CMPUT 415 - Project Checkpoint 1

Program Documentation

Mike Bujold

Dan Chui

James Osgood

Paul Vandermeer

October 5, 2012

The man page is available by typing *make man*

Introduction

The goals for this checkpoint are to have a functional compiler 'shell'. That is, it must be able to parse a program listing and report on any errors in the program in a way that is meaningful to the user so that corrective action may be taken. It must also recover from each error in such a manner that further reporting of errors is possible, up until the end of the program listing.

Handling of Lexical Units

Our lexer parses tokens in the following order: first, all single and multi-line comments are parsed, followed by reserved *pal* keywords, then numbers and variable names. Relational operators and other lexical units are parsed last. Any remaining unparsed tokens are presumed invalid, and the grammar handles reporting the error accordingly.

Syntax Error Reporting & Recovery

We created myerror.c to catch all the errors. When an error is found, it is added to a linked list. As soon as we finish parsing a line, we output that list, clear it, and move onto the next line.

Testing

To facilitate large scale testing and efficient reading of listing files, we created a python script to run the compiler on multiple tests and pipe the resulting output to a testing log.

Problems

Since we do not strip out comments when listing the parsed file, the listing file may not be further parsable (that is, the listing may cause additional errors if run through the compiler). This is because of block comments, in a line where an error occurred, whose closed brackets interrupt the block comment of the error message added to the listing.