**SIMPLE C - COMPILER**

**INTRODUCTION:**  
A compiler is a translating program that translates the instructions of high-level language to machine level language. A program which is input to the compiler is called a **Source program**. This program is now converted to a machine level language by a compiler is known as the **Object code.**

Following students of CSE 3rd year are the part of a team working on this project:

1. **Gourav Bansal – 191220020**
2. **Kishan Srivastava – 191220030**
3. **Vardan Agarwal – 191220051**

**PROBLEM STATEMENT:**

To build a complete compiler that accepts a high-level language as input and produces working assembly code as output.

**PROBLEM DOMAIN:**

● Looping construct: while, for, do-while

● Data types: (signed/unsigned) int, float

● Arithmetic and Relational Operators

● Data structure: Arrays

● User defined functions

● Keywords of C language

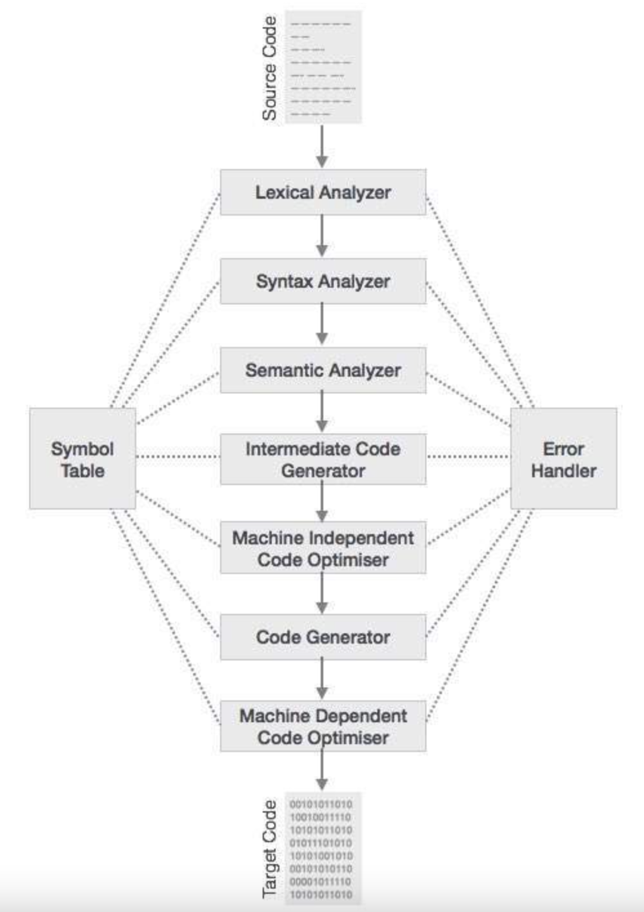
● Single and Multi-line comments

● Identifiers and Constant errors

● Selection statement: (nested) if-else

**DESIGN IDEA:**

The compilation process is a sequence of various phases. Each phase takes input from its previous stage, has its own representation of source program, and feeds its output to the next phase of the compiler.



**Lexical Analysis**

The first phase of scanner works as a text scanner. This phase scans the source code as a stream of characters and converts it into meaningful lexemes.

Implementation planned using a lex program.

### Syntax Analysis

The next phase is called the syntax analysis or **parsing**. It takes the token produced by lexical analysis as input and generates a parse tree (or syntax tree).

Implementation of parser planned using lex and yacc programs.

### Semantic Analysis

Semantic analysis checks whether the parse tree constructed follows the rules of language.

Implementation of parser planned using lex and yacc programs.

### Intermediate Code Generation

After semantic analysis the compiler generates an intermediate code of the source code for the target machine.

Implementation planned using lex and yacc programs.

\*Test cases will be .c type files.

### Code Optimization

The next phase does code optimization of the intermediate code.

### Code Generation

In this phase, the code generator takes the optimized representation of the intermediate code and maps it to the target machine language.

**HARDWARE REQUIREMENTS**

Assuming, that we are designing a very basic C compiler, it will have to take very few information as input as far as memory is concerned. Further the whole information has to be processed. So there is no need for a high specs system for running this.

All this suggest that the minimum hardware requirements should be:-

• Memory (RAM): 1 GB of RAM required.

• Hard Disk Space: 200 MB of free space required.

• Processor: Intel Pentium 4 or later.

• Cache: 512KB

**SOFTWARE REQUIRMENTS**

Following software should be installed for smooth functioning:

* Operating System: Windows 7/8/8.1/10/11/MacOs/LINUX
* FLEX
* BISON (yacc)
* gcc compiler
* Code editor for editing files