CS4XX- INTRODUTION TO COMPILER THEORY

WEEK 2

Reading:

Chapter 3 from Principles of Compiler Design, Alfred V. Aho & Jeffrey D. Ullman

Objectives:

1. To finish the study about one-pass compiler
2. To introduce about the first phase of compiler: Lexical analyzer

Concepts:

1. One-pass compiler (continued) ------------------------------ 2 hour
2. Lexical analyzer -- 1 hour

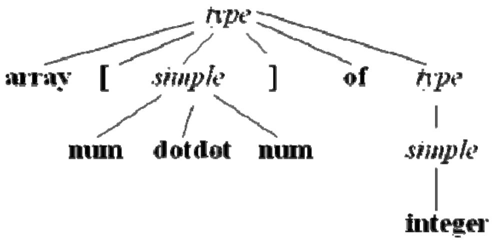
Outlines:

1. One-pass compiler
    a. Parsing
    b. Lexical analysis process
    c. A brief look at code generation
2. Lexical analyzer – introduction
    a. What does lexical analyzer do?
    b. How does it work?

1. One-pass compiler (continued)

    a. Parsing

    - Top-down parsing: Parse tree / derivation of a token string occur in a top down fashion.



        - Top-down process: Recursive Descent or Predictive Parsing: Parser Operates by Attempting to Match Tokens in the Input Stream

    - Predictive parsing: Designing a Predictive Parser

    b. Lexical analysis process

    - Input Token String Is Broken Down
    - White Space and Comments Are Filtered Out
    - Individual Tokens With Associated Values Are Identified
    - Symbol Table Is Initialized and Entries Are Constructed for Each “Appropriate” Token
    - More in-depth lexical analyzer introduced in next chapter

    c. A Brief Look at Code Generation: Focus on front-end process

    - Employ Statement Templates for Code Generation.
    - Each Template Characterizes the Translation
    - Different Templates for Each Major Programming Language Construct, if, while, procedure, etc.

2. Lexical analyzer – introduction (to be continued next week)

Basic Concepts & Regular Expressions

    a. What does a Lexical Analyzer do?

    b. How does it work?