

Übersetzerbau: Übung 05

von

Naja v. Schmude (4127652), Lisa Dohrmann (4130066)

Aufgabe 1

- a) Reguläre Definition von Gleitkommazahlen in Java: Wir wollen mit unserer Definition Zahlen wie 3E1, 3.01e-123, -3d, 3.04F, 3.01E+10D darstellen können, also sowohl Java `float` als auch `double`.

$$\begin{aligned} digit &\rightarrow 0|1|\dots|9 \\ digits &\rightarrow digit\ digit^* \\ exp &\rightarrow (E|e)(+|-|\epsilon)digits \\ double &\rightarrow (+|-|\epsilon)digits(.digits|\epsilon)(exp|\epsilon)(d|D|f|F|\epsilon) \end{aligned}$$

- b) Jetzt verwenden wir die erweiterten regulären Ausdrücke für die Definition aus a)

$$\begin{aligned} digit &\rightarrow [0-9] \\ digits &\rightarrow digit^+ \\ exp &\rightarrow (E|e)(+|-)?digits \\ double &\rightarrow (+|-)?digits(.digits)?(exp)?(d|D|f|F)? \end{aligned}$$

Aufgabe 2

- a) Alle Wörter aus kleinen Buchstaben, die die fünf Vokale in ihrer Reihenfolge enthalten.

$$\begin{aligned} letter &\rightarrow [b-df-hj-np-tv-z] \\ word &\rightarrow letter^*a\ letter^*e\ letter^*i\ letter^*o\ letter^*u\ letter^* \end{aligned}$$

- b) Alle Wörter aus kleinen Buchstaben, in der die Buchstaben in aufsteigender Reihenfolge erscheinen.

Wir gehen hier davon aus, dass in der Zeichenfolge Buchstaben mehrmals hintereinander stehen können, welches dem Sortierungskriterium ja nicht widerspricht. Will man keine doppelten zulassen, ist nur der Kleene-Stern durch '?' zu ersetzen.

$$word \rightarrow a^*b^*c^*\dots y^*z^*$$

- c) Alle Ziffernfolgen, in denen keine Ziffer mehrfach vorkommt.

Wenn man davon ausgeht, dass die Ziffernfolgen sortiert sind, ist die Definition leicht wie folgt aufzuschreiben.

$$number \rightarrow 0? 1? 2? 3? 4? 5? 6? 7? 8? 9?$$

Will man allerdings alle Permutationen der 10 Ziffern für alle möglichen Folgenlängen $1, \dots, 10$ erzeugen, haben wir keinen Weg gefunden dieses durch einen regulären Ausdruck aufzuschreiben, der nicht einfach nur alle Permutationen aufzählt.

d) Alle Ziffernfolgen, in denen höchstens eine Ziffer mehrfach vorkommt.

Auch hier gehen wir davon aus, dass es sich um sortierte Ziffernfolgen handelt, da wir ansonsten wie bei c) nur Möglichkeiten mit exponentiellem Aufwand gefunden haben.

$$\begin{aligned}
 zero &\rightarrow 0^* 1? 2? 3? 4? 5? 6? 7? 8? 9? \\
 one &\rightarrow 0? 1^* 2? 3? 4? 5? 6? 7? 8? 9? \\
 two &\rightarrow 0? 1? 2^* 3? 4? 5? 6? 7? 8? 9? \\
 &\dots \\
 nine &\rightarrow 0? 1? 2? 3? 4? 5? 6? 7? 8? 9^* \\
 number &\rightarrow zero \mid one \mid two \mid \dots \mid nine
 \end{aligned}$$

Aufgabe 3

Wir betrachten zunächst einmal das Beispiel $a\{2,4\}$. Hier können wir uns leicht überlegen, dass $(aa)a?a?$ ein äquivalenter Ausdruck zu dem vorherigen ist. Das können wir nun leicht auf den allgemeinen Fall übertragen

$$r\{m, n\} \Leftrightarrow r^m(r?)^{n-m}$$

wobei r^c für $c \in \mathbb{N}$ eine abkürzende Schreibweise für die c -malige Konkatenation von r sei.

Unsere Vermutung zeigen wir nun per vollständiger Induktion über n . Dabei sei m eine beliebige, feste Zahl mit $0 \leq m \leq n$.

I.A. $m = n = 1$: $r\{m, n\} = r\{1, 1\} = r = r^1(r?)^{1-1}$

I.S. $n = n + 1, n \geq m$:

$$\begin{aligned}
 r\{m, n + 1\} &= r\{m, n\}r? = r^m(r?)^{n-m}r? \\
 &= r^m(r?)^{n+1-m}
 \end{aligned}$$

□

Aufgabe 4

Schlüsselwörter, die „case insensitive“ erkannt werden sollen, brauchen statt einer „case sensitiven“ regulären Definition wie z.B. $select \rightarrow select$, eine etwas aufwändigere Definition, die für jeden Buchstaben seine Klein- oder Großschreibung zulässt:

$$select = [sS][eE][lL][eE][cC][tT]$$