LINGI2132 Assignement 3 report

Julien De Coster, Xavier Crochet

April 3, 2013

1 Conditional Expression

1.1 Implementation

Quite straightforward here, analyze the parameters of the *JConditional* expression, check whether the expression parameter is a boolean and that the two others are from the same type. Then, generate the code corresponding to the ternary op. Notice that we use the $GOTO$ instruction in the branch instruction.

1.2 Tests

As always, test tricky cases such as

- Recursive *JConditionalExpression*
- Ones with assignment in the parameters (to test implicit conversion)

2 For Statement

2.1 Basic

Because of the grammar of the statement, we have to check in the analyze function whether each parameter (*forInit*, *forUpdate*, *forUpdate*) exists and analyze it accordingly. For the codgen part, we have to generate the code for each AST and don't forget to add, after the statement code generation, the $GOTO$ jump instruction to go to the test expression once gain.

2.2 Enhanced

Here, i decided to translate the Enhanced loop into a basic one. It's quite simple, we just have to create a basic for with

- Two instructions in the *forInit* part the first initializing the integer we use to iter in the array, the second initializing the element at the index i of the array.
- A test to check if we are still in the range of the array
- Two instructions in the *forUpdate* part the update of the two *forInit* variables.

2.3 Tests

We tested

- If the scope of the declared variable (in the foor loop) is respected
- Empty basic for loops
- Nasty basic for loops full of statements in the *forInit* and *forUpdate* part
- ...

3 What does not work

The final modifier compile but is not implemented. One can easily add a final modifier in the *forInit* part of the for loop and still update it in the *forpdate* part. I couldn't find a proper way to support it and refuse myself to write dirty code to support it.