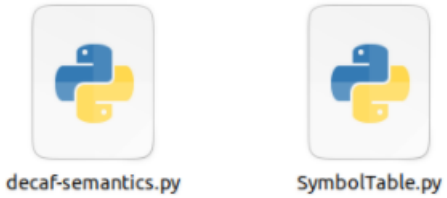
Lab 2: Decaf Semantic Analysis

Following the webinar, you now have some tools for finding and reporting semantic errors. If you were not present, or have not watched the recording of the webinar, you should use this opportunity to watch the webinar and follow along to learn the basics you need to complete this lab.

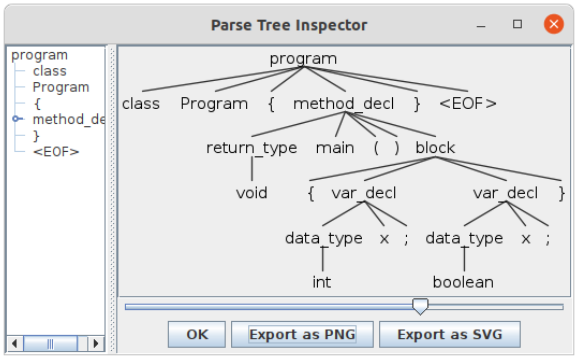
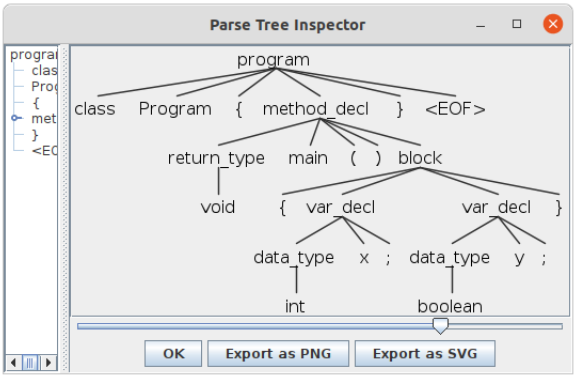
Download **decaf-semantics.py** and **SymbolTable.py** from the Moodle page.



For this lab, you need to refer to **Semantic Rules** in the Decaf Language.pdf coursework document.

Task 1: Implement *semantic rule 1* in the **Decaf Language.pdf** file: “No identifier is declared twice in the same scope”. Use the source files in testdata/semantics/ to test your implementation.

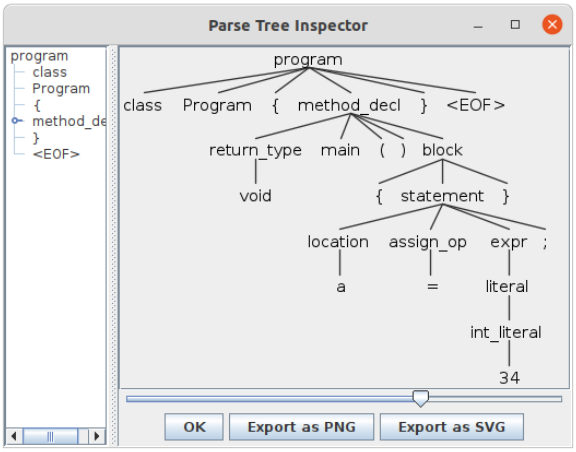
Note: Some of the test files purposely contain errors to enable robust testing of semantic rules. All the files with illegal in the name should produce an appropriate error, and all the files with legal in their name should produce no errors. For example:

testdata/semantics/illegal-01.dcf	testdata/semantics/legal-01.dcf
	
Error on line 3, ‘a’ is assigned before declared	

Task 2: Implement *semantic rule 2* in the **Decaf Language.pdf** file: “No identifier is used before it is declared”. Use the source files in testdata/semantics/ to test your implementation.

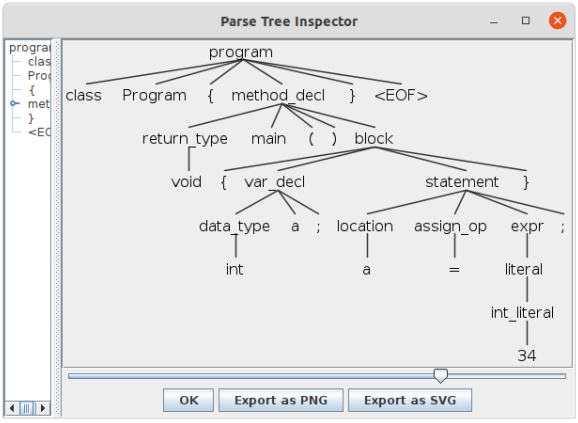
Note: Some of the test files purposely contain errors to enable robust testing of semantic rules. All the files with illegal in the name should produce an appropriate error, and all the files with legal in their name should produce no errors. For example:

testdata/semantics/Illegal-02.dcf



Error on line 4, ‘x’ was already declared on line 3

testdata/semantics/legal-02.dcf



Task 3: Now implement the remaining semantic rules (2-18). You are responsible for designing your own test cases. Complete as many rules as you can, even if only partially – it is not a requirement to complete the semantic checker before you can attempt code generation – but you must have completed the lexer and parser.

Please ***ask the tutor*** if you do not understand any of the semantic rules – the descriptions are not perfect and some intuition is required.

Note: your compiler should report all semantic errors to the programmer, rather than terminate compilation.