Lab 1

COEN 175 Compilers

Contact Info

Justin Cole

Email: Jcole@scu.edu

Office Hours: Tuesday 12:00-1:00PM / Thursday 12:30PM - 1:30PM

Orion Sun

Email: Osun@scu.edu

Office Hours: Tuesday 1:00-2:00PM / Wednesday 2:30-3:30 PM

Overview For Lab 1

Goal

- Write lexical analyzer for a simple c program that will print out what tokens it reads

Submission

- Submit a tarball of your cpp and make files
- Due Date: Sunday January 9th

Main Flow

- Main file will be lexer.cpp
- Write a while loop in main to read each character (can be a separate function)
- Main Loop:
    - Either look at currently read character (can be simpler to write) or peek next character
    - Identifies what it is and cout or ignores accordingly
    - May utilize peek or read subsequent characters in order to identify certain tokens
    - Clear current character and subsequent characters that are part of the identified token (and possibly set next character to be read)
- Example:
    - Int x;
- Cout:
    - keyword:int
    - identifier:x
    - operator:;

Reading Input Functions

- Included in <iostream>
- cin.get() - reads and removes the character at the top of the input stream
- cin.peek() - reads but does not remove character at top of input stream
- cin.putback() - puts back character into stream
- cin.eof() - returns true if no more characters to read
    - Should be used in your main while loop

Type Functions

Type functions included in <cctype> will be very useful for identifying

- isalpha() - returns true if character is a letter
- isspace() - returns true if character is space
- isalnum() - returns true if character is alphanumeric
- isdigit() - returns true if character is digit

Tips

- Most tokens can be determined from the first character
- Only will be given lexically correct programs so you can always read the character after an escape character
- The types are defined as regular expressions, but regular expressions will not be of much use to you in this program
- Spaces and comments do NOT get printed out
- Put single or double quotes when you output strings and characters
- Use checksub.sh (requires a makefile) to check your submission against the examples, that is what we will be using (along with other tests) to grade your programs
- READ THE LEXICAL RULES CAREFULLY

Running CHECKSUB.sh

- Create a tarball of your submission
    - Includes your .cpp files and your make file
- Create a tarball of your test files
    - Some example files are provided on Camino
- Run ./CHECKSUB.sh <submission tar> <test files tar>