This fold including 10 files as following:

(1) P2.java

(2) sym.java

(3) ErrMsg.java

(4) Makefile

(5) YES.jlex

(6) allTokens.in

(7) allError.in

(8) allError.txt

(9) chekLineCharNum.in

(10) checkLineCharNum.out

- The main class is P2.java, which contains the check for both legal and illegal tokens with corresponding error report, and line and char num checks.
- Both sym.java and ErrMsg.java are provided online, with no changes at all.
- To invoke the whole project by Makefile, including following instructions

- $ make
- $ make test

For makefile: Firstly, we need set $CLASSPATH as:

.:~/cs536/local_setup/deps_src:~/cs536/local_setup/java-cup-src/classes

Then run instruction $ make. It would compile YES.lex file and create the DFA transition table and output lexical analyzer code.

Then run instruction $ make test. It would compile P2.java and all diff instructions to check the correctness of our jlex file. If no diff errors report, it means that the program runs correctly for all of my test cases.

If you want to clean the corresponding files created during compiling

- $ make clean
- $ make cleantest

- allTokens.in provides test cases for all legal patterns allowed. After running diff instructions on allTokens.in and created file allTokens.out, we could check whether we have correct regular expression in YES.jlex
- allError.in provides test cases for different error patterns with warning and error report.

allError.txt provides the correct error log report. It could be compared with the output file allError.err after running diff instruction to check its correctness. I have redirect the system error log to be printed in allError.err file.

- checkLineCharNum.in is a file with different tokens. checkLineCharNum.txt provides the correct information of line num and char num of the checkLineCharNum.in file. After running diff instruction on checkLineCharNum.txt and checkLineCharNum.out, it proves that the line num and char num count are correct.