Reguläre Ausdrücke

Worum geht es?

Ein regulärer Ausdruck (*regular expression*) ist ein Term, um Klassen von Zeichenketten zu definieren.

Eine Zeichenkette (*string*) ist eine Folge von Symbolen (Buchstaben, Ziffern, Sonderzeichen).

Reguläre Ausdrücke sind von praktischer als auch theoretischer Bedeutung:

- Suchen und Ersetzen in Texten
- Zerlegung von Computercode in *Tokens*
- Beschreibung von Proteinklassen in der Bioinformatik
- als alternative Darstellung *endlicher Automaten*

Suchmuster

Für die Suche mit einem regulären Ausdruck benötigt man ein Suchmuster (*pattern*) und einen Text den wir nach diesem Muster durchsuchen wollen.

Abhängig vom Programm, das reguläre Ausdrücke verwendet, wird der gefundene Text selbst oder seine Position (Zeilen- und Spaltennummer) ausgegeben.

Einfache Muster (Sequenzen)

Um in einem Text nach dem Muster *die* zu suchen, verwendet man den regulären Ausdruck `/die/`.

In der Fachliteratur werden die Suchmuster oft durch Schrägstriche `/.../` (*slashes*) begrenzt. Bei dem Programm, das wir zum Testen der Ausdrücke verwenden, müssen diese jedoch nicht eingegeben werden.

Beispiel 1

Muster:	`/die/`
Text:	Die Radieschen sind rot.

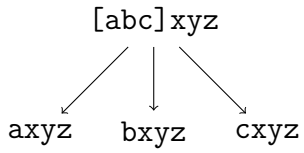
Reguläre Ausdrücke sind *case sensitive*; d. h. sie unterscheiden Gross- und Kleinschreibung.

Zeichenklassen

Zeichen, die innerhalb eckiger Klammern stehen, bilden eine Zeichenklasse.

`/[Dd]ie/` findet die Zeichenfolgen *Die* und *die*.

Man kann sich die Wirkung so vorstellen dass mit jedem Zeichen der Auswahl ein neuer Ausdruck gebildet wird.



Beispiel 2

Muster: `/[Dd]ie/`
Text: Die Radieschen sind rot.

Zusammenhängende Bereiche von Buchstaben und Ziffern können mit einem Bindestrich (*dash*) abgekürzt werden.

`/[A-Z]/` ein Grossbuchstabe
`/[a-z]/` ein Kleinbuchstabe
`/[0-9]/` eine Ziffer

(Teil-)Bereiche können kombiniert werden:

`/[a-zA-Z]/` ein Gross- oder Kleinbuchstabe
`/[a-fA-F0-9]/` eine Hexadezimalziffer

Ist das erste Zeichen der Zeichenklasse ein Zirkumflex (`^`), so wird nach allen Zeichen gesucht, die *nicht* in der angegebenen Klasse liegen.

`/[^A-Z]/` kein Grossbuchstabe
`/[^a-z]/` kein Kleinbuchstabe
`/[^0-9]/` keine Ziffer

Wenn der Zirkumflex das einzige Zeichen einer Zeichenklasse ist, behält er seine normale Bedeutung.

Ein Zirkumflex, der in einer Zeichenklasse mit mehr als einem Zeichen nicht an erster Stelle steht, behält seine normale Bedeutung.

Beispiel 3

(a) Muster: `/[1-9][0-9]/`
Text: 3, 05, 20, 37, 123, 45 67

(b) Muster: `/[1-9][^][1-9]/`
Text: 2^3

(c) Muster: `/[1-9][^^][1-9]/`
Text: 2+3

Vordefinierte Zeichenklassen

Da bestimmte Zeichenklassen häufig gebraucht werden, gibt es für sie Kurzformen:

`\d`	**d**igit	`[0-9]`
`\D`	no **d**igit	`[^0-9]`
`\w`	**w**ord character	`[a-zA-Z]`
`\W`	no **w**ord character	`[^a-zA-Z]`
`\s`	white**s**pace	`[␣\f\n\r\t\v]`
`\S`	no white**s**pace	`[^␣\f\n\r\t\v]`

Bedeutung der C-Steuerzeichen

`\f`: Form Feed (Seitenvorschub)
`\n`: Line Feed (Zeilenschaltung)
`\r`: Carriage Return (Wagenrücklauf)
`\t`: Horizontal Tab (Tabulator)
`\v`: Vertical Tab (Tabulator)

Beispiel 4

Muster: `/\d\w/`
Text: Klassen 1a und 1b

Quantoren

Anstatt ein Zeichen oder ein Zeichenklasse wiederholt aufzuschreiben, kann man mit Metasymbolen ein Häufigkeitsintervall des unmittelbar vorangehenden regulären Teilausdrucks angeben.

`*`	null oder mehr Wiederholungen (*Kleene-Star*)
`+`	mindestens eine Wiederholungen
`?`	höchstens eine Wiederholungen
`{n}`	genau `n` Wiederholungen
`{m,n}`	mindestens `m` und höchstens `n` Wiederholungen

- Da die Quantoren `*` und `?` auch auf null Vorkommen des davor stehenden regulären Ausdrucks passen, treffen sie möglicherweise auch auf null Zeichen; also auf Zeichenzwischenräume zu. (siehe Beispiel 5)
- Da die Quantoren `*` und `+` auf mindestens eine Wiederholung des davor stehenden regulären Ausdrucks passen, versuchen sie auf so viele Zeichen wie möglich zuzutreffen. Das Verhalten dieser Quantoren wird deshalb gierig (*greedy*) genannt.

Beispiel 5

(a) Muster: `/\d+/`
Text: 0, 42, 2017 aber auch 000

(b) Muster: `/[0-4]{2,4}/`
Text: 0, 42, 2017 aber auch 000

(c) Muster: `/b*/`
Text: |abb|a

(d) Muster: `/b?/`
Text: |abbbbbb|c

(e) Muster: `/[bc]+/`
Text: aabcbbbccdd

Platzhalter

Das Metasymbol `.` steht für ein beliebiges Zeichen und muss wegen seiner „Mächtigkeit“ besonders in Kombination mit Quantoren mit Umsicht verwendet werden.

Beispiel 6

(a) Muster: `/Re.e/`
Text: Rebe Rede Rehe

(b) Muster: /.*/
Text: Hier wird alles genommen.

Alternative

Die weiter oben beschriebenen Zeichenklassen ermöglichen die Auswahl jeweils *eines* Symbols aus einer Gruppe von Zeichen. Dieses Prinzip kann auch auf ganze reguläre Ausdrücke erweitert werden. Dazu gibt es den Operator `|` für die nichtausschliessende Oder-Verknüpfung (*Disjunktion*).

Um in einem Text nach den Mustern *Hund* oder *Katze* zu suchen, verwendet man den regulären Ausdruck

`/Hund|Katze/` *lies: Hund oder Katze*

Es können auch mehr als zwei Operanden mit dieser Operation verknüpft werden.

eine Zeichenklasse ist die Kurzform einer Disjunktion für einzelne Zeichen.

`\d` = `[0-9]` = `[0123456789]` = `0|1|2|3|4|5|6|7|8|9`

Beispiel 7

Muster: `/einer|eines|eine|ein/`
Text: Sie hörte eine Weile dem Gesang einer Lärche zu.

Eine Alternative wird von links nach rechts ausgewertet und der erste Treffer passt. Daher spielt die Reihenfolge eine Rolle, wenn eine Alternative ein *Präfix* einer anderen ist.

Gruppenbildung

Hierarchie der wesentlichen Operationen für reguläre Ausdrücke

1. Quantoren (*quantifier*)
2. Verkettung (*concatenation*)
3. Alternative (*disjunction*)

Um auch reguläre Ausdrücke zu formulieren, die von von dieser Rangordnung abweichen, werden runde Klammern `(...)` verwendet.

Beispiel 8

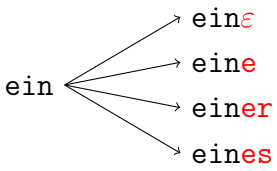
Muster: `/ein(er|es|e|)/`
Text: Sie hörte eine Weile dem Gesang einer Lärche zu.

Ein Operand kann fehlen, wenn man beispielsweise nach einem Wortstamm sucht. Dann muss dieses *leere Zeichen* ε aber wie im Beispiel am Ende stehen.

Beispiel 9

Das Resultat dieses Ausdrucks ist vermutlich unerwünscht:

Muster: `/ein(|e|er|es)/`
Text: Sie hörte eine Weile dem Gesang einer Lärche zu.



Zur besseren Verständlichkeit wird in der Grafik das Symbol ε für das leere Zeichen verwendet.

Anker

Ein Anker (*anchor*) ist ein spezielles Zeichen, das eine Zeichenkette an eine spezielle Position bindet.

- Das Zeichen `^` bezieht es sich auf den Beginn des Textes*.
- Das Zeichen `$` bezieht sich auf das Ende des Textes*.
- Das Zeichen `\b` bezieht sich auf eine Wortgrenze.
- Das Zeichen `\B` bezieht sich auf *keine* Wortgrenze.

* In der Regel kann eine Implementation für reguläre Ausdrücke mit dem *Single-Line-Modifier* `s` dazu gebracht werden, die Verankerung auf eine Zeile anstatt auf den gesamten Text anzuwenden.

Beispiel 10

(a) Muster: `/\bein\b/`
Text: Das war ihm ein wenig peinlich.

(b) Muster: `/\Bein\B/`
Text: Das war ihm ein wenig peinlich.

(c) Muster: `/^Ein/`
Text: Einmal mehr wollte er eine Einigung.

(d) Muster: `/␣+$/`
Text: Das ist ein Satz.␣␣␣

Metazeichen suchen

Wie wir oben gesehen haben, werden die Symbole

`[ ] ^ * + ? { } . | ( )`

Zur Bildung komplexer Muster verwendet. Wenn man nach diesen Zeichen selber suchen möchte, muss man ihre Metabedeutung durch einen unmittelbar davor stehenden Backslash (\) *maskieren*, d. h. aufheben.

Dasselbe Problem stellt sich natürlich auch mit dem Backslash selbst. Dieser wird einfach durch einen zweiten vorangestellten Backslash maskiert.

Beispiel 11

(a) Muster: `/\.^/`
Text: Er gab ihr Fr. 1.50 zurück.

(b) Muster: `/\[/`
Text: a + [+ b + c + d]

(c) Muster: `/[\.\?,]/`
Text: War das so? Ja, das war so.

(d) Muster: `/\\d/`
Text: \d steht für eine Ziffer.

Register

In gewissen Situationen soll sich ein regulärer Ausdruck daran „erinnern“, welche konkreten Zeichen er bereits vorher erkannt hat.

Hier kommen wieder die runden Klammern zum Zuge. Sie dienen nicht nur der Zusammenfassung von Ausdrücken sondern speichern auch den erkannten Inhalt in der Reihenfolge ihres Auftretens in Registern mit den Namen `\1`, `\2`, ...

Möchte man eine Folge von zwei gleichen Ziffern (00, 11, 22, ..., 99) erkennen, geht das mit folgendem regulären Ausdruck:

```
/([0-9])\1/
```

Wenn der erste Ausdruck eine einzelne Ziffer findet, so wird sie im Register `\1` gespeichert. Folgt unmittelbar darauf das Zeichen, das in diesem Register steht, erkennt der Ausdruck die Zeichenfolge.

Beispiel 12

(a) Muster: `/(\w)(\w)\2\1/`
Text: anna badet bei ebbe.

(b) Muster: `/(\w)\1+/`
Text: Er griff zum Massstab.

(c) Muster: `/\b([A-Z])\w+\b\s+\b\1\w+\b/`
Text: Kinder mögen Donald Duck und Micky Maus.

Der letzte Ausdruck erkennt einfache Alliterationen, sofern sie mit einem Grossbuchstaben beginnen.

Streng genommen gehören Register nicht zum theoretischen Konzept regulärer Ausdrücke. Bei der praktischen Anwendung sind sie jedoch hilfreich.