CS536 SP21 P2

Contributers

Yuting Yan, Yusen Liu

Run

```
$ cd $WORKING_DIR
$ make
$ make test
```

Output

After executing `$ make test`, the expected output by shell should be as follows:

```
java -cp ./deps:. P2 2> errors.out
diff testAllTokens.out testAllTokensExpect.out
diff testIDTokens.out testIDTokensExpect.out
diff testIntLitTokens.out testIntLitTokensExpect.out
diff testStrLitTokens.out testStrLitTokensExpect.out
diff testEOF.out testEOFExpect.out
diff testCommentEOF.out testCommentEOFExpect.out
diff testBadStrLitEOF.out testBadStrLitEOFExpect.out
diff errors.out errorsExpect.out
```

The output from the shell means that the real output of the test cases match our expected output.

Files included

- `cminusminus.jlex`: The JLex specification that defines the behavior of our scanner.
- `sym.java`: Token definitions that will eventually be generated by the parser generator.
- `ErrMsg.java`: Used to print error and warning messages.
- `P2.java`: Contains the main program that tests the scanner.
- `Makefile`: A Makefile that uses JLex to create a scanner, and also makes P2.class.

Test cases:

- `testAllTokens.in`: Test the reserved words, one- or two-character symbols and their combinations. Also includs comments and illegal characters.
- `testAllTokensExpect.out`: The expected output of testing `testAllTokens.in`.
- `testIDTokens.in`: Test the identifiers.
- `testIDTokensExpect.out`: The expected output of testing `testIDTokens.in`.
- `testIntLitTokens.in`: Test the integer literals including bad integer literals.
- `testIntLitTokensExpect.out`: The expected output of testing `testIntLitTokens.in`.
- `testStrLitTokens.in`: test the string literals including unterminated string literals and bad string literals.
- `testStrLitTokensExpect.out`: the expected output of testing `testStrLitTokens.in`.

- `testEOF.txt`: test that your scanner correctly handles an unterminated string literal with end-of-file before the closing quote.
- `testEOFExpect.out`: the expected output of testing `testEOF.txt`.
- `testCommentEOF.txt`: test that your scanner correctly handles an unterminated string literal with end-of-file before the closing quote.
- `testCommentEOFExpect.out`: the expected output of testing `testCommentEOF.txt`.
- `testBadStrLitEOF.txt`: test that your scanner correctly handles an unterminated string literal with end-of-file before the closing quote.
- `testBadStrLitEOFExpect.out`: the expected output of testing `testBadStrLitEOF.txt`.
- `errorsExpect.out`: the expected output of error messages.

All the test cases cover most of the possible cases.

P2.java

This program is to be used to test the C-- scanner. This version is set up to test all tokens including input that causes errors, character numbers, values associated with tokens.

- `private static void testAllTokens(String fileIn, String fileOut) throws IOException`: this method takes two parameters, of which the first is the test case to read-in, and `fileOut` is the output file read by our scanner.
- `main`: this function calls `testAllTokens()` 7 times that covers all the test cases we provide (described in the previous section). Each time after the function call, `CharNum.num` is set to be 1.