Programming Language Translators

A translator is a computer program that translates a program written in a given programming language into a functionally equivalent program in a different computer language, without losing the functional or logical structure of the original code (the "essence" of each program).

Translator Types:

Interpreter

In computing, an interpreter is a computer program that reads the source code of another computer program and executes that program. Because it is interpreted line by line, it is a much slower way of running a program than one that has been compiled but is easier for learners because the program can be stopped, modified and re-run without time-consuming compiles. An interpreter generally uses one of the following strategies for program execution:

- parse the source code and perform its behavior directly
- translate source code into some efficient intermediate representation and immediately execute this
- explicitly execute stored precompiled code made by a compiler which is part of the interpreter system

Compiler

A compiler is a computer program that transforms human readable source code of another computer program into the machine readable code that a CPU can execute. It is a software program that compiles program source code files into an executable program.

The act of transforming source code into machine code is called "compilation". This is the opposite to the process of interpretation. It is included as part of the integrated development environment IDE with most programming software packages.

Interpreter Vs. Compiler

S.no	Compiler	Interpreter
1	Compiler takes Entire program as Input	Interpreter takes Single instruction as input
2	Intermediate Object Code is Generated	No Intermediate Object Code is Generated

3	Conditional Control Statements are Executes faster	Conditional Control Statements are Executes slower
4	Memory Requirements: More (Since Object Code is Generated)	Memory Requirement is Less
5	Program need not be compiled every time	Every time higher level program is converted into lower level program
6	Errors are displayed after entire program is checked	Errors are displayed for every instruction interpreted (if any)
7	Example: C Compiler	Example: BASIC

Structures of C++

```
int main()
{
        Variable declaration
        section Function
        declaration section
        executable
        statements;
}
```

Document Section: It consists of set of comment lines which include name of a program, author name, creation date and other information

Links Section (File): It is used to link the required system libraries or header files to execute a program.

Definition Section: It is used to define or set values to variables.

Global variable declaration Section: It is used to declare global or public variable.

void main (): Used to start of actual C program. It includes two parts as declaration part and executable part.

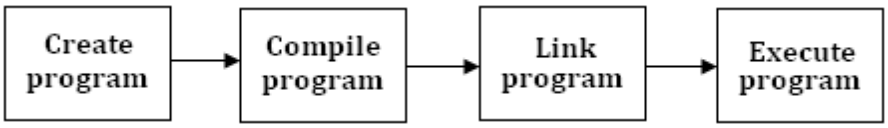
Variable declaration section: Used to declare private variable.

Function declaration section: Used to declare functions of program from which we get required output. Then, executable statements are placed for execution.

Function definition section: Used to define functions which are to be called from main ().

Compilation & Execution Process

C++ program executes in following 4 (four steps).



Creating a Program :

An editor like notepad or WordPad is used to create a C++ program. This file contains a source code which consists of executable code. The file should be saved as '*.c++ extension only.

Compiling the Program :

The next step is to compile the program. The code is compiled by using compiler. Compiler converts executable code to binary code i.e. object code.

Linking a Program to Library :

The object code of a program is linked with libraries that are needed for execution of a program. The linker is used to link the program with libraries. It creates a file with '*.exe' extension (in Windows) / a.out file (in Linux).

Execution of program :

The final executable file is then run by dos command prompt or by any other software.

Hello World Program:

```
//C++ hello world
#include
<iostream>

int main()
{
printf("Hello world”);
  return 0;
}
```