the foundation for the creation of UNIX. By early 1973, the rudiments of ultramodern C had been completed, according to the Bell Labs report. The language and compiler were both powerful enough to rewrite the UNIX kernel in C for the PDP-11. Brian Kernighan and Dennis Ritchie published The C Programming Language in 1978, which served as a reference for the language until a formal standard was established in the history of C language. Between 1973 and 1980, the language evolved slightly: unsigned, long, union, and enumeration types were added to the type structure, and structures became practically first-class objects (lacking only a notation for literals). Its environment, as well as the technology that accompanied it, saw significant changes. In the summer of 1983, American National Standard Institute (ANSI) formed the X3J11 committee under the guidance of CBEMA with the purpose of establishing a C standard. At the end of 1989, X3J11 published its report [ANSI 89], which was later recognised by ISO as ISO/IEC 9899-1990.

A List of Programming Languages Developed Before C Language:

Language	Year of Development	Developer
ALGOL	1960	International Group
BCPL	1967	Martin Richards
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K&R C	1978	Kernighan and Ritchie
ANSI C	1989	ANSI Committee

Language	Year of Development	Developer
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee

The Problems of B Programming Language:

The B programming language has a different importance in the history of C language. As its shortcomings made C a more robust language. The BCPL and B languages were employed on word-addressed machines, and the sole data type in these languages, the 'cell,' was easily equated with the hardware machine word. The introduction of the PDP-11 uncovered various flaws in B's semantic model. First, its character-handling techniques, which were inherited from BCPL with few changes, were cumbersome. Second, by specifying special operators, floating-point operations were introduced to BCPL in Multics and GCOS compilers, but the process was only conceivable because a single word on the relevant machines was large enough to represent a floating-point integer; this was not the case on the 16-bit PDP-11. Finally, the B and BCPL models implied overhead when dealing with pointers: the language rules caused pointers to be represented as word indices by defining a pointer as an index in an array of words. A run-time scale conversion from the pointer to the byte address anticipated by the hardware was generated for each pointer reference.

Standardization of C:

In 1983, ANSI formed the X3J11 committee to standardise the C programming language. The Accredited Standards Committee X3 (ASC X3), Information Technology, was in charge of this endeavour, which led in ANSI X3.159-1989: Programming Language C being ratified on December 14, 1989 and published in the spring of 1990. With some new additions, this original standard unified existing practises. The standard stated in the ANSI X3.159-1989 document was known as ANSI C at the time in the history of C language, however it was quickly superseded when ISO/IEC 9899:1990 was established as an international standard, thanks to the efforts of ISO/IEC JTC 1. While this is where the name ISO C came from, the national and international standards are now known as C89 and C90, respectively.There have been several updates and corrigenda produced in the years since the ISO/IEC 9899 international standard was established. The current C programming language is defined by ISO/IEC 9899:2018 – Information technology – Programming languages – C, the fourth edition of the standard. C11 is the informal term given to the C language established by the 2011 edition of the standard. While neither this, nor the titles ANSI C and ISO C are ever expressly referenced in the standard text, their occasional usage underscores the significance of the hard work carried out by the standards community over the previous thirty years in unifying this programming language.

List of all Keywords in C Language

This tutorial provides a brief information on all 32 keywords in C programming.

Keywords in C Programming

auto	break	case	char
const	continue	default	do
double	else	enum	extern

Keywords in C Programming

float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

Data Types in C

Each variable in C has an associated data type. Each data type requires different amounts of memory and has some specific operations which can be performed over it. Let us briefly describe them one by one:

Following are the examples of some very common data types used in C:

- **char:** The most basic data type in C. It stores a single character and requires a single byte of memory in almost all compilers.
- **int:** As the name suggests, an int variable is used to store an integer.
- **float:** It is used to store decimal numbers (numbers with floating point value) with single precision.
- **double:** It is used to store decimal numbers (numbers with floating point value) with double precision.

Different data types also have different ranges upto which they can store numbers. These ranges may vary from compiler to compiler. Below is list of ranges along with the memory requirement and format specifiers on 32 bit gcc compiler.

Integer Types

The following table provides the details of standard integer types with their storage sizes and value ranges –

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767

unsigned short	2 bytes	0 to 65,535
long	8 bytes or (4bytes for 32 bit OS)	-9223372036854775808 to 9223372036854775807
unsigned long	8 bytes	0 to 18446744073709551615

Floating-Point Types

The following table provide the details of standard floating-point types with storage sizes and value ranges and their precision –

Type	Storage size	Value range	Precision
float	4 byte	1.2E-38 to 3.4E+38	6 decimal places
double	8 byte	2.3E-308 to 1.7E+308	15 decimal places
long double	10 byte	3.4E-4932 to 1.1E+4932	19 decimal places