Documentation for LexDriver

Nicolas Brodeur-Champagne 27043651

Section 1. Lexical specifications: Identify the lexical specification, expressed as regular expressions, that

you used in the design of the lexical analyzer. Highlight any changes that you may have applied

to the original lexical specifications and justify your changes

To find any language reserved keyword and comments: A static dictionary is used for <key,value> pairs.

To find ids: "^([a-z]|[A-Z])([a-z]|[A-Z]|[0-9]|\_)\*$"

To find intnum: equals 0 or "^[1-9][0-9]\*$"

To find float: "^([1-9][0-9]\*|0)(\.([0-9]\*[1-9]))?e([+|-])?(0|[1-9][0-9]\*)$" or

"^([1-9][0-9]\*|0)\.([0-9]\*[1-9])$"

\*All documentation occurred after coding.

Section 2. Finite state automaton: Finite state machine diagram describing the operation of your lexical

analyzer.

Start(word) -> Is Dictionary key? -> No, Start with a letter? -> No, Starts with Number -> No, Return error

->Yes, return token ->Yes, is id? ->Yes, intnum or float?

->No, return error ->No, return error

->Yes, return token ->Yes, return token

Section 3. Design: Give a brief overview of the overall structure of your solution, as well as a brief

description of the role of each component of your implementation.

Program .exe: Takes first arg as a file, if doesn’t end with ‘.src’ stop application, if the path of file can’t be found stop application. If good file, for every word of the file use lexical analyzer to obtain tokens, each token gets printed to either and token and error file.

LexicalAnalyzer: accepts 2 arguments the word to parse and the line number the word was found. Runs regex on the word to output and token, that may contain an error and its type.

TokenData: Datatype that holds token type, token lexeme, and the line number the word was found on.

Section 4. Use of tools: Identify all the tools/libraries/techniques that you have used in your analysis or

implementation and justify why you have used these particular ones as opposed to others.

C# language

Visual Studio

Using:

using System;

using System.Collections.Generic;

using System.IO;

using System.Text.RegularExpressions;