Blatt 01: Reguläre Sprachen

BC George, Carsten Gips (HSBI)

1 Sprachen von regulären Ausdrücken (1P)

Welche Sprache wird von dem folgenden regulären Ausdruck beschrieben?

$a + a\,(a + b)^*\,a$

2 Bezeichner in Programmiersprachen (3P)

Betrachten Sie eine Programmiersprache, in der die Bezeichner (= Namen für Variablen, Funktionen, Klassen, Methoden, …) folgenden Aufbau haben:

- Alle Variablennamen beginnen mit **V** oder **v**
- Handelt es sich um globale Variablen, beginnen Sie mit **V**, lokale beginnen mit **v**
- Funktions- und Methodenparameter beginnen mit **p**, KLassenparameter (bei der Definition von Vererbung) beginnen mit **P**
- Weitere Bezeichner müssen mit einem Buchstaben (a-z, A-Z) beginnen
- Die folgenden Zeichen dürfen Buchstaben, Ziffern und ein Untersreich sein
- Bezeichner dürfen nicht mit einem Unterstrich enden
- Alle Bezeichner müssen aus mindestens zwei Zeichen bestehen

Entwickeln Sie einen regulären Ausdruck, der den Aufbau der Bezeichner beschreibt. Beachten Sie, dass Ihr regex alle zulässigen Bezeichner beschreiben muss, aber keinen einzigen unzulässigen beschreiben darf. Wählen Sie zwei Bezeichner aus der Sprache und zeigen Sie, wie sie vom regex gematcht werden.

Entwickeln Sie einen DFA, der diese Bezeichner akzeptiert. Beachten Sie, dass Ihr DFA alle zulässigen Bezeichner akzeptieren muss, aber keinen einzigen unzulässigen akzeptieren darf. Wählen Sie zwei Bezeichner aus der Sprache und zeigen Sie, wie sie vom Automaten zeichenweise gelesen und akzeptiert werden.

Entwickeln Sie eine reguläre Grammatik, die diese Bezeichner generiert. Beachten Sie, dass Ihre Grammatik alle zulässigen Bezeichner generieren können muss, aber keinen einzigen unzulässigen generieren darf. Wählen Sie zwei Bezeichner aus der Sprache und zeigen Sie die Ableitungsbäume dazu.

3 Gleitkommazahlen in Programmiersprachen (2P)

Recherchieren Sie zunächst den Aufbau von Gleitkommazahlen in Python und Java.

Erstellen Sie für jede der beiden Programmiersprachen reguläre Ausdrücke, DFAs und reguläre Grammatiken wie in Aufgabe 1. Verifizieren Sie Ihre Lösungen wie in Aufgabe

1.

4 Mailadressen? (1P)

Warum ist der folgende regex ungeeignet für die Verarbeitung von Mailadressen?

$(a-z)^{+}@(a-z).(a-z)$

Bitte beachten Sie, dass die Schreibweise a-z nicht unserer Definition genügt. Eigentlich müsste jedes Zeichen aufgeführt werden:

$a+b+c+c+\ldots+z$ ist besser, aber immer noch nicht richtig. Warum?

Anmerkung: Diese Darstellung wird ab jetzt akzeptiert.

Verbessern Sie den gegebenen regulären Ausdruck.

5 Der zweitletzte Buchstabe (1P)

Entwickeln Sie einen DFA, der nur Wörter über $\Sigma = \{1, 2, 3\}$ akzeptiert, deren zweitletztes Zeichen dasselbe ist wie das zweite.

6 Sprache einer regulären Grammatik (2P)

Welche Sprache generiert die folgende Grammatik?

$$
\begin{aligned}
S &\rightarrow aA \\
A &\rightarrow dB \mid bA \mid cA \\
B &\rightarrow ac \mid bC \mid cA \\
C &\rightarrow \epsilon
\end{aligned}
$$

> **Last modified:** f7ac9d2 (reformat using shorter lines, 2025-08-09)