C-Subset Bauminterpreter in Python

Christoph Weiler
1029175
christoph.weiler@tuwien.ac.at

26. Juni 2014

Das C-Subset

Mein C-Subset besteht aus folgenden Sprachelementen:

- Datentypen: *int*, *char*
- Logischen Operatoren: &&, ||, !
- Arithmetische Operatoren: $+, -, *, /, <<, >>$
- Vergleichs-Operatoren: $<, >, ==$
- Funktionen
- Selection-Statements: *if* , *if-else*
- Iteration-Statements: *while*, *do-while*, *for*

Verwendete Technologien

Generell verwendet mein Interpreter reines Python mit nur wenigen Libraries:

- LexerGenerator aus dem Package *rply*
- ParserGenerator aus dem Package *rply*
- *Colorama* für eine farbige Ausgabe der Tests
- Einige Standardlibraries der Python-API (*os*, *resource*, *sys*)

Genereller Aufbau des Interpreters

Der Aufbau gliedert sich in:

- Lexikalische Analyse
- Parsing zu Abstrakten Syntaxbaum
- Semantische Analyse und Interpretation des Baums (dynamisches Typsystem)

Lexikalische Analyse

```
lg = LexerGenerator()

# Control Elements
lg.add("IF", r"if")
lg.add("ELSE", r"else")
lg.add("DO", r"do")
lg.add("WHILE", r"while")
lg.add("FOR", r"for")
lg.add("RETURN", r"return")

# Arithmethics
lg.add("PLUS", r"\+")
lg.add("MINUS", r"\-")
lg.add("TIMES", r"\*")
lg.add("DIVIDE", r"\/")
...
```

Parser

```
    ...

@pg.production('IfStatement : IF LPAREN BooleanExpr
                    RPAREN LBRACE Statements RBRACE')
def ifstatement(p):
    boolean = p[2];
    statements = p[5]
    return abstractsyntaxtree.IfElse(boolean,
        statements, None)

    ...
```

Abstrakter Syntaxbaum

```
class While(BaseBox):
  ...
  def eval(self, variables, root):
    # Copy variables to maintain variable scope
    whilevariables = variables.copy()

    # Execute while-block while condition holds
    while self.condition.eval(whilevariables, root):
      returnval = self.whileblock.eval(
          whilevariables, root)
      if type(returnval) == Number:
        return returnval

    # Recombine variables with while-variables
    for key in whilevariables.keys():
      if key in variables:
        variables[key] = whilevariables[key]
```