**Introduction to C Language**

**Introduction to programming Language**:

 A computer is a machine that accepts, stores, processes the data and produces the results accurately as instructed by a program. **To write programs some languages are required, called programming languages.**

**1.1 Define various levels of computer languages:**

 A computer language is defined as a medium through which a computer is instructed to do a particular task. Basically computer languages are classified into 3-levels.

1. Low level language.
2. Middle level language.
3. High level language.

Low level language:

* Low level language is defined as **a language which uses 0 s and 1s and can be understood by the computers directly**.

e.g: + can be written as 0001 and minus can be written as 0011

* It is also called as **Machine Level language** or **Binary language**.
* **Computer can understand** this language **without any translator**.
* But, User has to remember many lengthy binary values to write programs.
* Programs written in low level language will be **larger in size**.
* It is **difficult to learn and understand**.
* Debugging (detecting and correcting errors) is so difficult.
* It takes **more time to develop**.
* Machine dependent.
* E.g: machine language only.

Middle level language:

* Middle level language is defined as **a language which uses codes like ADD,SUB, MUL, IDIV…**to develop programs.

e.g: ADD A, B /\*Indicates A+B\*/

* It is also called as **Assembly Level language** or **Coded (mnemonic) language**.
* **Computer cannot understand** this language directly, but it has to be translated into machine language with the help of a translator called an **assembler**.
* But, User has to remember many codes to write programs.
* Programs written in middle level language will be **in moderate size**.
* It is **moderate to learn and understand**.
* Debugging (detecting and correcting errors) is moderate.
* It takes **moderate time to develop**.
* Machine dependent.
* E.g: 8085, 8086, 80186………..

High level language:

* High level language is defined as **a language which uses English language symbols like +,-, a, b, main** to develop programs.

e.g: c=a+b; /\*Indicates A+B\*/

* It is also called as **user friendly language**.
* **Computer cannot understand** this language directly, but it has to be translated into machine language with the help of a translator called a **compiler**.
* But, User has to **remember the syntaxes only** but not any codes to write programs.
* Programs written in high level language will be **very small in size**.
* It is **easy to learn and understand**.
* Debugging (detecting and correcting errors) is easy.
* It takes **less time to develop programs**.
* Machine in-dependent.
* E.g: ALGOL, BASIC, B, BCPL, C, C++, JAVA, VB……….
	1. **History of C programming language**:

 C language was derived from the earlier language B, B is derived from BCPL (Basic Combined Programming Language), BCPL was derived from ALGOL(Algorithmic Language).C uses many concepts from these languages and added the concept of data types and other powerful features.

1. ALGOL was the first computer language which gave the concept of **structured programming.**
2. The name C is given to this language because it was derived from B (**C was successor of B**)
3. C is born at “**AT & T’s Bell Laboratory**” of USA in **1972**.
4. C was written by a person called **Dennis Ritchie**, that’s why he is called as “father of C”

|  |  |  |
| --- | --- | --- |
| Language | Development year | Developed By |
| ALGOL | 1960 | International Group |
| BCPL | 1967 | Martin Richards |
| B | 1970 | Ken Thompson |
| Traditional C | 1972 | Dennis Ritchie |
| K & R C | 1978 | Brain Kernighan & Dennis Ritchie |
| ANSI C | 1989 | ANSI committee |
| ANSI C/ ISO C | 1990 | ISO committee |

**1.3Importance/features of C language**:

 The C language has gained popularity due to its many desirable qualities. They are:

1. **C is a robust language**: robust means ‘**strong**’. It is a robust language whose rich set of built-in functions and operators can be used to write any complex program.
2. It is **well suited for writing both system software and business packages**, because its c compiler combines the capabilities of assembly language with the features of a high-level language.
3. Programs written in C are **efficient and fast**: this is due to its variety of data types and powerful operators. It is many times faster than BASIC, e.g: a program to increment a variable form 0 to 15000 takes about 1-second in C while it takes more than 50-seconds in BASIC.
4. Several standard functions are available which can be used for developing programs.
5. C language is well suited for structured programming. This makes program debugging, testing and maintenance easier.
6. **Ability to extend itself**: we can continuously add our own functions to C library. With the availability of a larger number of functions, the programming task becomes simple.
7. **C is highly portable**: this means that C programs written for one computer can be run on another.

**1.4Structure of C language program**:

 To write programs in C one should follow the basic structure. Structure of C program includes various sections/parts. All these sections may or may not be used in all the programs.

|  |
| --- |
| Documentation section (or) comment line section |
| Link section |
| Definition section |
| Global declaration section |
| main() function section{

|  |
| --- |
| Declaration part |
| Execution part |

} |
| Subprogram section (user defined functions)

|  |
| --- |
| Function1 |
| Function2 |
| …… |
| …… |
|  Function n |

 |

1. Documentation section: Consists of a set of lines **gives the brief explanation of the program, author, date …….**

 e.g.: /\* C –program to find factorial of a given number, date:29/07/15,2.30\*/

1. Link section: This section **instructs the compiler to link functions from the system library.**

 e.g.: #include<stdio.h>

1. Definition section: Defines all the symbolic constants.

 e.g.: #define PI 3.14

1. Global declaration section: There are some variables that are used in more than one function. Such variables are called **global variables** and **are declared in the global declaration section.**

 e.g.: int total=0;

This section also declares all the user defined functions.

1. main() function section: Every C program must have one main() function. The program execution begins at the opening brace and ends at the closing brace. This section contains 2 parts:
* declaration part: declares all the variable used in the execution part
* execution part.
1. The sub program section : Contains all the user-defined functions that are called in the main () function.

 Note: all sections, except the main() function are optional.

**1.5Programming Style of C language**:

 The style of a C language describes **how to use various elements of C language to increase clarity, readability and understandability of c programs**.

* We must develop the habit of **writing programs in lowercase letters**. C program statements are in lowercase letters. Uppercase letters used only for symbolic constants.
* **Use braces**, to group statements together and mark the beginning and the end of functions.
* A proper indentation of braces and statements would make a program easier to read and debug.

e.g: main() { printf(“Welcome to C lab”); }

can be re written as :

 main ()

 {

 printf(“Welcome to C lab”);

 }

* **Inserting comments inside a program** can increase the readability and also helps to understand the program logic.

**1.6Execution of the C program**:

 Executing a program written in c involves a series of steps. These are:

1. Creating the program.
2. Compiling the program.
3. Linking the program with functions that are needed from c library.
4. Executing the program.

🡪**Creating**: we can create or write a c program using turbo c –IDE. In this, tc.exe will be there double click on this we get blue screen then type the c program and save the file using f2 key with an extension .c .

🡪**Compiling**: in turbo C compiler use Alt+f9 to compile the c-program, to check whether the source code contains any errors.If source contains errors then goto step1 to modify the source program. After compiling the source program we get .obj file.

🡪**linking** : this process is inbuilt in step2 in turbo Compiler. The .obj file is given as input to linker. The linker checks for errors .if object code contains any errors then goto step1 to modify. If the object code is error free then linker converts .obj file into .exe file.

🡪**executing**: CTRL+F9 in turbo C.

The pictorial representation of the above steps is:

