Group Assignment

Compilers

Oliver Mikkel Winther Cristensen (klw741)

Mathias Brndum Leroul (swn848)

Sebastian O. Jensen (gjx653)

Turn-in date: 18-12-2015

Contents of the Report
It is largely up to you to decide what you think is important to include in the report, as long as the following requirements are met: Your report should justify all your changes to the compiler modules, in particular, the lexer, parser, interpreter, type checker, machine-code generator, and the optimization modules. All major design decisions should be presented and justified. When evaluating your work, the main focus will be on verifying that your implementation of the language is correct. While we do not put particular emphasis on compiler optimizations in this course, we will also evaluate the quality of the generated code: if there are obvious inefficiencies that could have been easily solved you will be penalized, as they testify either wrong priorities or lack of understanding. You should not include the whole compiler in your report, but you must include the parts that were either added, i.e. new code, or substantially modified. Add them as code listings, and use the appendix if they get too big. Ideally, we should not need to read your source code. Your report should describe whether the compilation and execution of your input/test (FASTO) programs results in the correct/expected behavior. If it does not, try to explain why this is. In addition, (i) it must be assessed to what extent the delivered test programs cover the language features, and (ii) if the implementation deviates from the correct/expected behavior than the test program(s) should illustrate the implementation shortcomings to your best extent. Known shortcomings in type checking and machine-code generation must be described, and, whenever possible, you need to make suggestions on how these might be corrected. The report should not exceed this document in size, and should have an appropriate level of detail. You might be penalized if your report includes too many irrelevant details.

1 Introduction

We're given the grand task of

2 Implementation

Notes

- Implemented Boolean Litterals: true and false.

  Added test files bool.fo, bool.in, bool.out

3 Results

4 Conclusion