Dr. Annette Bieniusa
M.Sc. Peter Zeller

TU Kaiserslautern
Fachbereich Informatik
AG Softwaretechnik

Lab 5: Compiler and Language Processing Tools (WS 2018)

Deadline: 22.01.2019, 10:00

1 Translation of Object-oriented constructs (12 points)

Extend your MiniJava to miniLLVM translator with the translation of object oriented constructs. With this step you will have a complete compiler for the MiniJava language.

This exercise is a continuation of exercise 4, so no new material is provided. The material in `ex4_material.zip` already contains test-inputs with classes in `testdata/translation/classes/`.

Hints:

1. Translate method calls to procedure calls, where the address of the procedure is read from the virtual method table of the receiver object.
2. Generate a struct type for every class. The struct type should have a reference to the virtual method table as the first field. Additionally, it should include fields for all the fields from the Java class. Remember that the order of fields is important and that the memory layout of classes has to be compatible with the memory layout of their super-classes.
3. For representing virtual method tables, you can create a struct type and a corresponding global, constant variable. The fields of the virtual method tables are pointers to procedures, so you have to calculate a corresponding type.
4. Create one constructor-procedure for each class. In the constructor you should allocate memory for the object (`alloc` instruction) and initialize it. In particular, you have to set the reference to the virtual method table.
5. You can use the `SizeOf` operand to calculate the size of a struct-type.
6. Remember, that objects can be `null` in Java.
7. There is no subtyping between struct types in LLVM, so you will have to use casts (`bitcast` instruction) in places where Java allows subtyping (assignments, equality checks, method calls, return statements).