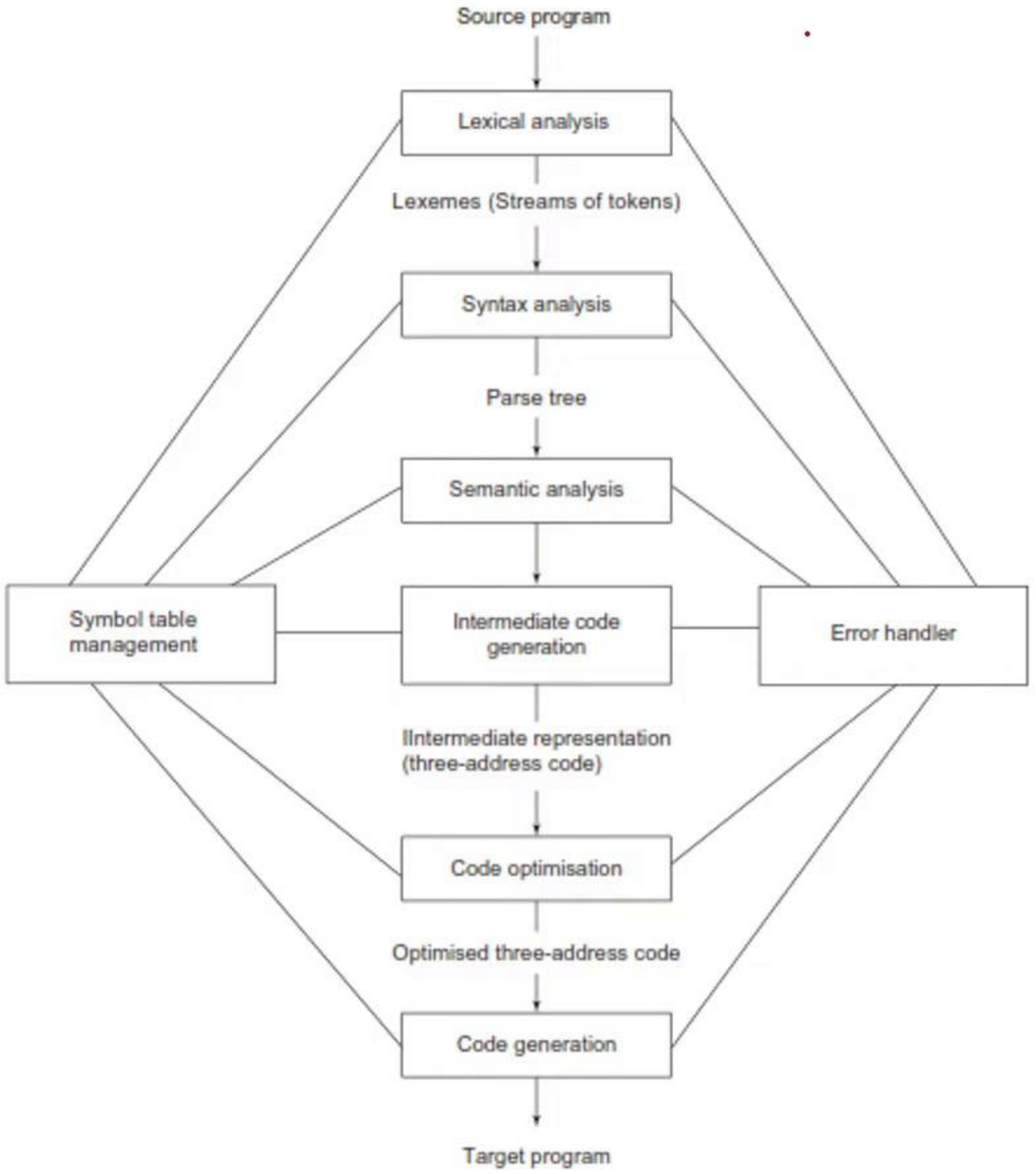
PHASES OF COMPILER

Objectives

- To introduce the phases of compilation and relate them to language constructs

Phases of Compiler



Lexical Analysis

- Interface of the compiler to the outside world
- Source Program → Lexical analyser→ Stream of tokens
- Scans input source program, identifies valid words of the language.
- Also, removes extra white spaces, comments, etc. from the program
- Tokens- logically cohesive sequence of characters
- Implemented as finite automata.
- Common examples of tokens are:
  - Keywords
  - Operators
  - Identifiers
  - Symbols
  - Constants
  - Strings

Lexical Analysis

C code which prints numbers from 1 to 10,

```
void main( )
{
int count;
for(count = 1; count <= 10; count ++)
printf("%d", count);
printf("\n");
}
```

34 tokens will be identified as follows:

```
void        main        (          )
{
int         count       ;
for   (     count       =          1   ;   count       <=   10   ;          count       ++
            )
printf  (   "%d"    ,           count       )          ;
printf  (     "\n“      )          ;
}
```

Lexical Analysis

- The symbol table stores

<token-type, attribute value> pairs for all the tokens identified.

- Consider the statement: int count = 1;

Token	Token type	Symbol table entries
int	keyword	<keyword, int>.
count	identifier	<id, pointer to symbol table entry of count>
=	operator	<assign_op, >
1	constant	<constant, 1>
;	symbol	<symbol, ;>