Objectives.

1. Support multiline comment.
2. Support long and double basic types.
3. Support operators.
4. Support conditional expression and switch statement.
5. Support do-while and for statements.
6. Support exception handlers.
7. Support interface type declaration.

In this project you will only modify the JavaCC specification file `$j/j--/src/jminusminus/j--.jj` for *j*-- to add more Java tokens and programming constructs to the *j*-- language. In the first part, you will modify the scanner section of the `j--.jj` file to support the Java tokens that you handled as part of Project 2 (Scanning). In the second part, you will modify the parser section of the file to support the Java programming constructs that you handled as part of Project 3 (Parsing). To compile the *j*-- compiler with the JavaCC front-end, ie, with the scanner and parser generated by JavaCC, run the following command:

```
$ ant clean javacc compileJavaCC jar
```

Part I: Additions to JavaCC Scanner

To scan your *j*-- programs using the JavaCC scanner, you need to run the `javaccj--` command as follows:

```
$ $j/j--/bin/javaccj-- -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/javaccj-- -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/javaccj-- -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/javaccj-- -t tests/Operators1.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/javaccj-- -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/javaccj-- -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>
```

Part II: Additions to JavaCC Parser

To parse your *j--* programs using the JavaCC parser, you need to run the `javaccj--` command as follows:

```
$ $j/j--/bin/javaccj-- -p P.java
```

which will only parse `P.java` and print the AST for the program in XML format.

Note.

1. Consult Appendix C of our text for the grammar (ie, formal specification) for each new construct you will be supporting in *j--*.

2. The AST shown (as XML) for each problem is only a suggestion as to what the AST ought to look like once the syntactic constructs for that problem are implemented in *j--*. You are *not* expected to produce exactly the same AST, but just something similar. The autograder will not match your AST against ours for correctness, but instead will test if your parser parses our pass tests without errors and our fail tests with suitable error messages.

Problem 6. (*Long and Double Basic Types*) Add support for the `long` and `double` basic types.

```
$ $j/j--/bin/javaccj-- -p tests/BasicTypes.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/BasicTypes.java"/>
  <Imports>
    <Import name="java.lang.Double"/>
    <Import name="java.lang.Long"/>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="5" name="BasicTypes" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="6" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="6" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="6">
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="7" name="a" type="double">
                  <Initializer>
                    <JMessageExpression line="7" name="parseDouble">
                      <Arguments>
                        <Argument>
                          <JArrayExpression>
                            <TheArray>
                              <JVariable name="args"/>
                            </TheArray>
                            <IndexExpression>
                              <JLiteralInt line="7" type="" value="0"/>
                            </IndexExpression>
                          </JArrayExpression>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="8" name="b" type="long">
                  <Initializer>
                    <JMessageExpression line="8" name="parseLong">
                      <Arguments>
                        <Argument>
                          <JArrayExpression>
                            <TheArray>
                              <JVariable name="args"/>
                            </TheArray>
                            <IndexExpression>
                              <JLiteralInt line="8" type="" value="1"/>
                            </IndexExpression>
                          </JArrayExpression>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JStatementExpression line="9">
              <JMessageExpression line="9" name="println">
                <Arguments>
```

```
                  <Argument>
                    <JBinaryExpression line="9" type="" operator="*">
                      <Lhs>
                        <JBinaryExpression line="9" type="" operator="*">
                          <Lhs>
                            <JLiteralDouble line="9" type="" value="3.14159D"/>
                          </Lhs>
                          <Rhs>
                            <JVariable name="a"/>
                          </Rhs>
                        </JBinaryExpression>
                      </Lhs>
                      <Rhs>
                        <JVariable name="a"/>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JStatementExpression line="10">
              <JMessageExpression line="10" name="println">
                <Arguments>
                  <Argument>
                    <JBinaryExpression line="10" type="" operator="*">
                      <Lhs>
                        <JLiteralLong line="10" type="" value="1729L"/>
                      </Lhs>
                      <Rhs>
                        <JBinaryExpression line="10" type="" operator="+">
                          <Lhs>
                            <JVariable name="b"/>
                          </Lhs>
                          <Rhs>
                            <JVariable name="b"/>
                          </Rhs>
                        </JBinaryExpression>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 7. (*Operators*) Add support for the following operators, obeying precedence rules (see Appendix C).

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

```
$ $j/j--/bin/javaccj-- -p tests/Operators2.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/Operators2.java"/>
  <Imports>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
```

```
    <JClassDeclaration line="3" name="Operators2" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="4" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="4" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="4">
            <JStatementExpression line="5">
              <JMessageExpression line="5" name="println">
                <Arguments>
                  <Argument>
                    <JBinaryExpression line="5" type="" operator="||">
                      <Lhs>
                        <JBinaryExpression line="5" type="" operator="&amp;&amp;">
                          <Lhs>
                            <JUnaryExpression line="5" type="" operator="!">
                              <Operand>
                                <JLiteralTrue line="5" type=""/>
                              </Operand>
                            </JUnaryExpression>
                          </Lhs>
                          <Rhs>
                            <JLiteralFalse line="5" type=""/>
                          </Rhs>
                        </JBinaryExpression>
                      </Lhs>
                      <Rhs>
                        <JLiteralFalse line="5" type=""/>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JStatementExpression line="6">
              <JMessageExpression line="6" name="println">
                <Arguments>
                  <Argument>
                    <JBinaryExpression line="6" type="" operator="!=">
                      <Lhs>
                        <JLiteralTrue line="6" type=""/>
                      </Lhs>
                      <Rhs>
                        <JLiteralFalse line="6" type=""/>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="7" name="x" type="int">
                  <Initializer>
                    <JLiteralInt line="7" type="" value="42"/>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JStatementExpression line="8">
              <JBinaryExpression line="8" type="" operator="-=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="8" type="" value="2"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="9">
              <JBinaryExpression line="9" type="" operator="*=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="9" type="" value="2"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="10">
              <JBinaryExpression line="10" type="" operator="/=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="10" type="" value="10"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="11">
              <JBinaryExpression line="11" type="" operator="%=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="11" type="" value="3"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="12">
              <JMessageExpression line="12" name="println">
                <Arguments>
                  <Argument>
                    <JUnaryExpression line="12" type="" operator="post++">
                      <Operand>
                        <JVariable name="x"/>
                      </Operand>
                    </JUnaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JStatementExpression line="13">
              <JMessageExpression line="13" name="println">
                <Arguments>
                  <Argument>
                    <JUnaryExpression line="13" type="" operator="--pre">
                      <Operand>
                        <JVariable name="x"/>
                      </Operand>
                    </JUnaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JStatementExpression line="14">
```

```
              <JBinaryExpression line="14" type="" operator="&gt;&gt;=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="14" type="" value="1"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="15">
              <JBinaryExpression line="15" type="" operator="&gt;&gt;&gt;=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="15" type="" value="2"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="16">
              <JBinaryExpression line="16" type="" operator="&lt;&lt;=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="16" type="" value="3"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="17">
              <JBinaryExpression line="17" type="" operator="^=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="17" type="" value="2"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="18">
              <JBinaryExpression line="18" type="" operator="|=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="18" type="" value="4"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="19">
              <JBinaryExpression line="19" type="" operator="&amp;=">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JLiteralInt line="19" type="" value="8"/>
                </Rhs>
              </JBinaryExpression>
            </JStatementExpression>
            <JStatementExpression line="20">
              <JMessageExpression line="20" name="println">
                <Arguments>
                  <Argument>
                    <JBinaryExpression line="20" type="" operator="&lt;">
                      <Lhs>
                        <JVariable name="x"/>
                      </Lhs>
                      <Rhs>
```

```
                        <JLiteralInt line="20" type="" value="100"/>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
            <JStatementExpression line="21">
              <JMessageExpression line="21" name="println">
                <Arguments>
                  <Argument>
                    <JBinaryExpression line="21" type="" operator="&gt;=">
                      <Lhs>
                        <JVariable name="x"/>
                      </Lhs>
                      <Rhs>
                        <JLiteralInt line="21" type="" value="50"/>
                      </Rhs>
                    </JBinaryExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 8. (*Conditional Expression*) Add support for conditional expression (`e1 ? e2 : e3`).

```
$ $j/j--/bin/javaccj-- -p tests/ConditionalExpression.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/ConditionalExpression.java"/>
  <Imports>
    <Import name="java.lang.Integer"/>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="4" name="ConditionalExpression" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="5" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="5" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="5">
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="6" name="x" type="int">
                  <Initializer>
                    <JMessageExpression line="6" name="parseInt">
                      <Arguments>
                        <Argument>
                          <JArrayExpression>
```

```
                            <TheArray>
                              <JVariable name="args"/>
                            </TheArray>
                            <IndexExpression>
                              <JLiteralInt line="6" type="" value="0"/>
                            </IndexExpression>
                          </JArrayExpression>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JStatementExpression line="7">
              <JMessageExpression line="7" name="println">
                <Arguments>
                  <Argument>
                    <JConditionalExpression line="7" type="" operator="?">
                      <TestExpression>
                        <JBinaryExpression line="7" type="" operator="==">
                          <Lhs>
                            <JBinaryExpression line="7" type="" operator="%">
                              <Lhs>
                                <JVariable name="x"/>
                              </Lhs>
                              <Rhs>
                                <JLiteralInt line="7" type="" value="2"/>
                              </Rhs>
                            </JBinaryExpression>
                          </Lhs>
                          <Rhs>
                            <JLiteralInt line="7" type="" value="0"/>
                          </Rhs>
                        </JBinaryExpression>
                      </TestExpression>
                      <TrueClause>
                        <JLiteralString line="7" type="" value="&quot;even&quot;"/>
                      </TrueClause>
                      <FalseClause>
                        <JLiteralString line="7" type="" value="&quot;odd&quot;"/>
                      </FalseClause>
                    </JConditionalExpression>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 9. (*Switch Statement*) Add support for a switch statement.

```
$ $j/j--/bin/javaccj-- -p tests/SwitchStatement.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/SwitchStatement.java"/>
  <Imports>
    <Import name="java.lang.Integer"/>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="4" name="SwitchStatement" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="5" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="5" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="5">
            <JSwitchStatement line="6">
              <TestExpression>
                <JMessageExpression line="6" name="parseInt">
                  <Arguments>
                    <Argument>
                      <JArrayExpression>
                        <TheArray>
                          <JVariable name="args"/>
                        </TheArray>
                        <IndexExpression>
                          <JLiteralInt line="6" type="" value="0"/>
                        </IndexExpression>
                      </JArrayExpression>
                    </Argument>
                  </Arguments>
                </JMessageExpression>
              </TestExpression>
              <SwitchBlockStatementGroup>
                <CaseLabel>
                  <JLiteralInt line="7" type="" value="1"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="8" type="" value="2"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="9" type="" value="3"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="10" type="" value="4"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="11" type="" value="5"/>
                </CaseLabel>
                <Body>
                  <JStatementExpression line="12">
                    <JMessageExpression line="12" name="println">
                      <Arguments>
                        <Argument>
                          <JLiteralString line="12" type="" value="&quot;Spring&quot;"/>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </JStatementExpression>
                </Body>
                <Body>
                  <JBreakStatement line="13">
                  </JBreakStatement>
                </Body>
              </SwitchBlockStatementGroup>
              <SwitchBlockStatementGroup>
                <CaseLabel>
                  <JLiteralInt line="14" type="" value="6"/>
                </CaseLabel>
                <CaseLabel>
```

```
                  <JLiteralInt line="15" type="" value="7"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="16" type="" value="8"/>
                </CaseLabel>
                <Body>
                  <JStatementExpression line="17">
                    <JMessageExpression line="17" name="println">
                      <Arguments>
                        <Argument>
                          <JLiteralString line="17" type="" value="&quot;Summer&quot;"/>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </JStatementExpression>
                </Body>
                <Body>
                  <JBreakStatement line="18">
                  </JBreakStatement>
                </Body>
              </SwitchBlockStatementGroup>
              <SwitchBlockStatementGroup>
                <CaseLabel>
                  <JLiteralInt line="19" type="" value="9"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="20" type="" value="10"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="21" type="" value="11"/>
                </CaseLabel>
                <CaseLabel>
                  <JLiteralInt line="22" type="" value="12"/>
                </CaseLabel>
                <Body>
                  <JStatementExpression line="23">
                    <JMessageExpression line="23" name="println">
                      <Arguments>
                        <Argument>
                          <JLiteralString line="23" type="" value="&quot;Fall&quot;"/>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </JStatementExpression>
                </Body>
                <Body>
                  <JBreakStatement line="24">
                  </JBreakStatement>
                </Body>
              </SwitchBlockStatementGroup>
              <SwitchBlockStatementGroup>
                <DefaultLabel/>
                <Body>
                  <JStatementExpression line="26">
                    <JMessageExpression line="26" name="println">
                      <Arguments>
                        <Argument>
                          <JLiteralString line="26" type="" value="&quot;Error!&quot;"/>
                        </Argument>
                      </Arguments>
                    </JMessageExpression>
                  </JStatementExpression>
                </Body>
              </SwitchBlockStatementGroup>
            </JSwitchStatement>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 10. (*Do-while Statement*) Add support for a do-while statement.

```
$ $j/j--/bin/javaccj-- -p tests/DoWhileStatement.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/DoWhileStatement.java"/>
  <Imports>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="3" name="DoWhileStatement" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="4" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="4" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="4">
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="5" name="i" type="int">
                  <Initializer>
                    <JLiteralInt line="5" type="" value="0"/>
                  </Initializer>
                </JVariableDeclarator>
                <JVariableDeclarator line="5" name="sum" type="int">
                  <Initializer>
                    <JLiteralInt line="5" type="" value="0"/>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JDoWhileStatement line="6">
              <Body>
                <JBlock line="6">
                  <JStatementExpression line="7">
                    <JBinaryExpression line="7" type="" operator="+=">
                      <Lhs>
                        <JVariable name="sum"/>
                      </Lhs>
                      <Rhs>
                        <JUnaryExpression line="7" type="" operator="post++">
                          <Operand>
                            <JVariable name="i"/>
                          </Operand>
                        </JUnaryExpression>
                      </Rhs>
                    </JBinaryExpression>
                  </JStatementExpression>
                </JBlock>
              </Body>
              <TestExpression>
                <JBinaryExpression line="8" type="" operator="&lt;=">
```

```
                  <Lhs>
                    <JVariable name="i"/>
                  </Lhs>
                  <Rhs>
                    <JLiteralInt line="8" type="" value="10"/>
                  </Rhs>
                </JBinaryExpression>
              </TestExpression>
            </JDoWhileStatement>
            <JStatementExpression line="9">
              <JMessageExpression line="9" name="println">
                <Arguments>
                  <Argument>
                    <JVariable name="sum"/>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 11. (*For Statement*) Add support for a for statement.

```
$ $j/j--/bin/javaccj-- -p tests/ForStatement.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/ForStatement.java"/>
  <Imports>
    <Import name="java.lang.System"/>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="3" name="ForStatement" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="4" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="4" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="4">
            <JVariableDeclaration>
              <Modifiers>
              </Modifiers>
              <VariableDeclarators>
                <JVariableDeclarator line="5" name="sum" type="int">
                  <Initializer>
                    <JLiteralInt line="5" type="" value="0"/>
                  </Initializer>
                </JVariableDeclarator>
              </VariableDeclarators>
            </JVariableDeclaration>
            <JForStatement line="6">
              <InitialExpression>
                <JVariableDeclaration>
                  <Modifiers>
                  </Modifiers>
                  <VariableDeclarators>
                    <JVariableDeclarator line="6" name="i" type="int">
                      <Initializer>
                        <JLiteralInt line="6" type="" value="1"/>
                      </Initializer>
                    </JVariableDeclarator>
                  </VariableDeclarators>
                </JVariableDeclaration>
              </InitialExpression>
              <TestExpression>
                <JBinaryExpression line="6" type="" operator="&lt;=">
                  <Lhs>
                    <JVariable name="i"/>
                  </Lhs>
                  <Rhs>
                    <JLiteralInt line="6" type="" value="10"/>
                  </Rhs>
                </JBinaryExpression>
              </TestExpression>
              <UpdateExpression>
                <JStatementExpression line="6">
                  <JUnaryExpression line="6" type="" operator="post++">
                    <Operand>
                      <JVariable name="i"/>
                    </Operand>
                  </JUnaryExpression>
                </JStatementExpression>
              </UpdateExpression>
              <Statement>
                <JBlock line="6">
                  <JStatementExpression line="7">
                    <JBinaryExpression line="7" type="" operator="+=">
                      <Lhs>
                        <JVariable name="sum"/>
                      </Lhs>
                      <Rhs>
                        <JVariable name="i"/>
                      </Rhs>
                    </JBinaryExpression>
                  </JStatementExpression>
                </JBlock>
              </Statement>
            </JForStatement>
            <JStatementExpression line="9">
              <JMessageExpression line="9" name="println">
                <Arguments>
                  <Argument>
                    <JVariable name="sum"/>
                  </Argument>
                </Arguments>
              </JMessageExpression>
            </JStatementExpression>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 12. (*Exception Handlers*) Add support for exception handling, which involves supporting the `try`, `catch`, `finally`, `throw`, and `throws` clauses.

```
$ $j/j--/bin/javaccj-- -p tests/ExceptionHandlers.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/ExceptionHandlers.java"/>
  <Imports>
  </Imports>
  <TypeDeclarations>
    <JClassDeclaration line="1" name="ExceptionHandlers" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <ClassBlock>
      <JMethodDeclaration line="2" name="f" returnType="void">
        <Modifiers>
          <Modifier name="private"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
        </FormalParameters>
        <Exceptions>
          <Exception type="Exception1"/>
          <Exception type="Exception2"/>
        </Exceptions>
        <Body>
          <JBlock line="2">
            <JThrowStatement line="3">
              <JNewOp line="3" type="Exception1"/>
                <Arguments>
                </Arguments>
              </JNewOp>
            </JThrowStatement>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      <JMethodDeclaration line="6" name="main" returnType="void">
        <Modifiers>
          <Modifier name="public"/>
          <Modifier name="static"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="6" name="args" type="String[]"/>
        </FormalParameters>
        <Body>
          <JBlock line="6">
            <JTryCatchFinallyStatement line="7">
              <TryBlock>
                <JBlock line="7">
                  <JStatementExpression line="8">
                    <JMessageExpression line="8" name="f">
                      <Arguments>
                      </Arguments>
                    </JMessageExpression>
                  </JStatementExpression>
                </JBlock>
              </TryBlock>
              <CatchBlock>
                <JFormalParameter line="10" name="e1" type="Exception1"/>
                <JBlock line="10">
                  <JEmptyStatement line="10"/>
                </JBlock>
              </CatchBlock>
              <CatchBlock>
                <JFormalParameter line="11" name="e2" type="Exception2"/>
                <JBlock line="11">
                  <JEmptyStatement line="11"/>
                </JBlock>
              </CatchBlock>
              <FinallyBlock>
                <JBlock line="12">
                  <JEmptyStatement line="12"/>
                </JBlock>
              </FinallyBlock>
            </JTryCatchFinallyStatement>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Problem 13. (*Interface Type Declaration*) Implement support for interface declaration.

```
$ $j/j--/bin/javaccj-- -p tests/Interface.java
<?xml version="1.0" encoding="utf-8"?>
<JCompilationUnit line="1">
  <Source fileName="tests/Interface.java"/>
  <Imports>
  </Imports>
  <TypeDeclarations>
    <JInterfaceDeclaration line="1" name="A">
      <Modifiers>
      </Modifiers>
      <InterfaceBlock>
        <JMethodDeclaration line="2" name="f" returnType="int">
          <Modifiers>
            <Modifier name="public"/>
          </Modifiers>
          <FormalParameters>
            <JFormalParameter line="2" name="x" type="int"/>
          </FormalParameters>
        </JMethodDeclaration>
      </InterfaceBlock>
    </JInterfaceDeclaration>
    <JClassDeclaration line="5" name="B" super="java.lang.Object">
      <Modifiers>
        <Modifier name="public"/>
      </Modifiers>
      <Implements>
        <Implement name="A"/>
      </Implements>
      <ClassBlock>
      <JMethodDeclaration line="6" name="f" returnType="int">
        <Modifiers>
          <Modifier name="public"/>
        </Modifiers>
        <FormalParameters>
          <JFormalParameter line="6" name="x" type="int"/>
        </FormalParameters>
        <Body>
          <JBlock line="6">
            <JReturnStatement line="7">
              <JBinaryExpression line="7" type="" operator="*">
                <Lhs>
                  <JVariable name="x"/>
                </Lhs>
                <Rhs>
                  <JVariable name="x"/>
                </Rhs>
              </JBinaryExpression>
            </JReturnStatement>
          </JBlock>
        </Body>
      </JMethodDeclaration>
      </ClassBlock>
    </JClassDeclaration>
  </TypeDeclarations>
</JCompilationUnit>
```

Files to Submit

1. `j--.tar.gz` (*j--* source tree as a single gzip file)
2. `report.txt` (project report)

Before you submit:

- Make sure you create the gzip file `j--.tar.gz` 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