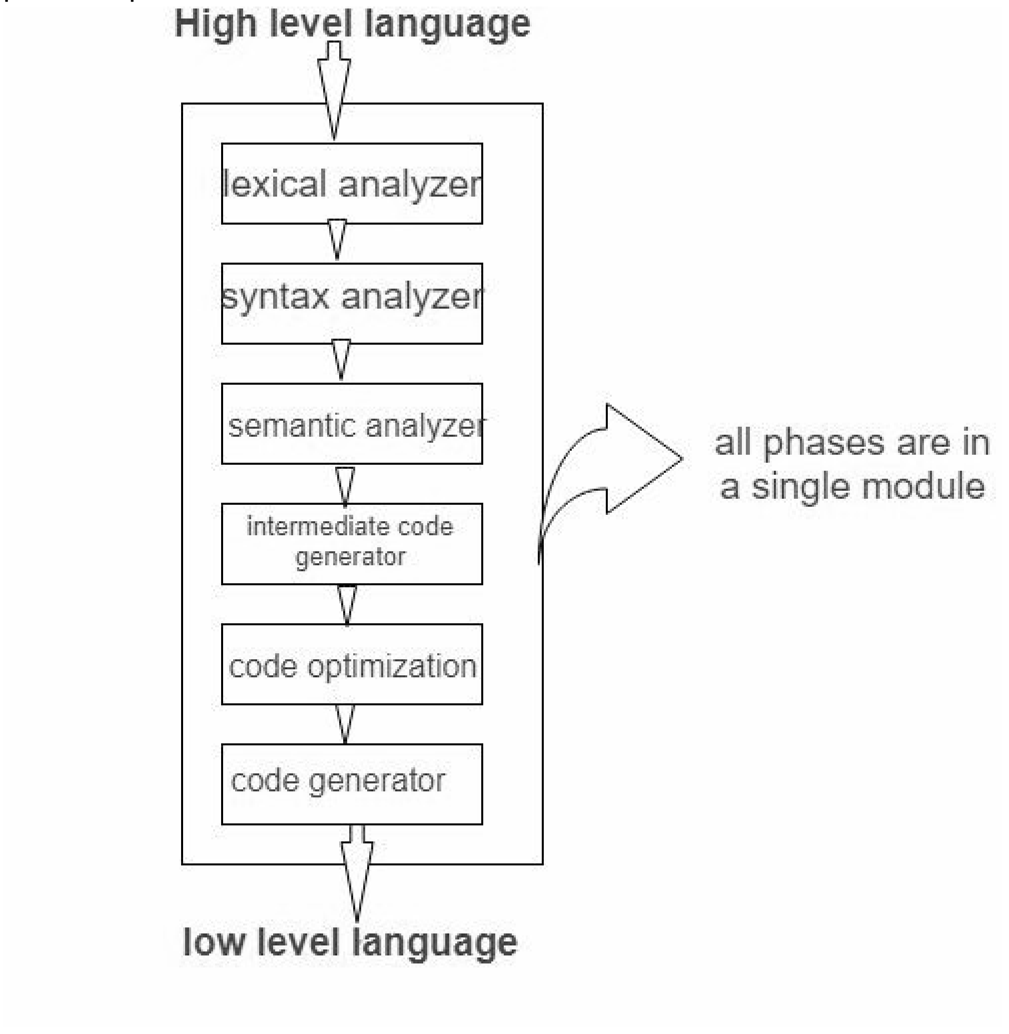
SINGLE PASS AND MULTI-PASS COMPILER

A **Compiler** pass refers to the traversal of a compiler through the entire program. Compiler pass are two types: Single Pass Compiler, and Two Pass Compiler or Multi Pass Compiler.

1. Single Pass Compiler:

If we combine or group all the phases of compiler design in a single module known as single pass compiler.



In above diagram there are all 6 phases are grouped in a single module.

A one pass/single pass compiler is that type of compiler that passes through the part of each compilation unit exactly once.
Single pass compiler is faster and smaller than the multi pass compiler.

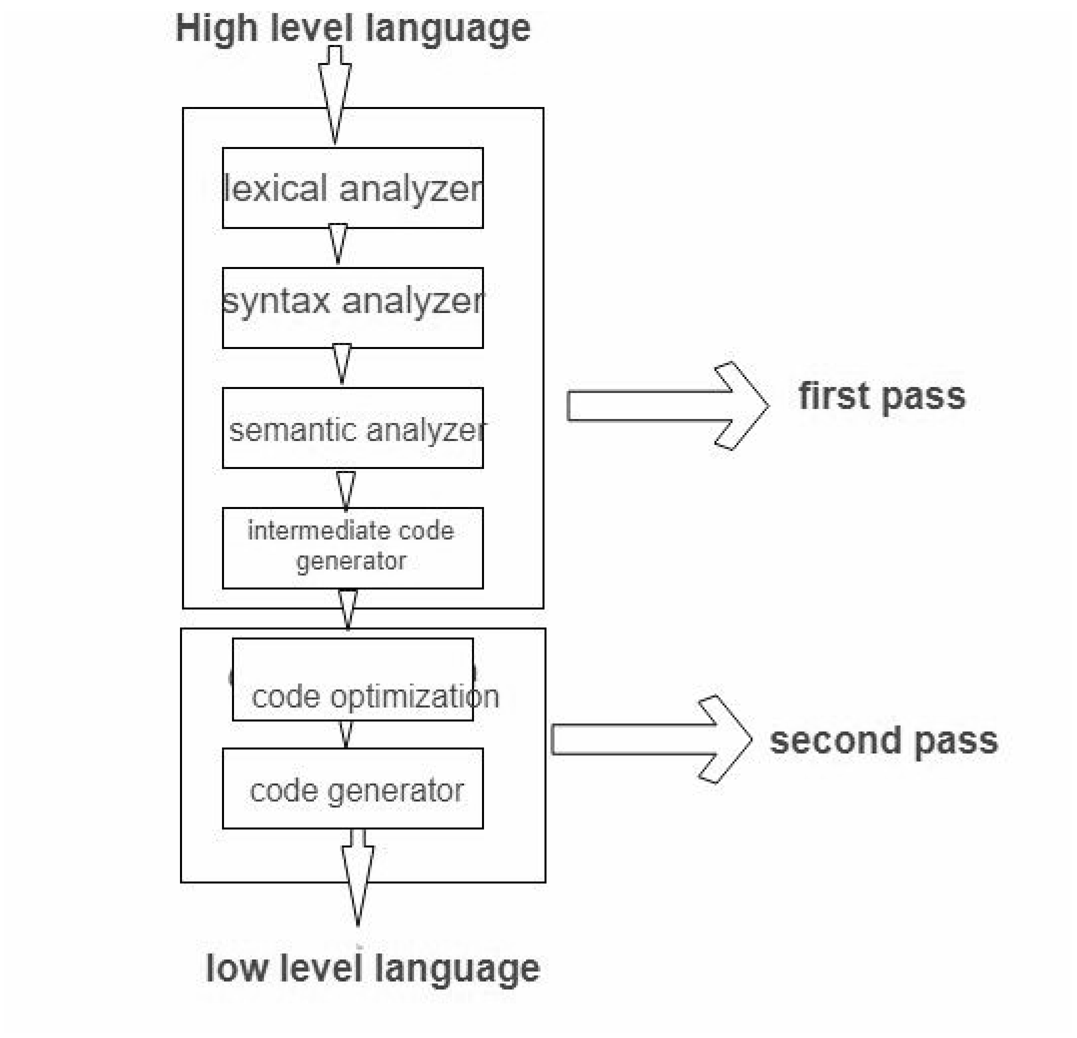
As a **disadvantage** of single pass compiler is that it is less efficient in comparison with multipass compiler.
Single pass compiler is one that processes the input exactly once, so going directly from lexical

analysis to code generator, and then going back for the next read.

Problems with single pass compiler:

We can not optimize very well due to the context of expressions are limited.
As we can’t backup and process, it again so grammar should be limited or simplified.
Command interpreters such as bash/sh/tcsh can be considered as Single pass compiler, but they also execute entry as soon as they are processed.

2. Two Pass compiler or Multi Pass compiler:
A Two pass/multi-pass Compiler is a type of compiler that processes the source code or abstract syntax tree of a program multiple times. In multipass Compiler we divide phases in two pass as:



In first pass the included phases are as Lexical analyzer, syntax analyzer, semantic analyzer, intermediate code generator are work as front end and analytic part means all phases analyze the High level language and convert them three address code and first pass is platform independent because the output of first pass is as three address code which is useful for every system and the requirement is to change the code optimization and code generator phase which are comes to the second pass.

In second Pass the included phases are as Code optimization and Code generator are work as back end and the synthesis part refers to taking input as three address code and convert them into Low level language/assembly language and second pass is platform dependent because final stage of a typical compiler converts the intermediate representation of program into an executable set of instructions which is dependent on the system.

Differences between Single Pass and Multipass Compilers:

Parameters	**Single pass**	**multi Pass**
Speed	Fast	Slow
Memory	More	Less
Time	Less	More
Portability	No	Yes