Name: A. Hasith
Reg no: 19A21470

* Assignment - 3 *

* Lexical Analyser

→ also called as scanner.

→ Reach the successive line.

→ It breaks into terms like identifier, Operator, delimiter.

→ Analyzer constructs symbol table.

→ The symbol table allocates memory.

How it works

Tools : LEX.

1. **Input processing** this stage involves, cleaning up input text send preparing it for lexical analysis. This may include remaining contents, whitespace and other non-essential characters from input text.

a. **Tokenisation :** This is the process of breaking input text into sequence of tokens. This is usually done by matching characters in input text against let of patterns.

→ The lexes checks that tool taken is valid according to makes of programming language.

* Syntactic Analyser

→ The Syntatic analyser refers to the expression, statement, declaration identified → It is aided by formal grammar by programming language.

→ Also called as parsing.

* **purpose :** Is to draw exact meaning → checks text for meaningfulness comparing to rules of grammar.

* Create parse tree on abstract syntax force of source code which is hierarchical representation of source code that reflects grammatical structure of program.

Features:

* Syntax tree. Context free Grammar
* Top-down & bottom-up passing
* Error detection.
* Intermediate code generation
* optimization.

Advantages :

⇒ Structural Validation.
⇒ Improved Code Generation.

* Semantic Analyzer:

→ It is also called as phase bridge
→ Analysis phase of syntax
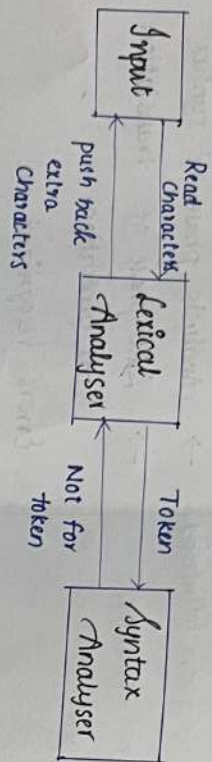→ Last phase of translation is code generation.

errors recognized are
○ Type mismatch
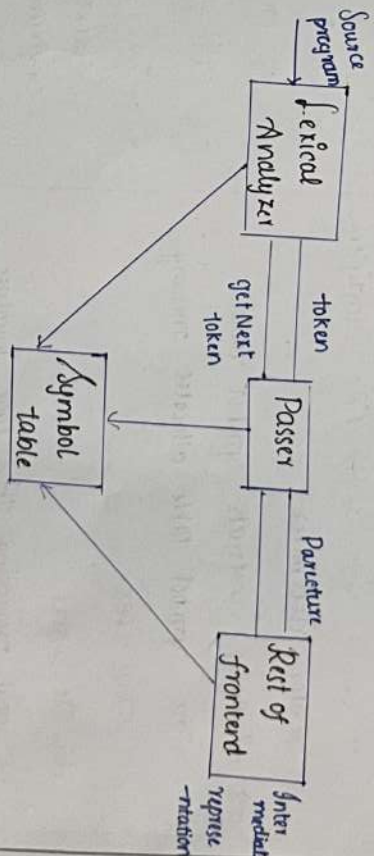○ Undeclared Variables.
○ Reserved identified Number.

Functions :

1. Type checking - Ensures that data types are used in a way consisted with there definition.

a. Label Checking - A program should contain label refrences.

3. Flow Control Check - keeps a check that contrast structures are used in a proper manner.

→ Static Semantic : There are checked at Compile time.

→ Dynamic Semantic: Different units of program like expression & Statements.

LEXICAL ANALYSER:

Input → Read Characters → Lexical Analyser → Token → Syntax Analyser

Lexical Analyser → push back extra Characters → Input

Syntax Analyser → Not for token → Lexical Analyser

SYNTATIC ANALYSER

Source program → Lexical Analyzer → token → Passer → Parseture → Rest of frontend → Inter mediate represe ntation

Lexical Analyzer → Symbol table

Passer → get Next token → Lexical Analyzer

Passer → Symbol table

Rest of frontend → Symbol table

SEMANTIC ANALYSER

Assignment Statement

Identifier Sum → Expression

Expression → Identifier Num 1 + Identifier Num 2