UNIT 01 - Introduction to C Programming

1.0 Objectives

- Understand the history and development of C programming language
- Identify the features and strengths of C programming language
- Describe the basic structure of a C program
- Develop a basic understanding of programming concepts and syntax in C programming language.

1.1 Introduction

C programming language is a general-purpose, high-level language that provides low-level memory access, system-level programming, and a relatively simple syntax that makes it accessible to beginners. It was originally developed by Dennis Ritchie at Bell Labs in the early 1970s to create the UNIX operating system, and has since become one of the most widely used programming languages in the world.

This unit provides an introduction to C programming, covering the history, features, and basic structure of the language. Students will learn about the strengths and limitations of C programming, as well as the fundamental concepts and syntax needed to write simple C programs

1.2 Overview of the C language

C is a high-level, general-purpose programming language that was developed in the 1970s by Dennis Ritchie at Bell Labs. It is a compiled language that allows developers to create efficient, low-level programs while also providing high-level abstractions that make it easy to write complex programs.

C is a popular language for system programming, including operating systems, device drivers, embedded systems, and other low-level applications. It is also widely used in application programming, including business applications, scientific applications, and games.

One of the main features of C is its simplicity. It has a relatively small set of keywords and operators, making it easy to learn and use. At the same time, it provides powerful features for low-level programming, such as direct access to memory, efficient use of resources, and precise control over program flow.

C is also a portable language, meaning that programs written in C can be compiled and run on a wide range of computer systems, including desktops, servers, and embedded devices. This is due to the use of a standard set of libraries and compilers, such as the GNU Compiler Collection (GCC), which are available on many platforms.

Overall, the C language has had a significant impact on the field of computer science and programming, and it remains a popular choice for developers today due to its power, efficiency, and flexibility.

1.3 History of C language

C programming language was created by Dennis Ritchie at Bell Laboratories in 1972. However, its development began in 1969 as an internal project at Bell Labs, with Ken Thompson as the lead developer. The development of C language was initially intended to create a language for developing the Unix operating system. It was designed to be a low-level language that provided direct access to the system hardware, making it suitable for system programming.

C programming language was developed as a successor to the B programming language. B was developed by Ken Thompson at Bell Labs in 1970, which was derived from the BCPL programming language developed by Martin Richards. B was an interpreted language, and its performance was poor. Ritchie then developed C programming language, which was faster and more efficient than B.

The early versions of C were developed on the DEC PDP-11 computer using the Unix operating system. C programming language became popular with the release of Unix Version 6 operating system in 1975. As Unix became more popular, so did C programming language.

In 1978, the first edition of "The C Programming Language" book, also known as the K&R C book, was published by Brian Kernighan and Dennis Ritchie. The book became very popular and played a significant role in the widespread adoption of C programming language.

C programming language was standardized by the American National Standards Institute (ANSI) in 1989. The ANSI C standard, also known as C89 or C90, provided a common specification for C programming language, which helped in portability and interoperability of C programs.

In 1999, the ISO (International Organization for Standardization) released the C99 standard, which added several new features to the language, such as inline functions, variable-length arrays, and support for IEEE floating-point arithmetic.

Since then, there have been several revisions to the C programming language, including C11 and C18. The latest version of the C standard, C2x, is currently under development and is expected to be released in the coming years.

Despite being over 50 years old, C programming language is still widely used today for system programming, embedded systems, and high-performance computing applications. It has influenced the development of many other programming languages, including C++, Java, and Python, and has left a significant impact on the field of computer science.

1.4 Features

C programming language has several features that make it a popular choice for developing a wide range of applications. Some of the key features of C programming language are:

- Simple and Easy to Learn: C programming language is a simple language with a small number of keywords and constructs, making it easy to learn for beginners.
- Portable: C programs can be written and compiled on different platforms, including Windows, Mac, and Linux, making it a portable language.
- High-level Language: C programming language is a high-level language that provides abstracted programming concepts, allowing programmers to focus on the logic of the program rather than machine-level details.
- Structured Programming: C programming language supports structured programming constructs like loops, conditional statements, and functions, allowing developers to write structured, easy-to-read code.

- Efficient Memory Management: C programming language provides manual memory management, allowing developers to control the allocation and deallocation of memory, resulting in efficient memory utilization.
- Rich Library Support: C programming language has a rich set of libraries that provide access to a wide range of functions, making it easy to implement complex functionality.
- Speed: C programming language is a compiled language, resulting in faster execution times compared to interpreted languages.
- Low-level Language Features: C programming language provides low-level language features like direct access to memory and pointers, making it possible to implement low-level functionality.
- Platform-Specific Programming: C programming language allows for platform-specific programming, making it possible to write code that interacts directly with the hardware.

Overall, the combination of simplicity, portability, efficiency, and rich functionality makes C programming language a powerful tool for developing a wide range of applications.

1.5 Strengths of C programming language

C programming language has various strengths, including:

- Efficiency: C is a low-level programming language, which means it is closer to the machine language that the computer understands. As a result, it allows developers to write programs that are highly optimized for the hardware, making them run faster and use less memory.
- Portability: C programming language is highly portable, meaning that the same code can be compiled and executed on different platforms without any significant changes. This portability is one of the primary reasons why C is so widely used for system programming.
- Modularity: C programming language supports modular programming, which means that large programs can be divided into smaller, more manageable modules.

This modularity makes it easier to develop, test, and maintain large software systems.

- Flexibility: C programming language is a flexible language that allows developers to choose the best programming approach to solve a problem. C supports both procedural and object-oriented programming paradigms, making it suitable for a wide range of applications.
- Large community: C programming language has a vast and active community of developers, which means that there is a wealth of information and resources available to developers who are just starting with the language.
- Widely used: C programming language is widely used in various fields such as system programming, embedded systems, gaming, and scientific computing. Its widespread adoption means that there is a high demand for C programmers in the job market.
- Powerful language: C programming language is a powerful language that can be used to write complex and efficient programs. It offers low-level memory access and supports bit manipulation, making it suitable for developing high-performance programs.

1.6 Limitations of C programming language

Some of the limitations of C Programming language:

- Lack of built-in support for OOP: C is not an object-oriented programming (OOP) language, and it does not have built-in support for OOP concepts like inheritance and polymorphism. This makes it more difficult to write large, complex programs.
- No automatic garbage collection: Unlike some other programming languages, C does not have automatic garbage collection. This means that the programmer has to manually manage memory allocation and deallocation, which can be error-prone and time-consuming.
- Lack of standard libraries: C does not have a large standard library compared to some other programming languages. This means that programmers often have to rely on third-party libraries or write their own code to perform common tasks.

- Difficulty with string manipulation: C does not have built-in support for string manipulation, which can make it more difficult to write programs that work with strings.
- Vulnerable to buffer overflows: C is vulnerable to buffer overflows, which can lead to security vulnerabilities in programs. This is because C does not perform bounds checking on arrays by default, so it is up to the programmer to ensure that their code is safe from buffer overflow attacks.
- Lack of exception handling: C does not have built-in support for exception handling, which can make it more difficult to write code that handles errors gracefully.
- Lack of platform independence: C code is not inherently platform independent, which means that code written for one platform may not work on another platform without modification. This can be a significant limitation for programs that need to run on multiple platforms.
- Despite these limitations, C remains a widely used programming language, particularly in systems programming and embedded systems.

1.7 General Syntax

The general syntax of C programming language is as follows:

```
/* Header Files */

/* Global Variable Declaration */

int main() {

   /* Local Variable Declaration */

      /* Statements */

return 0;

}
```

Here is an example of a simple C program that prints "Hello, World!”

```
#include <stdio.h>

int main()

{

printf("Hello, World!");

return 0;

}
```

In this program, the #include statement is used to include the standard input/output library stdio.h. The intmain() function is the entry point of the program, and it returns an integer value. Inside the main function, the printf() function is used to display the string "Hello, World!" on the screen. The return 0 statement indicates that the program has completed successfully.

1.8 Summary

In this unit, we were introduced to the C programming language, including its history, features, strengths, and limitations. We learned that C is a powerful, efficient, and widely used programming language that is commonly used for system programming, embedded systems, and game development, among other applications.

We also learned about the basic structure of a C program and its components, including comments, preprocessor directives, functions, and variables. Additionally, we covered the concept of constants, variables, and data types, including integer, floating-point, character, and void data types.

Overall, this unit provided a solid foundation for understanding the basics of C programming, which we will build upon in future units.

1.9 Keywords

▪ C programming language ▪ Syntax ▪ Program structure ▪ Compiler ▪ History ▪ Features	▪ Portability ▪ Efficiency ▪ Object-oriented programming (OOP) ▪ Procedure-oriented programming (POP)

1.10 Questions for Self Study

1. Write the history of the C programming language? Explain its evolution.
2. What are the features and strengths of the C programming language?
3. What are the limitations of the C programming language?
4. Explain the basic structure of a C program with an example.

1.11 REFERENCE BOOKS

1. "Structured Programming Approach in C" by Forouzan and Gilberg
2. "Programming with C" by E. Balagurusamy
3. "The C Programming Language" by Brian W. Kernighan and Dennis M. Ritchie
4. "C Programming Absolute Beginner's Guide" by Greg Perry and Dean Miller