HISTORY OF C PROGRAMMING LANGUAGE

Evolution of C programming language

C has often been termed as a "Pseudo high level language" or a "Middle level language" by many programmers. This is not because of its lack of programming power but because of its capability to access the system's low level functions. In fact C was invented specifically to implement UNIX. C instructions are compiled to assembly code, therefore, depending on the complexity of the code and on the compiler optimization capabilities, C code may run as fast as assemby.

Ken Thompson created the B language in 1969 from Martin Richard's BCPL (Basic Combined Programming Language). He used assembly language and B to produce the initial versions of the UNIX operating system. BCPL and B were typeless languages in which variables were simply words in memory. Dennis Ritchie of Bell Laboratories later converted B into C by retaining most of B's syntax in 1972 and wrote the first compiler. This was implemented on DEC's PDP 11 and it was first used as the system's language for rewriting the UNIX operating system. Later on, UNIX, its tools and C grew simultaneously. In 1978, Kernighan and Ritchie wrote a book entitled 'The C Programming Language' that became the language definition for almost a decade. Beginning in 1983, the ANSI X3J11 committee was asked to standardize the C language. The result was ANSI C, a standard which was adopted in 1988. It is not forced upon any programmer, but since it is so widely accepted, it would be economically unwise for any systems programmer or compiler writer not to conform to the standard.

Features of C Language

- C is a procedure-based programming language. This means the program is viewed as a means to solve a problem. Various functions modules or code blocks are thus, written to solve this problem.
- C functions can accept parameters and return values and perform variety of tasks like input from the user, displaying the information, etc.
- C is simple and easy to learn and use. The main components like built-in functions, operators, keywords are small in number.
- In C, errors are checked only at compile time. The compiled code though have no safety checks for bad type casts, bad array indices, or bad pointers.
- C works best for small projects where performance is important.
- C contains the capability of assembly language with the features of high level language which can be used for creating software packages, system software etc.
- C is highly portable. C programs written on one computer can run on other computer without making any changes in the program.

;

Uses of C

C's wide acceptance and efficiency is the reason why libraries of several other applications are often implemented in C. Some of the applications using C in its kernels are:

Uses of C are many in addition to Systems programming. Some of which are as follows :

- Language compilers and interpreters
- Device drivers
- Telecom applications
- Network programming
- Digital Signal processing applications
- Database applications
- Text editors

Compilers available

Most flavors of UNIX operating system have a C compiler. So if a student has access to an UNIX system, he or she can straight away start coding and compiling C sample programs and exercises given in this course. With respect to personal computers , Microsoft C for IBM PC's and compatibles or Borland C are the two most commonly used programming environments. There are a number of free C compilers available on the web for installation on the PC, one can use them as well.

Future of C

The current popularity of C++ may seem to have displaced C's position in the programming world. But C is here to stay for a very long time. One main factor is that C++ has inherited most of its syntax from C but has incorporated several new concepts which form the basis of Object Oriented programming. It is better to know C in order to learn C++ though there are many who advocate the theory that one has to unlearn procedural programming habits in order to learn Object Oriented programming. GUI based C++ programming environments are more popular, and use lot of disk space and extended memory. They use complex class libraries and are not well suited for developing small programs that run on smaller systems. C is a better option when it comes to programming device drivers, embedded applications and utility programs.

Source : http://www.peoi.org/Courses/Coursesen/cprog/frame1.html