ANTRIKSH SHARMA 2007012202 1
BTECH CS-A 20-24

CC LAB 01 | LEX AND YACC

Q.) Write answers to all questions:

Q.1) What are compilers and assemblers ?

A.) ① Compilers: A compiler is a software program that takes the entire source code written in a high level language (eg. C, C++, Java, etc.) and translates it into an equivalent program in a low-level language, typically machine or assembly or intermidiate code.

② Assemblers: Assemblers translate assembly language code into machine code. Assembly is a low-level language, in which each operation resembles a specific machine instruction.

2007012202l

Q.2) Explain in detail, various phases of compilation process.

A) ⊙ Lexical Analysis : (Scanning). This phase reads the character streams and converts (breaks) into a stream of tokens.

⊙ Syntax Analysis : (Pausing) Takes token stream as input and checks if they conform to the grammar of the programming language. and generates an abstract syntax tree (AST)

⊙ Semantic Analysis : This phase checks if the program conforms to language's types system and other semantic rules

⊙ Intermidiate Code Generation :

An intermidiate representation of the code is generated which can be easily translated into machine-code

⊙ Optimization: This phase applies various optimization techniques to intermidiate code to improve its performance of the machine code.

20070122021

⊙ Code Generation: From the optimized intermidiate code, actual machine code is generated that can be executed by the target hardware.

Q.3) How lexical & Syntax analysis work? What are the tools for that purpose.?

A.) ⊙ Lexical Analysis:

Also known as scanning, lexical analysis is the first step in the compilation proces.
It reads source code characters by character and groups them into meaningful tokens.
It uses regular expressions or finite automata to identify & categorize these tokens based on language's lexical grammar.

⊙ Syntax Analysis:

Also known as pasing, is the second phase of compilation proces. Takes sequence of lexems generated by lexical & arranges them into hierarchial structure called the abstract syntax tree

(AST) or parse tree.
Parser verifies the arrangement of tokens adheres to rules defined by language's syntax.

↳ Tools:

1.) lex & Flex: Generate lexical analyzers (scanners) based on regular expressions.

2.) YACC & Bison: Generate parsers based on formal grammar typically specified using BNF.

Q.4) How token, patterns and lexems differ?

A.) ⊙ A token is a group of characters having collective meaning: typically a word or punctuation mark, seperated by a lexical analyzer and passed to a parser

⊙ A lexeme is an actual character sequence forming a specific instance of a token, such as num.

⊙ Pattern: A rule that describes the set of strings associated to a token. Expressed as a regular expression & describing how a particular token can be formed.

Date
Page

20070122021

Q.5) Explain in detail, LEX and YACC specification.

A.) LEX

Lex is a lexical analyzer that generates C, C++ program that scans input text based on a formal description of regular expressions.

- Definition: Includes necessary C/C++ code declarations, regular expression definitions & other global configurations.
- Rules: List of regular expression along with corresponding C/C++ code that should execute when a particular regular expression is matched.

YACC (or Bison).

Parser generator that takes a formal context-free grammar description of a language and produces a C or C++ program that parses input text based on provided grammar rules.

- **Declaration:** Includes any necessary C/C++ code, type defⁿ, and external func defⁿ.
- **Grammar Rules:** Defines the context-free grammars of the language using BNF notation.
- **C code:** Includes C/C++ code snippets that are executed during parsing process.