Dynamic Languages – Übungen Blatt 10

Abgabedatum: `14. Juli 2010`

The purpose of this sheet is to turn the current AST-based interpreter into a bytecode-based one. Since this is a sizeable task, it will count for 20 points. The `eval`-method of the `Interpreter` class should first turn its first argument (an AST) into a bytecode object and then run that. In this way, all existing tests should continue to pass. In the file `test_bytecode.py` there are some additional tests, with increasing complexity of the bytecode. They also don't use any builtin code to make things easier.

The compiler that turns the ASTs into bytecode is provided for you in the file `simple/compile.py`. The file also contains documentation about the instruction set of the bytecode. The most important function in that file is `compile(ast)` which returns a bytecode object.

In addition there is also a disassembler in file `simple/disass.py` that can be used for debugging.

A function that you will probably need is the builtin `ord`, which turns a single character into its ASCII value. A stack can be implemented by using a simple list and the `append` and `pop` methods.