ECGR 3183 - Fall 2004: Lab 6

Parsing – Due 12/8/04

Learning Objectives

This lab will require you to write C code to parse a string that is entered in from the keyboard. It will determine if the string is a valid "C equation".

Prerequisites

You should be familiar with the following concepts:

- With basic programming skills and also be familiar with C language.
- Completed Lab 5 successfully. Have a good idea of the requirements of Lab 6.

Prelab Assignment

Write a flowchart of the lab solution. You must turn this in with the lab report.

Laboratory Assignment

You are required to write code which will print a message on the screen to enter a line of C code (equation only) from the keyboard. Your program will assess if the line is valid, and print a response. Cease the program when the enter key is pressed without any other character. Negative numbers will not be entered. An example of the console display would be (***bold/italic*** characters are user inputs):

Welcome to the ECGR 3183 parser. This program will assess a line of C code that you type in and tell you if it is valid.

Enter an equation (or type the enter key alone to end): ***a = j-k*4;***
This is a valid line of C code.

Enter an equation (or type the enter key alone to end): ***4 = j-k*4;***
This is an INVALID line of C code.

Enter an equation (or type the enter key alone to end): ***circumference = 2* radius*22/7;***
This is an INVALID line of C code.

Enter an equation (or type the enter key alone to end): ***circumfnce = 2* radius*22/7;***
This is a valid line of C code.

Enter an equation (or type the enter key alone to end): ***circumfnce = 2* radius15*22/7;***
This is an INVALID line of C code.

Enter an equation (or type the enter key alone to end): ***area = radius^2*22/7;***
This is an INVALID line of C code.

Enter an equation (or type the enter key alone to end): ***(enter key pressed)***
Thanks for using the ECGR 3183 parser!

Steps

1. Write a flowchart or state diagram of the lab.
2. Build your program slowly, testing along the way. Solve each requirement one at a time. Make sure comments are written as you progress.
3. Use the Unix cc or gcc application to compile and run your code.
4. Continue to build and test the program until all of the requirements have been met. Did we mention you should write your comments as you progress, not at the end?
5. Once all the requirements have been met ensure that everything works.
6. Demonstrate the working program to your TA or professor.
7. Finish lab write-up and turn in your report as a print out. Also, email the code to the lab TA.

Requirements

Req. 1 – The code generated is to be written in C language.
Req. 2 – The code is well commented and easy to follow.
Req. 3 – Your lab report should include the final code listing.
Req. 4 – When the program starts, print a welcome message. Then print the "enter a string" message.
Req. 5 – The general format of a valid equation is:

variable = *operand operation operand* [*operation operand* . . .];

where: *variable* is one to ten alphabetic character variable name, all in lower case.
= must follow the variable.
operand is either a one to ten alphabetic character variable name, all in lower case, or one to three digit positive integer.
operation is one of the arithmetic operations of + - * / %
[ ] the *operation/operand* pair inside is optional, and up to three more operations can be entered.
; a line must be ended by a semicolon.

Req. 6 – If a character is not listed above, it is invalid. This includes: capital letters, decimal points, parenthesis, etc.
Req. 7 – Only lines with the above format are valid. Other C statements are not valid, and variable names with embedded numbers are invalid.
Req. 8 – If you find a string is invalid, you do not need to parse the rest of the string.
Req. 9 – Continue to have users to input lines and parse them until the enter key is pressed alone.