Objectives.

1. Support multiline comments.

2. Support additional tokens (reserved words, operators, and separators).

3. Support `long` and `double` literals.

In this project, you will only be updating the hand-crafted scanner, which means that the only program files you will be modifying under `$j/j--/src/jminusminus` are `TokenInfo.java` and `Scanner.java`. To compile (just scan for now) your *j--* programs, you need to run the `j--` command as follows:

```
$ $j/j--/bin/j-- -t P.java
```

which only scans `P.java` and prints the tokens in the program along with the line number where each token appears.

Problem 1. (*Multiline Comment*) Add support for multiline comment, where all the text from the ASCII characters `/*` to the ASCII characters `*/` is ignored.

```
$ $j/j--/bin/j-- -t tests/MultiLineComment.java
5        : public = public
5        : class = class
5        : <IDENTIFIER> = MultiLineComment
5        : { = {
9        : public = public
9        : static = static
9        : void = void
9        : <IDENTIFIER> = main
9        : ( = (
9        : <IDENTIFIER> = String
9        : [ = [
9        : ] = ]
9        : <IDENTIFIER> = args
9        : ) = )
9        : { = {
13       : } = }
14       : } = }
15       : <EOF> = <EOF>
```

Problem 2. (*Reserved Words*) Add support for the following reserved words.

```
break       case        catch
continue    default     do
double      final       finally
for         implements  interface
long        switch      throw
throws      try
```

```
$ $j/j--/bin/j-- -t tests/ReservedWords.java
1        : break = break
1        : case = case
1        : catch = catch
2        : continue = continue
2        : default = default
2        : do = do
3        : double = double
3        : final = final
3        : finally = finally
4        : for = for
4        : implements = implements
4        : interface = interface
5        : long = long
5        : switch = switch
5        : throw = throw
6        : throws = throws
6        : try = try
7        : <EOF> = <EOF>
```

Problem 3. (*Operators*) Add support for the following operators.

```
?       ~         !=        /         /=
-=      --        *=        %         %=
>>      >>=       >>>       >>>=      >=
<<      <<=       <         ^         ^=
|       |=        ||        &         &=
```

```
$ $j/j--/bin/j-- -t tests/Operators.java
1          : ? = ?
1          : ~ = ~
1          : != = !=
1          : / = /
1          : /= = /=
2          : -= = -=
2          : -- = --
2          : *= = *=
2          : % = %
2          : %= = %=
3          : >> = >>
3          : >>= = >>=
3          : >>> = >>>
3          : >>>= = >>>=
3          : >= = >=
4          : << = <<
4          : <<= = <<=
4          : < = <
4          : ^ = ^
4          : ^= = ^=
5          : | = |
5          : |= = |=
5          : || = ||
5          : & = &
5          : &= = &=
6          : <EOF> = <EOF>
```

Problem 4. (*Separators*) Add support for the separator : (colon).

```
$ $j/j--/bin/j-- -t tests/Separators.java
1          : ; = ;
2          : : = :
3          : , = ,
4          : . = .
5          : [ = [
5          : { = {
5          : ( = (
5          : ) = )
5          : } = }
5          : ] = ]
6          : <EOF> = <EOF>
```

Problem 5. (*Literals*) Add support for (just decimal for now) long and double literals.

```
<int_literal> = 0 | (1-9) {0-9} // decimal

<long_literal> = <int_literal> (l | L)

<digits> = (0-9) {0-9}

<exponent> = (e | E) [(+ | -)] <digits>

<suffix> = d | D

<double_literal> = <digits> . [<digits>] [<exponent>] [<suffix>]
                 | . <digits> [<exponent>] [<suffix>]
                 | <digits> <exponent> [<suffix>]
                 | <digits> [<exponent>] <suffix>
```

```
$ $j/j--/bin/j-- -t tests/Literals.java
1         : <INT_LITERAL> = 0
1         : <INT_LITERAL> = 2
1         : <INT_LITERAL> = 372
2         : <LONG_LITERAL> = 1996l
2         : <LONG_LITERAL> = 777L
2         : <DOUBLE_LITERAL> = .3D
3         : <DOUBLE_LITERAL> = 3.14
3         : <DOUBLE_LITERAL> = 6.022137e+23
3         : <DOUBLE_LITERAL> = 1e-9d
4         : <EOF> = <EOF>
```

Files to Submit

1. `j--.zip` (*j--* source tree as a single zip file)

2. `report.txt` (project report)

Before you submit:

- Make sure you create the zip file `j--.zip` such that it only includes the source files and not the binaries, which can be done on the terminal as follows:

```
$ cd $j/j--
$ ant clean
$ cd ..
$ tar -cvf j--.tar j--/*
$ gzip j--.tar
```

- Make sure your report isn't too verbose, doesn't contain lines that exceed 80 characters, and doesn't contain spelling/grammatical mistakes