Introduction

The programming language C was originally developed by Dennis Ritchie of Bell Laboratories and was designed to run on a PDP-11 with a UNIX operating system. Although it was originally intended to run under UNIX, there has been a great interest in running it under the MS-DOS operating system on the IBM PC and compatibles. It is an excellent language for this environment because of the simplicity of expression, the compactness of the code, and the wide range of applicability. Also, due to the simplicity and ease of writing a C compiler, it is usually the first high level language available on any new computer, including microcomputers, minicomputers, and mainframes.

C is not a good beginning language because it is somewhat cryptic in nature. It allows the programmer a wide range of operations from high level down to a very low level, approaching the level of assembly language. There seems to be no limit to the flexibility available. One experienced C programmer made the statement, "You can program anything in C", and the statement is well supported by my own experience with the language. Along with the resulting freedom however, you take on a great deal of responsibility because it is very easy to write a program that destroys itself due to the silly little errors that a good Pascal compiler will flag and call a fatal error. In C, you are very much on your own as you will soon find.

I ASSUME YOU KNOW A LITTLE PROGRAMMING

Since C is not a beginners language, I will assume you are not a beginning programmer, and I will not attempt to bore you by defining a constant and a variable. You will be expected to know these basic concepts. You will, however, be expected to know nothing of the C programming language. I will begin with the most basic concepts of C and take you up to the highest level of C programming including the usually intimidating concepts of pointers, structures, and dynamic allocation. To fully understand these concepts, it will take a good bit of time and work on your part because they are not particularly easy to grasp, but they are very powerful tools. Enough said about that, you will see their power when we get there, just don't allow yourself to worry about them yet.

Programming in C is a tremendous asset in those areas where you may want to use Assembly Language but would rather keep it a "simple to write" and "easy to maintain" program. It has been said that a program written in C will pay a premium of a 20 to 50% increase in runtime because no high level language is as compact or as fast as Assembly Language. However, the time saved in coding can be tremendous, making it the most desirable language for many programming chores. In addition, since most programs spend 90 percent of their operating time in only 10 percent or less of the code, it is possible to write a program in C, then rewrite a small portion of the code in Assembly Language and approach the execution speed of the same program if it were written entirely in Assembly Language.

Even though the C language enjoys a good record when programs are transported from one implementation to another, there are differences in compilers as you will find anytime you try to use another compiler. Most of the differences become apparent when you use nonstandard extensions such as calls to the DOS BIOS when using MS-DOS, but even these differences can be minimized by careful choice of programming constructs.

Throughout this tutorial, every attempt will be made to indicate to you what constructs are available in every C compiler because they are part of the accepted standard of programming practice.

WHAT IS THE ANSI-C STANDARD?

When it became evident that the C programming language was becoming a very popular language available on a wide range of computers, a group of concerned individuals met to propose a standard set of rules for the use of the C programming language. The group represented all sectors of the software industry and after many meetings, and many preliminary drafts, they finally wrote an acceptable standard for the C language. It has been accepted by the American National Standards Institute (ANSI), and by the International Standards Organization (ISO). It is not forced upon any group or user, but since it is so widely accepted, it would be economical suicide for any compiler writer to refuse to conform to the standard.

YOU MAY NEED A LITTLE HELP

Modern C compilers are very capable systems, but due to the tremendous versatility of a C compiler, it could be very difficult for you to learn how to use it effectively. If you are a complete novice to programming, you will probably find the installation instructions somewhat confusing. You may be able to find a colleague or friend that is knowledgeable about computers to aid you in setting up your compiler for use.

This tutorial cannot cover all aspects of programming in C, simply because there is too much to cover, but it will instruct you in all you need for the majority of your programming in C, and it will introduce essentially all of the C language. You will receive instruction in all of the programming constructs in C, but what must be omitted are methods of programming since these can only be learned by experience. More importantly, it will teach you the vocabulary of C so that you can go on to more advanced techniques using the programming language C. A diligent effort on your part to study the material presented in this tutorial will result in a solid base of knowledge of the C programming language. You will then be able to intelligently read technical articles or other textbooks on C and greatly expand your knowledge of this modern and very popular programming language.

HOW TO USE THIS TUTORIAL

This tutorial is written in such a way that the student should sit before his computer and study each example program by displaying it on the monitor and reading the text which corresponds to that program. Following his study of each program, he should then compile and execute it and observe the results of execution with his compiler. This enables the student to gain experience using his compiler while he is learning the C programming language. It is strongly recommended that the student study each example program in the given sequence then write the programs suggested at the end of each chapter in order to gain experience in writing C programs.

THIS IS WRITTEN PRIMARILY FOR MS-DOS

This tutorial is written primarily for use on an IBM-PC or compatible computer but can be used with any ANSI standard compiler since it conforms so closely to the ANSI standard. In fact, a computer is not even required to study this material since the result of execution of each example program is given in comments at the end of each program.

RECOMMENDED READING AND REFERENCE MATERIAL

"The C Programming Language - Second Edition", Brian W. Kernigan & Dennis M. Ritchie, Prentiss-Hall, 1988

This is the definitive text of the C programming language and is required reading for every serious C

programmer. Although the first edition was terse and difficult to read, this edition is easier to read and extremely useful as both a learning resource and a reference guide.

Any ANSI-C textbook

Each student should posess a copy of a book that includes a definition of the entire ANSI-C specification and library. Go to a good bookstore and browse for one.

Index Next . . . . . *C Tutorial*

The Webwizard