Compiler Project

Yuhua Mai, Lei Li

Lexical Analysis - Due Jan 27

1. How you handle comments?

To handle comments or even nested comments, we set up a initial state <INITIAL> and a counter commentNum to keep track of the nested comments.

- Everytime the program encounters "/*", it goes into <COMMENT> state while commentNum increases by 1.

- Everytime the program meet a "*/", firstly commentNum decreases by 1, and then we peek its value:
    - 0 : it means the comment has ended, the program goes back to <INITIAL> state.
    - not 0 : the program continue in <COMMENT> state.
- Otherwise, stay in <COMMENT> state.

2. How you handle strings?

To handle strings, we introduce two states (<STRING_STATE> and <BACKSLASH_STATE>), a value stringVal to hold strings.

- Everytime the program encounters a " in <INITIAL> state, it sets stringVal as an empty string, and then enter <STRING_STATE>.
- In <STRING_STATE>, if a \ is encounted, the program enters <BACKSLASH_STATE>; if a " is encounted, the program exits back to <INITIAL> state and converts the string literal to Tokens; otherwise, append the text at the back of stringVal.
- In <BACKSLASH_STATE>, if regular expression [A-Za-z\"\\]|{digit3} is encounted, peek yytext:
    - if yytext = [bnt\"\\], append "\yytext" at the back of stringVal.
    - if yytext is a string of size 3, convert it to integer. if in the range of [0, 255], append it at the back of stringVal and continue; otherwise print error message.
    - otherwise print error message that the character is illegal.

return back to <STRING_STATE> and continue.

- In <BACKSLASH_STATE>, if regular expression [\t\n]*\\ is encounted, just ignore, and return back to <STRING_STATE>.

3. Error handling

Print error message and continue lexing from the next character.

4. End-of-file handling

Convert EOF to a Token.

Parsing - Due Feb 10

1. How do we deal with shift-reduce conflicts?

2. Notes

How to use prabsyn.sml? PrintAbsyn.print(TextIO.openOut "testPrint.txt", Parse.parse"test/test1.tig");

Semantic Analysis - Due Feb 28

Frame Analysis and Intermediate Representation - Due Mar 19

Instruction Selection - Due Mar 31

Register Allocation - Due Apr 14

Working compiler, produces assembly - Due Apr 22