

## Parser

For this milestone you will extend your program from the previous milestone to parse Mini-Java programs. I expect that you will use a parser generator tool to construct your parser. I leave the choice of tool to you.

Your parser will take the name of a text file on disk as an argument. The program will parse the file according to the syntax given in the Term Project Description. Your program will output the productions matched in order – one per line.

Note that you will need to rewrite the grammar to properly encode precedence and associativity. Precisely how you do this will depend on the parser generator tool you have chosen. Because you must rewrite the grammar, I cannot precisely specify the expected output. I give an example below.

Consider the following input file:

```
/** This is a test. */
class Test {
    public static void main (String[] args) {
        System.out.println(2 + 13); // cool
    }
}
```

A top-down parser might output something like:

```
<Program> ::= <MainClassDecl>
<MainClassDecl> ::= class ID { public static void main (String[ ] ID) {
<StmtList> } }
<StmtList> ::= <Stmt> <StmtList>
<Stmt> ::= System.out.println(<Expr>);
<Expr> ::= ...
... // intentionally omitted to avoid giving away too much
<StmtList> ::= epsilon
```

A bottom-up parser might approximately reverse this order.

## DELIVERABLES

### Sample Input

You must submit 4 sample input files, **by class time at least three days before** the milestone deadline. *Three of the sample inputs should be valid MiniJava programs. The fourth should include at least one syntax error.* These sample input files should be wrapped in a zip or rar archive and uploaded to the drop box on Angel. Alternatively, you may commit the sample in-

put files to your Subversion repository and let me know where to look for them. Please limit your sample inputs to no more than 250 lines and no more than 80 characters per line. Note that you do not have to submit sample output at this time. You may wish to check your sample input by running it through a Java compiler, though be careful – MiniJava is a strict subset of Java.

I will select one sample input from each team and give the set of these to every team at least two days before the milestone deadline. I will give the remaining sample inputs to every team at class time on the milestone deadline. (We will generally not have class on milestone days. The sample inputs will be posted on Angel.)

### Sample Output and Code

By midnight on the milestone deadline you must upload to the dropbox on Angel a zip or rar archive containing:

- the output of your program for each sample input, one output file per input file, with the same name as the input file but the extension changed to “.out”
- the source code for your program

Alternatively, you may commit the sample output and source code to your Subversion repository and let me know where to look for it.

### Grading

Your grade will be based on whether you submit the required sample inputs and the percentage of the sample input files that your program processes correctly.

I'll award bonus points for sample inputs that expose bugs in other teams' programs.