CISC / CMPE 458 Course Project Winter 2020

Phase 1. Scanner/Screener

(Due 4:30pm Wednesday February 5)

J.R. Cordy – January 2020

In this phase you will undertake the modifications to the Scanner/Screener phase of the PT Pascal compiler to turn it into a Scanner/Screener for Qust. The hardest part of this phase is getting used to using Linux, PT Pascal and S/SL. The actual changes are relatively easy once you figure out how everything works.

Because the PT scanner/screener is run as a co-routine with the parser all in one pass (called *parser.out*), it's necessary to take some special steps to test the result of your modifications. Instead of simply building the parser using the *make* command, you will build it using the command *make scanner*, which will make a special version of *parser.out* that runs the scanner/screener only. You will use the *–t1* and *–o1* flags of the *ptc* command to run your scanner/screener, and the *ssltrace* command to print out the tokens it recognizes – ask your TA for details.

Suggestions for Implementing Phase 1

The modification of the Scanner/Screener to implement Qust should be relatively straightforward, so no particular implementation hints will be given for this phase. You will need to change the files *scan.ssl, parser.pt,* and *stdIdentifiers* in the PT Pascal *parser* subdirectory. The following is a list of the changes that are likely to be required.

Keywords

Update the keywords and predefined identifiers in *stdIdentifiers* for Qust.
Remove the old PT keywords :

div mod or and not then end until do array
program var procedure begin case repeat

Add the new Qust keywords :

pub mod main let mut fn loop break match

Replace the old PT Pascal predefined identifiers :

integer char boolean write writeln

With the corresponding new Qust predefined identifiers :

int str bool print println

Character Classes

Add new Qust input character classes (e.g., *lQuestionMark*) for the characters question mark ("?"), slash ("/"), percent ("%"), ampersand ("&"), or bar ("|"), exclamation ("!") and underscore ("_") in *scan.ssl*. Reuse the existing *lQuote* character class to refer to the Qust double quote (") instead of the PT Pascal single quote ('). Don't forget to update the initialization of the character class map for all these characters in the *Initialize* procedure of *parser.pt.*

Syntax Tokens

Update the syntax token definitions in *scan.ssl* to remove the old PT keywords (see above), and add the new Qust syntax tokens (e.g., *pMod*) for the new Qust keywords **pub, mod, main, let, mut, fn, loop, break** and **match.** Add the new Qust syntax tokens (e.g., pLeftBrace) for the new Qust symbols left brace ("{"), right brace ("}"), plus equals ("+="), minus equals ("−="), slash ("/"), percent ("%"), double equals ("=="), double ampersand ("&&"), double or bar ("||"), or bar ("|"), not ("!"), underscore ("_"), question mark ("?") and equal greater ("=>"), replacing or removing the old PT symbols that are no longer used in Qust.

String Literals

Replace the scanning and screening of PT single-quoted char literals (e.g. 'hi there') with the scanning and screening of Qust double quoted string literals (e.g. "hi there"). Reuse the old PT character class *lQuote* to represent the double quote character and reuse the old PT char literal compound token *pStringLiteral* to represent the new Qust double-quoted strings. Remember that in Qust, the null string ("") is allowed.

Comments

Replace the scanning of PT { } and (* *) comments with the scanning of Qust // to end of line and /* */ comments. Be careful to preserve the right number of *pNewLine* tokens in the scanned output when skipping // comments by always emitting a *pNewLine* token for the ending newline of the comment.

What to Hand In

Submission of your assignment will be electronic - ask your TA for details. Hand in the following :

(1) Documentation of the modifications you have made to the scanner/screener, including a list of the changes made to the S/SL source, a description of the changes to the token streams and tables, and an explanation of any new error signals. The documentation should be easily understandable to someone who already knows the basics of the PT Pascal compiler.

(2) A complete copy of your Qust compiler so far with the changed scanner/screener source in the parser subdirectory, with changes indicated by appropriate comments. Your compiler should be ready for us to compile and run using the "make scanner" command to compile your source so far.

(3) A suite of test inputs (small Qust programs) designed to demonstrate the (partial) correctness of your scanner/screener by forcing it through every new or modified logic path in the S/SL source at least once. The test suite should be accompanied by comments indicating the purpose of each test input. Modifying the old PT test suite in the test/ subdirectory of the PT Pascal source to be Qust programs may help as a start.

The purpose of what you hand in is to clearly indicate exactly what you have done, provide enough information about the changes to allow further maintenance of the new scanner/screener by someone who already knows the PT Pascal compiler, and to convince them that the changes you have made are complete and correct according to the specification of Qust.

Details of exact expectations and electronic submission will be given in tutorial.

See the handout *"CISC 458 Project Requirements"* for marking criteria and general expectations on project results.