Date & Time of Examination - 01/12/2021 @ 9:30 AM

Examination Roll No. - 18312911011

Name of Program - B.Tech (IT & MI)

Semester / Year - VII Semester / IV year

Unique Paper Code - 911711

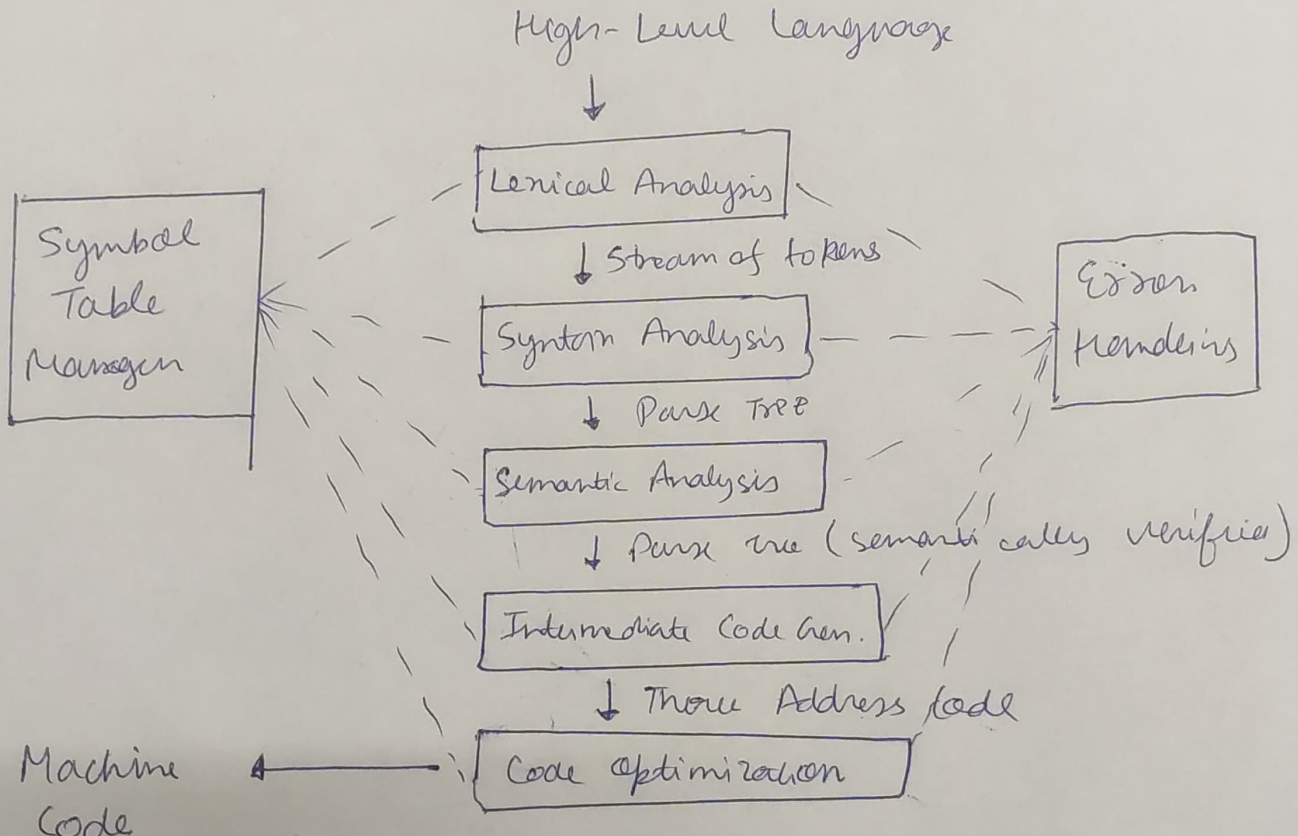
Title of Paper - Computer Language, design and Engineering

Q.1 Compiler operates in various phases, each phase transforms the source program from one representation to another.

Every phase takes inputs from its previous stage & feeds its output to the next phase of the compiler.

There are 6 phases of compiler which help in converting the high-level language to the machine code.

Phases are: (Structure)



1) Phase 1 : Lexical Analysis

- First Phase when compiler scans the source code
- Process can be left to right, character by character
- Group these characters into tokens
- Identifies token which is not part of the language

2) Phase 2 : Syntax Analysis

- Obtains tokens from lexical Analyser
- Checks if the expression is syntactically correct or not
- Reports all syntax errors
- Construct Parse tree

3) Phase 3 : Semantic Analysis

- Keeps in storing type information gathered & save it in symbol table or syntax tree
- Allows to perform type checking
- Semantic Error shown in case of type mismatch.
- Collects type information & checks for type compatibility
- Checks if the source language permits the operands or not.

4) Phase 4: Intermediate Code Generation

- Generated from semantic representation of the source program.
- Holds values computed during translation process
- Helps translate intermediate code into target language.
- Allows maintaining precedence ordering of the source language.
- Holds the correct number of operands of the instruction

5) Phase 5 : Code Optimization

- Trade-off between execution & compilation speed
- Improves running code of the target program
- Generates streamlined code still in intermediate representation
- Removing unreachable code & getting rid of unused variables
- Removing statements which are not altered from the loop

6) Phase 6 : Code Generation

- Gets input from code optimization phases
- Produces the page code or object code
- Allocates storage & Generate relocatable Machine Code

Ex:

Input $x = a + b * c$

↓ ①

Lexical Analyser — lex

↓ Convert to stream of tokens

$id = id + id * id$

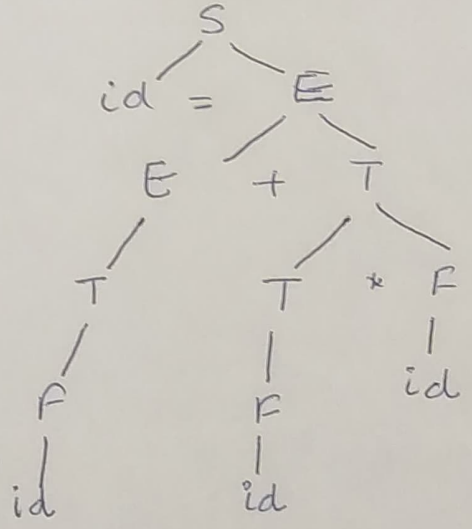
↓ ②

Syntax Analyser ← Yacc

$S \rightarrow id = E ;$
$E \Rightarrow E + T / T$
$T \rightarrow T * F \mid F$
$F \rightarrow id$

S - statement
E - Expression
T - Term
F - Factor

→ Syntax Error gets caught here



③ → Semantic Analyser

Frontend

↳ Left side should be variable
↳ Meaningfully gets the meaning of parse

↓ ④ (Parse Tree Semantically Correct)

ICG → Intermediate Code Generator

↓ ⑤

$t_1 = b * c ;$
$t_2 = a + t_1 ;$
$x = t_2$

⑥ → CO

$t_1 = b * c ;$
$x = a + t_2$

TCG → Target Code Generator

⑧

Backend

mul	R_1, R_2	$a \rightarrow R_0$
add	R_1, R_2	$b \rightarrow R_1$
mov	R_2, X	$C \rightarrow R_2$