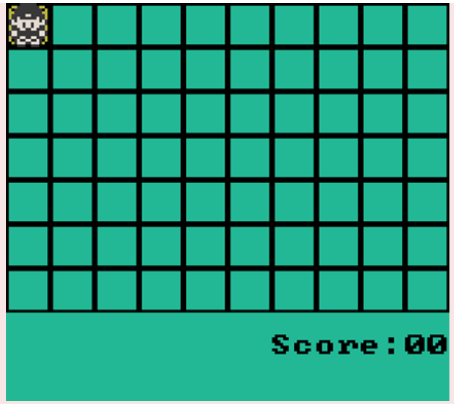
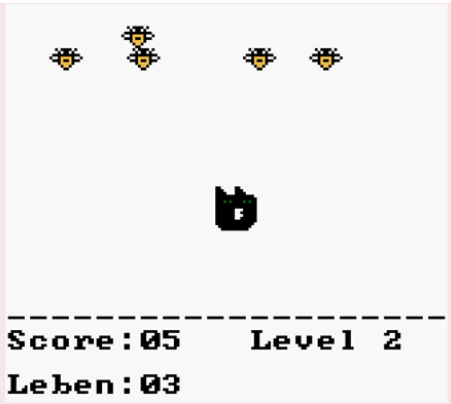
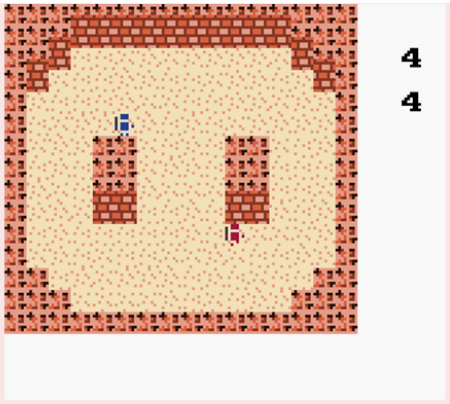
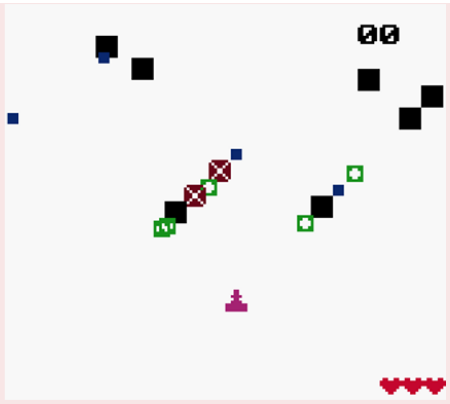
Software Engineering
Programming Languages

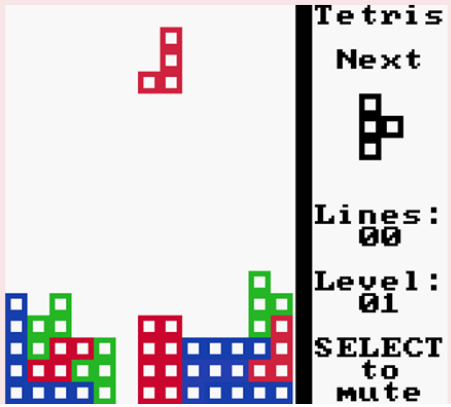
ulm university universität ulm

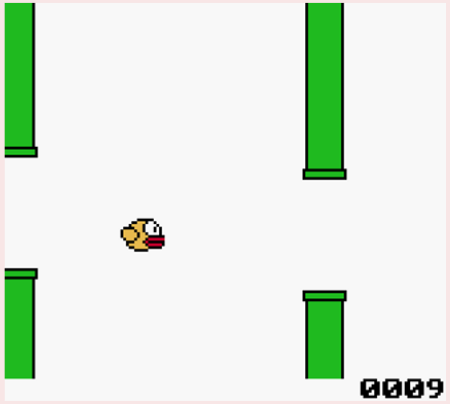










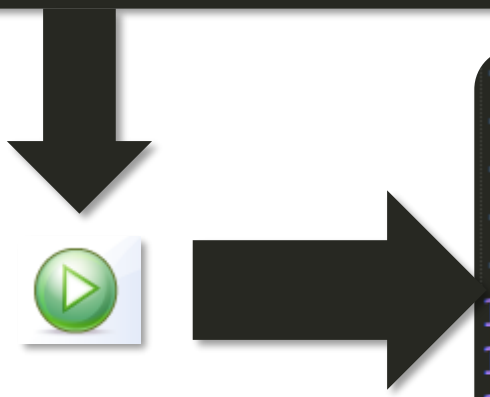


ProSeminar Compilerbau

Was passiert, wenn ihr `javac` aufruft?

In diesem Seminar lernt ihr, wie ein Werkzeug, das ihr jeden Tag verwendet, eigentlich funktioniert.

```
for(int i = 99; i >= 0; i--) {
  System.out.println(i + " bottles of beer on the wall.");
}
```

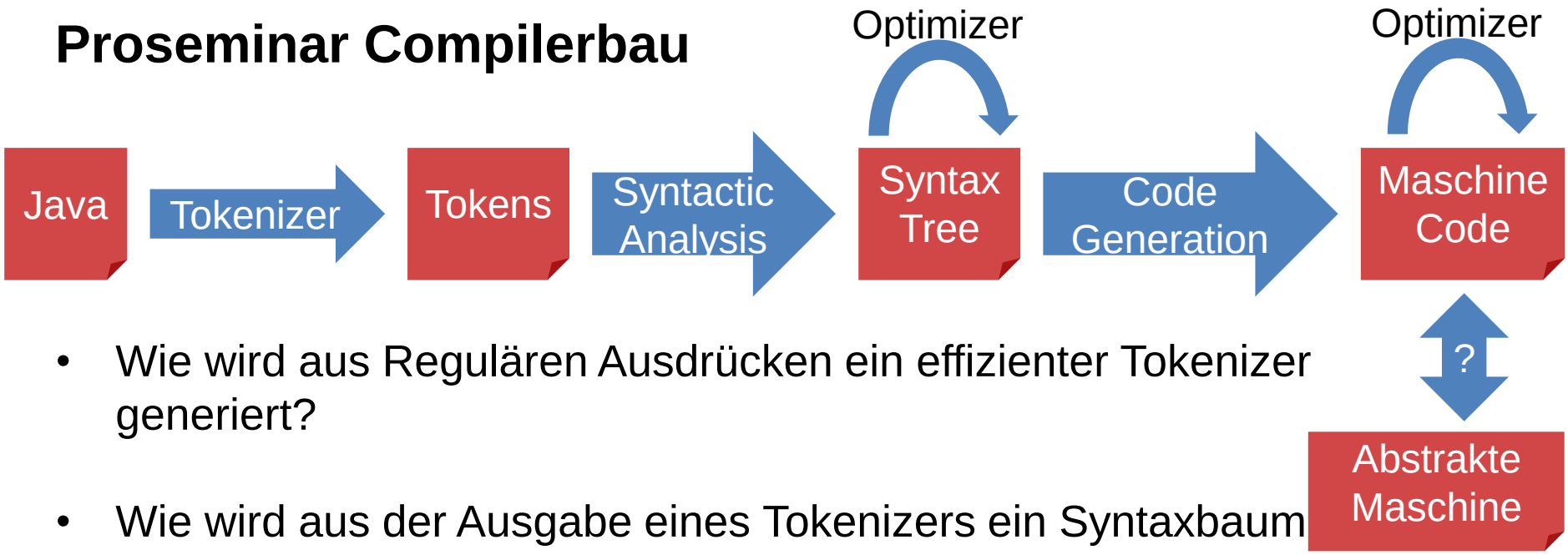


```
 0: bipush        99
 2: istore_1
 3: iload_1
 4: iflt          38
 7: getstatic     # Field System.out
10: new           # class StringBuilder
13: dup
14: invokespecial # new String
17: iload_1
18: invokevirtual # Method append
21: ldc           # " bottles of beer on the wall."
23: invokevirtual # Method append
26: invokevirtual # Method toString
29: invokevirtual # Method println
32: iinc          1, -1
35: goto          3
38: return
```

Aufbau des ProSeminars

- Wir bieten ein ProSeminar in dem ihr…
  - … lernt wie ein Compiler funktioniert
  - … lernt wie man wissenschaftlich arbeitet und Vorträge halt
  - … optional einen Teil eines Compilers implementieren könnt (gerne auch in Haskell)

Proseminar Compilerbau



- Wie wird aus Regulären Ausdrücken ein effizienter Tokenizer generiert?
- Wie wird aus der Ausgabe eines Tokenizers ein Syntaxbaum aufgebaut?
- Wie wird aus einem Syntaxbaum Maschinencode generiert?
- Welche Optimierungen können auf dem Syntaxbaum und generiertem Maschinencode durchgeführt werden?
- Was ist eine abstrakte Maschine und was hat sie mit Compilern zu tun?

Erwartungen

- Selbständiges recherchieren und lernen, aber ihr könnt immer vorbei kommen und fragen stellen.

- Eine 20 minütige Präsentation (Deutsch oder Englisch)
- 8-10 Seiten Ausarbeitung auf Englisch
- Peer-Reviews von zwei anderen Ausarbeitungen

Gliederung und Literaturliste

- Gliederung enthält Kapitel und Unter-Kapitel
- Mindestens ein Satz oder Stichpunkte über den Inhalt pro Unter-Kapitel
- Mindestens drei weitere wissenschaftliche Quellen
  - Wikipedia ist keine wissenschaftliche Quelle
  - Keine Quellen angeben, die ihr nicht mindestens überflogen habt
- Die Gliederung ist eure erste Gelegenheit Feedback zu bekommen
- Die Gliederung ist leichter zu ändern als eure Ausarbeitung

Vortrag: Wissenschaftliches Arbeiten

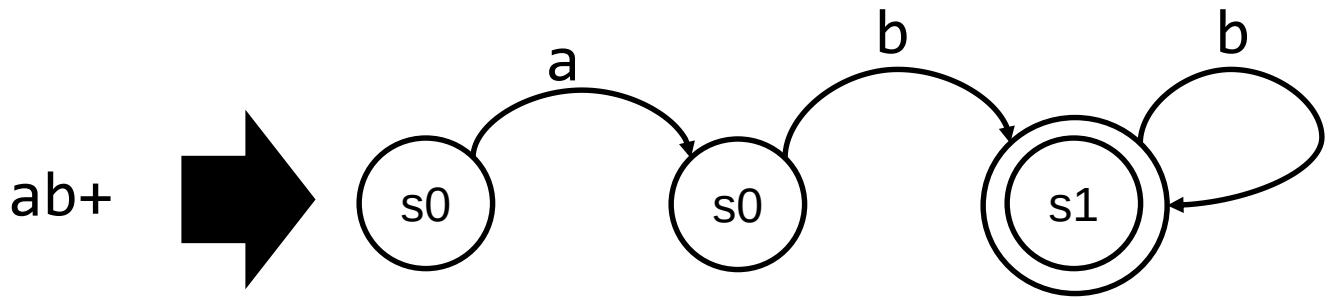
- Pflicht für Proseminar Teilnehmer
- Mittwoch (25.04.), 10 -12 Uhr, O29/1001

Wichtige Termine

Datum	Kalenderwoche	
20/04/2018	16	Einführungsveranstaltung
25/04/2018	17	Vortrag: Wissenschaftliches Arbeiten
30/04/2018	18	Abgabe Gliederung und Literaturliste
11/05/2018	19	Verbindliche Anmeldung im LSF
14/05/2018	20	Abgabe 1. Version (Feedback KW 20)
28/05/2018	22	Abgabe 2.Version für Peer-Review
04/06/2018	23	Abgabe Peer-Review
11/06/2018	24	Abgabe 3. Version (Feedback KW 25)
25/06/2018	26	Abgabe Finale Version
02/07/2018	27	Abgabe Folien (Feedback selbe Woche)
09/07/2018	28	Vortrag (genauer Termin wird noch bekannt gegeben)

Reguläre Ausdrücke und Automaten

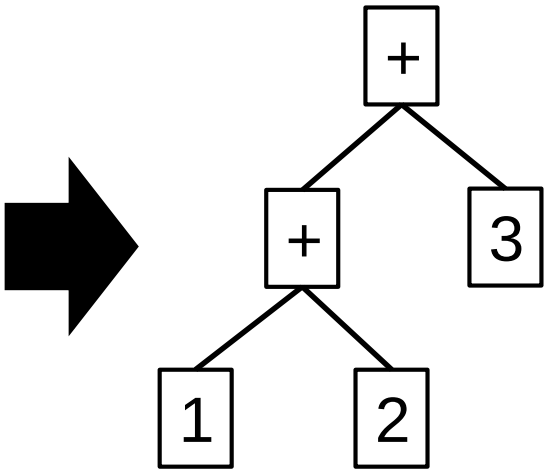
- Wie können Reguläre Ausdrücke effizient umgesetzt werden?
- Was ist ein Tokenizer und wofür ist er gut?



Grammatiken und Syntaktische Analyse

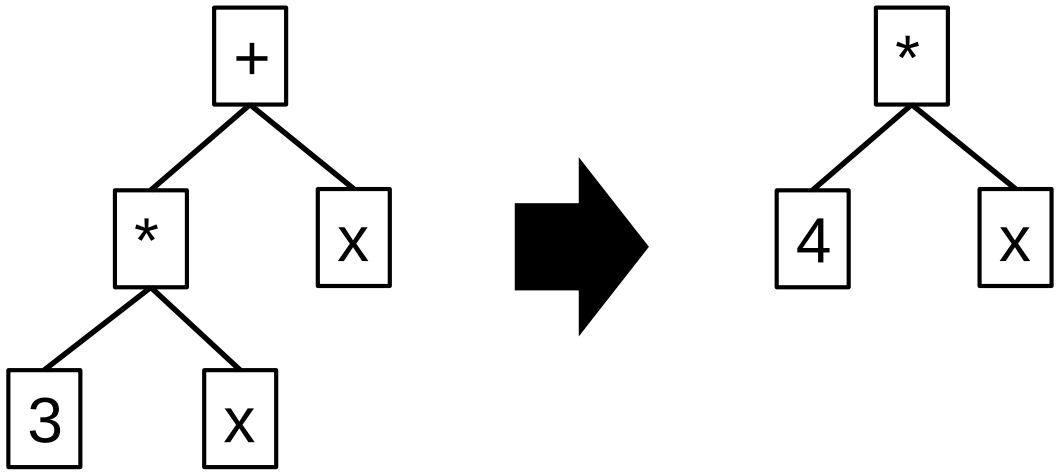
- Was ist eine EBNF?
- Was ist Syntaktische Analyse?
- Wie kann man aus einer Grammatik einen Parser erzeugen/programmieren?
  - Beschreibe eine Methode ausführlich

```
S   = Bin | Num
Bin = Num Op Bin
Num = “1” | “2” | “3” | …
Op  = “+” | “*”
```



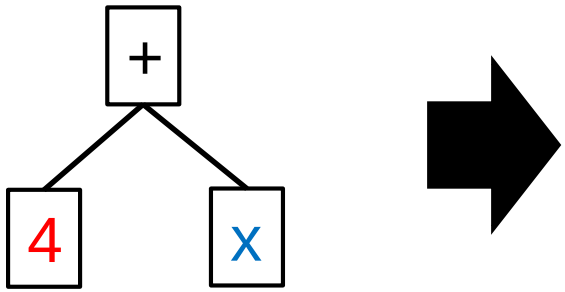
Syntaxbäume und Optimierungen

- Was ist ein Syntaxbaum?
- Wofür wird er verwendet?
- Welche Optimierungen und Analysen gibt es auf Syntaxbäumen?



Code-Generierung

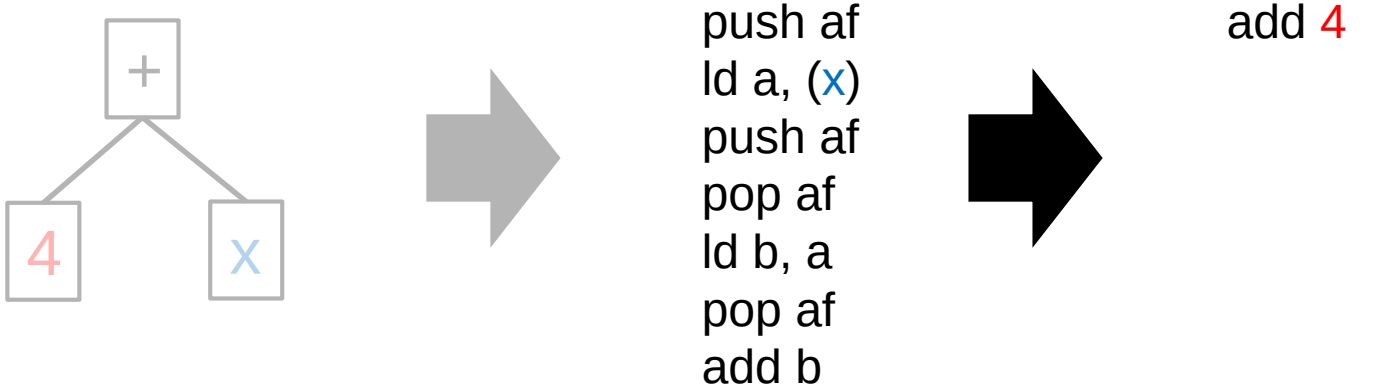
- Wie wird aus einem Syntaxbaum Code generiert?
- Was ist eine Stackmaschine?
- Was ist eine Registermaschine?




```
ld a, 4
push af
ld a, (x)
push af
pop af
ld b, a
pop af
add b
```

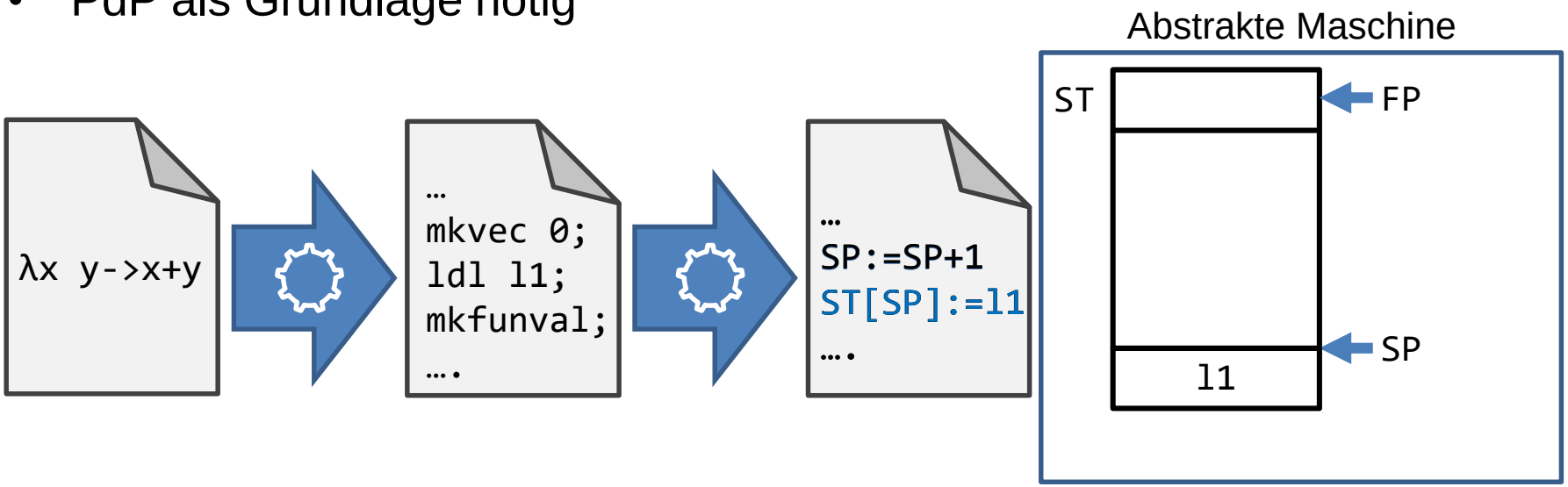
Optimierungen auf Assembler Niveau

- Was für Optimierungen kann man auf Assembler Code durchführen?
- Was ist Peephole Optimierung?
- Was ist Register Allokation?



Abstrakte Maschinen für funktionale oder logische Programmiersprachen

- Wofür wird eine abstrakte Maschine benötigt?
- Wie werden die Konzepte einer funktionalen oder logischen Programmiersprache übersetzt?
  - Wie wird Lazy-Evaluation/Unifikation realisiert?
- PdP als Grundlage nötig



Literatur

- R. Wilhelm, D. Maurer: Übersetzerbau. 2. Aufl. Springer 1997
- A, Aho, S Lam, R. Sethi, J. Ullman: Compiler. 2. Aufl. Pearson Studium 2008
- R. Wilhelm, H. Seidl: Übersetzerbau – Virtuelle Maschinen. Springer 2007

Themenliste

- Reguläre Ausdrücke und Automaten
- Grammatiken und Syntaktische Analyse
- Syntaxbäume und Optimierungen
- Code-Generierung
- Optimierungen auf Assembler Niveau
- Abstrakte Maschinen
  - für funktionale und logische Programmiersprachen
  - LLVM