1. Instructions

You will be continuing work with your group on this project. Your whole group will only need to submit one completed project. This can be done from any group members Blackboard account. Your solution must be uploaded to Blackboard by the deadline, read section 4 to learn what to submit and by when.

2. Project Objectives

The objective of this project is to have you become familiar with dynamically allocated memory in C and Scanning input strings to parse them into components. These components will be stored in a new programming construct called a struct in C.

3. Problem Statement

You will start your project with a proper design and test cases. That means you will need to create pseudocode and a table of all the tests you want to run for the program. This program will differ slightly in that you will have one additional table for the main function as well as all of the other functions in the program. The tests for the main function are called system tests and you should treat them slightly different than you do for functional tests. You should write tests that cover more cases of valid input and do not have any invalid input. You should be creative in coming up with these valid test cases for your program to cover all of the tricky cases you can think of.

25 pts. You must write functional tests using the same criteria as the previous assignment, and you will submit your pseudocode for each function, test tables, and test code along with your source code which is detailed in the following paragraphs. This assignment isn't broken into two phases so you will only turn in one assignment. **Please** do not wait until near the deadline to turn this in. If you do, you will not be able to complete it, engineering is an iterative process requiring time, and you cannot push this back.

75 pts. You will write a source code scanner, which reads a line from a source code file, prints that to the screen with the line number and then proceeds to move through the line character by character building tokens. Each token must be identified as what it is; a keyword (reserved word) in Pascal, a literal (which is an identifier, a number, or a string), or one of many special characters in Pascal.

I have provided some code for you on this project which you can use to finish the project. I have provided a table of Pascal reserved words in the Scanner.c file, so you should refer to this if your token is a reserved word. I have also provided an enumeration of all the different tokens in Pascal in common.h. You should refer to this when you have determined what type of token you are looking at. Finally, I have a corresponding table in print.c that has the exact

same types of token codes as the enumeration in common.h. You should use this so you can print the proper token code once you have found the proper token.

Your program should consist of six files (however if you come up with a better design please take what you want from mine, and do yours instead). Scanner contains several function to help identify all the tokens. get_char() is a function that checks the current state of the source_buffer and gets a new line of code if it is at the end of a line. If it sees a Pascal Comment it skips the comment (a pascal comment is anything between '{' and '}').

Scanner is activated when main calls get_token(). get_token() is a function that first skips past any blanks(the character ' '), then with the first character in the token it decides to get_word()(if it is a letter), get_number()(if it is a number), get_string()(if it is a quote '), if it is the end of the file, or else it must have a special so it can get_special(). It determines the type of token it has then it sets a "code" to a token code type, and it puts the token into "token_string". If the type is a literal then it creates a Literal object and creates the proper literal type and data. A literal is either a string or a number (a literal usually appears on the right hand side of the equals sign '=').

get_word() is a function that finds a keyword or identifier in pascal. A keyword can have letters or numbers in them. Once it finds the end of a keyword or identifier, it backs up the character and makes the word all lowercase. It then checks is this word a reserved word? You can use the arrays provided in the source code I gave out to find out if something is a reserved word or not. If not then this token is an identifier.

get_number() is a function that finds two types of numbers; integers and real numbers. If it sees an integer it creates a new literal and stores the proper type and data in that literal. A real number is any number that has the character period '.' in it and/or the character 'e' in it. After the 'e' there can be a negative sign, and the 'e' stands for exponent. If get_number() sees a real number it creates a new literal and stores the data and type in that literal. Finally, the get_number() method stores the type of token it has. The "token_string" does not need to be filled out since the literal has the proper data.

get_string() is a method that finds a string token. A string is between a single quote '\'' and another single quote '\''. The get_string() method creates a new literal object and stores the proper literal type and data. Finally the get_string() method sets the token "code" to the correct token code type.

get_special() is a method that finds a special character in Pascal. If you would like to see a list of these special characters, you can look at the symbol_strings array in the print.c file. The fifth through the twenty fifth elements are all the special characters get_special() looks for. Finally, get_special() will set "token_string" with the special characters and set the "code" to the correct type.

Once the token is all set, a token struct must be dynamically allocated (actually this can be done before parsing the source code if you prefer), and returned to main. Main will store the token in a linked list, and then send the token to print.c to print the token and/or literal if one was created. print_token() will print the token on its own line and will symbol_string value followed by literal value (if it's a literal) or a symbol or keyword. Once the token has been printed, then main will then ask the scanner to get the next token. This process will continue until the token is a period '.', in Pascal a period token signifies the end of a program.

4 What to Submit for Grading and Assignment Deadline

The project will be due Thursday March 20th at **11:59:59PM.** You will need to include a table clearly on your document that contains each team member's name. In that table you will place a number 0, 1, or 2 next to the name. Please see the class announcement I made on Blackboard for what this number system refers to. **Failure to include the table with the team members names and a number next to their name results in all team members receiving 50% off for the project**. You should include your design document(s) and the URL of your Repository so I can go to it, and look at how everyone contributed. All of this information should be in a pdf document. A portion of your grade will be determined by everyone contributing something to the repository. Please see project one for more information on this. You should also include your commented source code, and your test code files. Please put all files together as a tar file.

You will need to include a URL of your repository, and a list of each team members name and their corresponding repository ID. I will be going to your site and checking the actual contribution of each team member. If I see everyone is contributing then all team members will receive equal points. If I see some team members have contributed significantly less than the others I will further investigate and deduct a percentage of that team members points based on their contribution. If you are worried that one team member didn't contribute much to the repository, but they were instrumental in the project, then you can comment on this when you submit the assignment.

Please put your group members names at the top of every document you submit including code (this must be a comment). The file name of your tarball should be cse220_lab3.tar.
Please see project one for more information on the penalty for not submitting a pdf file for documents that must be in pdf format.