Visualisierung regulärer Ausdrücke

Ein Vortrag von Martin Häcker und Felix Schwarz

Was uns beschäftigte...

```
^\(IBOutlet[ ]*\)*\([_a-zA-Z][_a-zA-Z0-9]*\)[ ]*\(\**\)[ ]*\([_]*\)\([a-zA-Z]\)\([_a-zA-Z0-9]*\)[ ]*;/\2/
^\(IBOutlet[ ]*\)*\([_a-zA-Z][_a-zA-Z0-9]*\)[ ]*\(\**\)[ ]*\([_]*\)\([a-zA-Z]\)\([_a-zA-Z0-9]*\)[ ]*;/\3/
^\(IBOutlet[ ]*\)*\([_a-zA-Z][_a-zA-Z0-9]*\)[ ]*\(\**\)[ ]*\([_]*\)\([a-zA-Z]\)\([_a-zA-Z0-9]*\)[ ]*;/\4/
^\(IBOutlet[ ]*\)*\([_a-zA-Z][_a-zA-Z0-9]*\)[ ]*\(\**\)[ ]*\([_]*\)\([a-zA-Z]\)\([_a-zA-Z0-9]*\)[ ]*;/\5/
^\(IBOutlet[ ]*\)*\([_a-zA-Z][_a-zA-Z0-9]*\)[ ]*\(\**\)[ ]*\([_]*\)\([a-zA-Z]\)\([_a-zA-Z0-9]*\)[ ]*;/\6/
```

Findet die Variablendeklarationen in Objective-C Programmen

Das Problem

- „Some people, when confronted with a problem, think »I know, I'll use regular expressions.« Now they have two problems.“
  Jamie Zawinski, in comp.lang.emacs (from the fortune file)

Was sind reguläre Ausdrücke?

- flexibles Werkzeug zum Parsen von Text
- Syntax nicht ganz einheitlich, hier Verwendung der Syntax von Perl 5
- lange in Gebrauch, erste Ansätze 1941
- selbst komplexe Muster werden in einer Zeile kodiert

kleine Syntaxkunde (I)

.	ein beliebiges Zeichen
\d	eine Ziffer (also 0 bis 9)
\w	ein alphanumerisches Zeichen
\s	ein Leerzeichen oder ein Tabulator

kleine Syntaxkunde (II)

\d+	ein oder mehrere Ziffern
\d*	keine oder beliebig viele Ziffern
\d{3}	genau drei Ziffern
\d{2,5}	2 bis 5 Ziffern

kleine Syntaxkunde (III)

(a-z)	erkennt die Zeichen 'a' bis 'z'
(Uni|Schule)	erkennt die Strings 'Uni' und 'Schule' (Alternative)
[abc]	erkennt die Zeichen 'a', 'b', 'c' Sonderzeichen werden maskiert, z.B: \\.

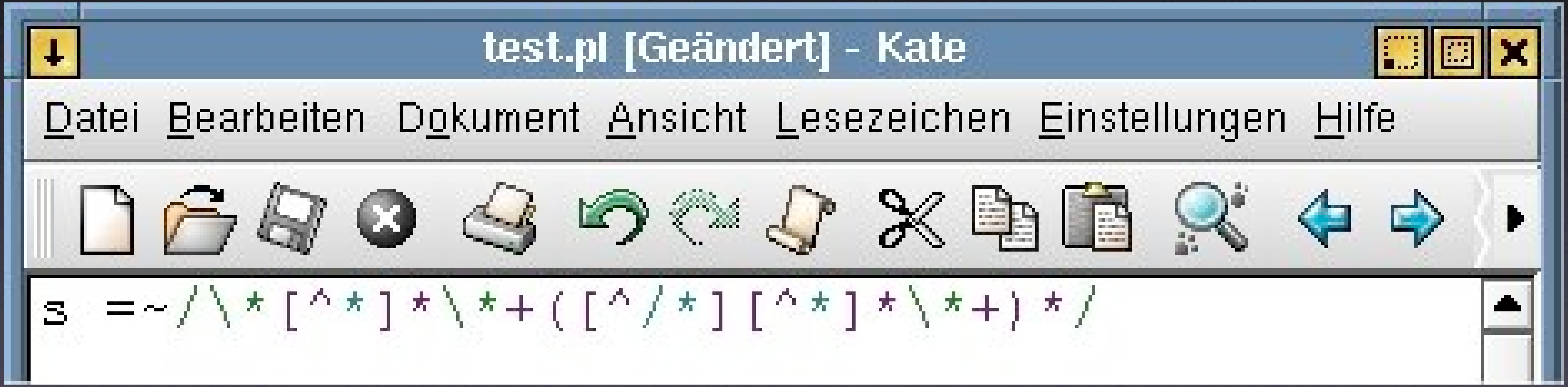
kleine Syntaxkunde (IV)

(\d\d),(\d\d) Euro	Cent-Betrag in $2, Euro-Betrag in $1
(\d\d),\1 Euro	Beträge, bei denen Cent=Euro sind

Darstellungsprobleme, kurz analysiert

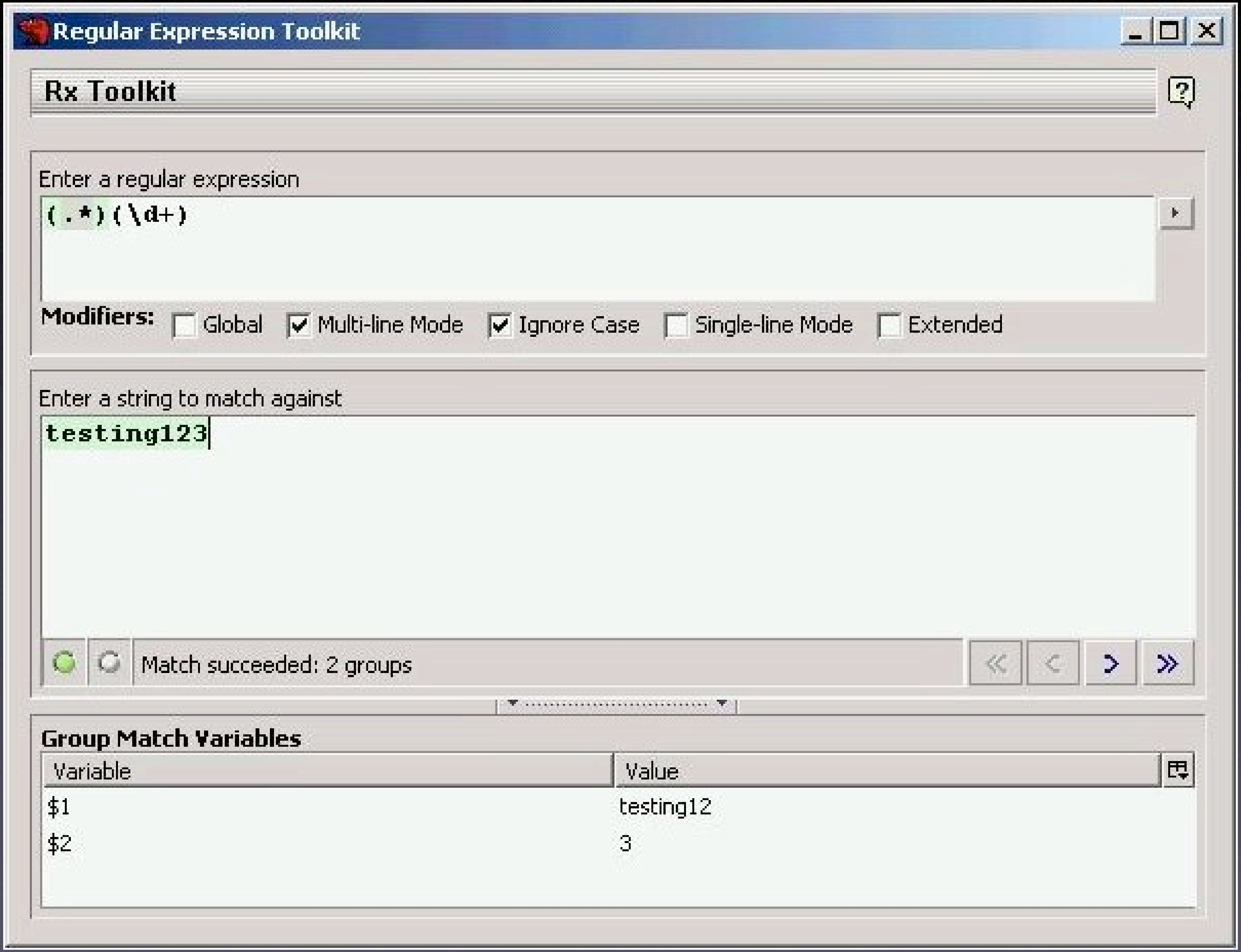
Existierende Werkzeuge

Kate


test.pl [Geändert] - Kate
Datei Bearbeiten Dokument Ansicht Lesezeichen Einstellungen Hilfe

```
s =~/\*[^*]*\*+([^/*][^*]*\*+)*/
```

Komodo



Regular Expression Toolkit

Rx Toolkit

Enter a regular expression

```
(.*)(\d+)
```

Modifiers: ☐ Global ☑ Multi-line Mode ☑ Ignore Case ☐ Single-line Mode ☐ Extended

Enter a string to match against

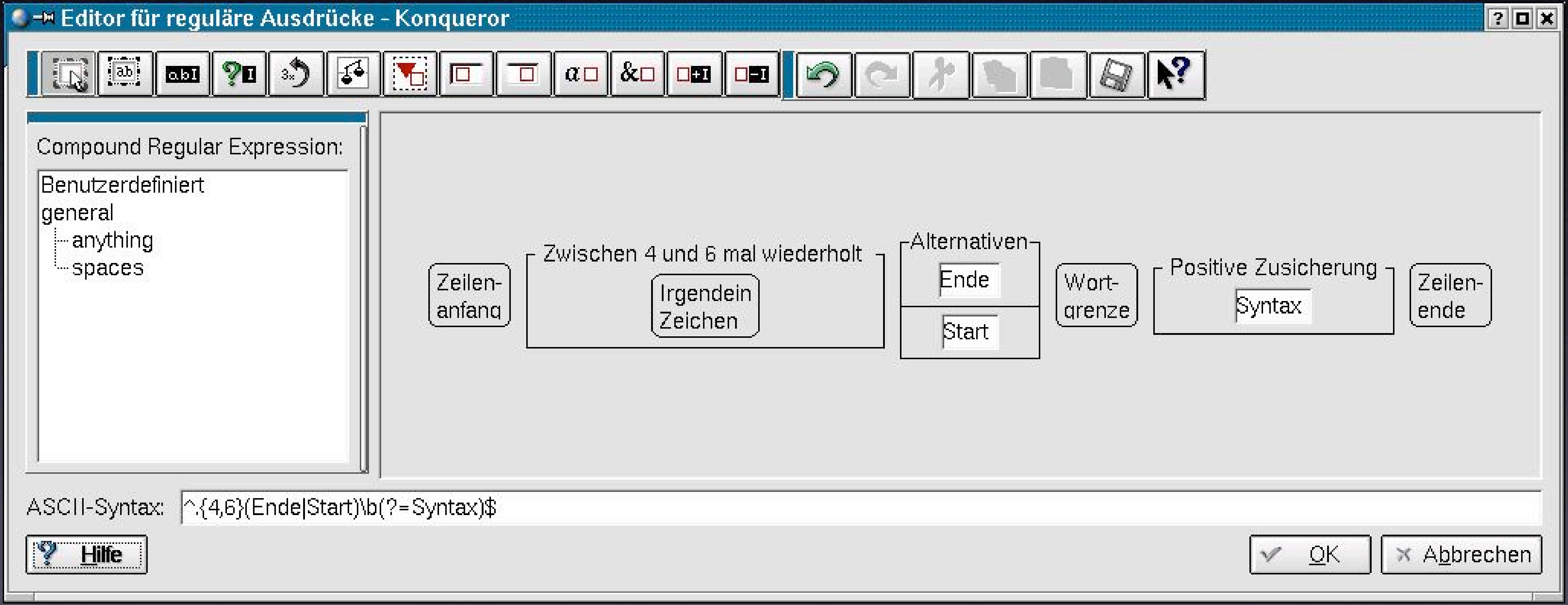
```
testing123
```

Match succeeded: 2 groups

Group Match Variables

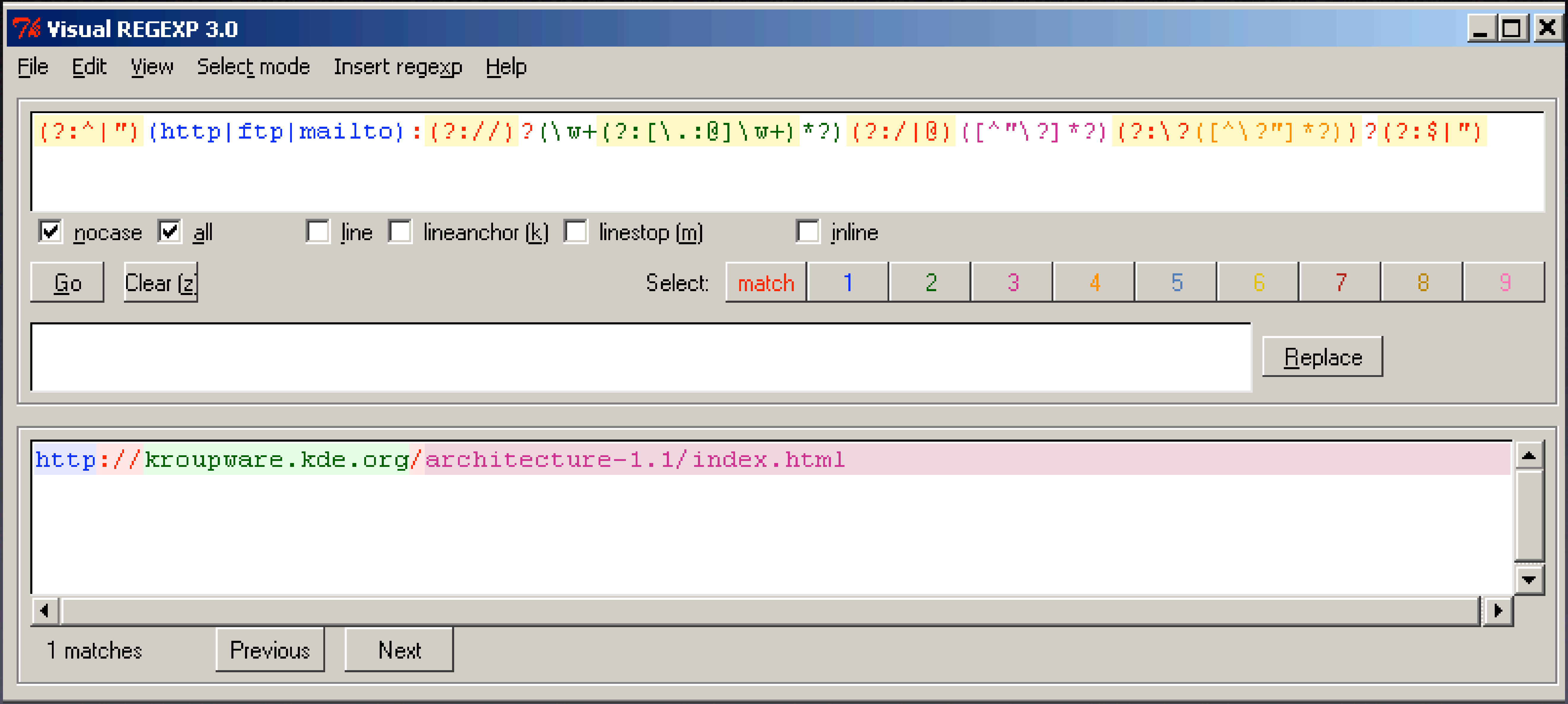
Variable	Value
$1	testing12
$2	3

kregexpeditor



ASCII-Syntax: `^.{4,6}(Ende|Start)\b(?=Syntax)$`

Visual Regexp


Visual REGEXP 3.0
File Edit View Select mode Insert regexp Help
Go
Clear (z)
Select: match 1 2 3 4 5 6 7 8 9
Replace
Previous
Next

```
(?:^|")(http|ftp|mailto):(?://)?(\w+(?:[\.:@]\w+)*?)(?:/|@)([^"\?]*?)(?:\?([^\?"]*?))?(?:$|")
```

[✓] nocase [✓] all [ ] line [ ] lineanchor (k) [ ] linestop (m) [ ] inline

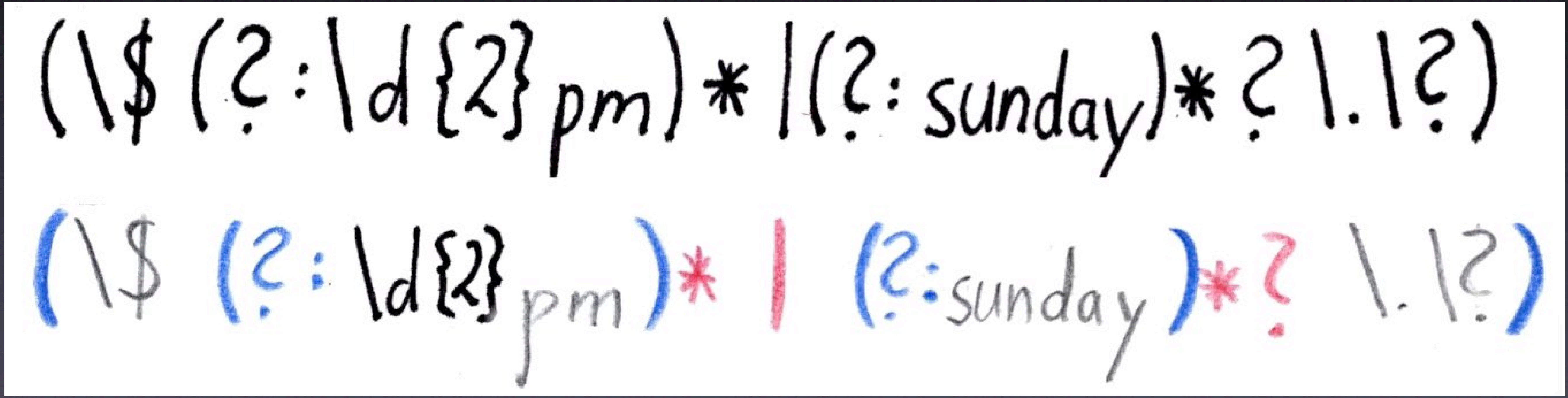
```
http://kroupware.kde.org/architecture-1.1/index.html
```

1 matches

Überblick über die Visualisierungstechniken

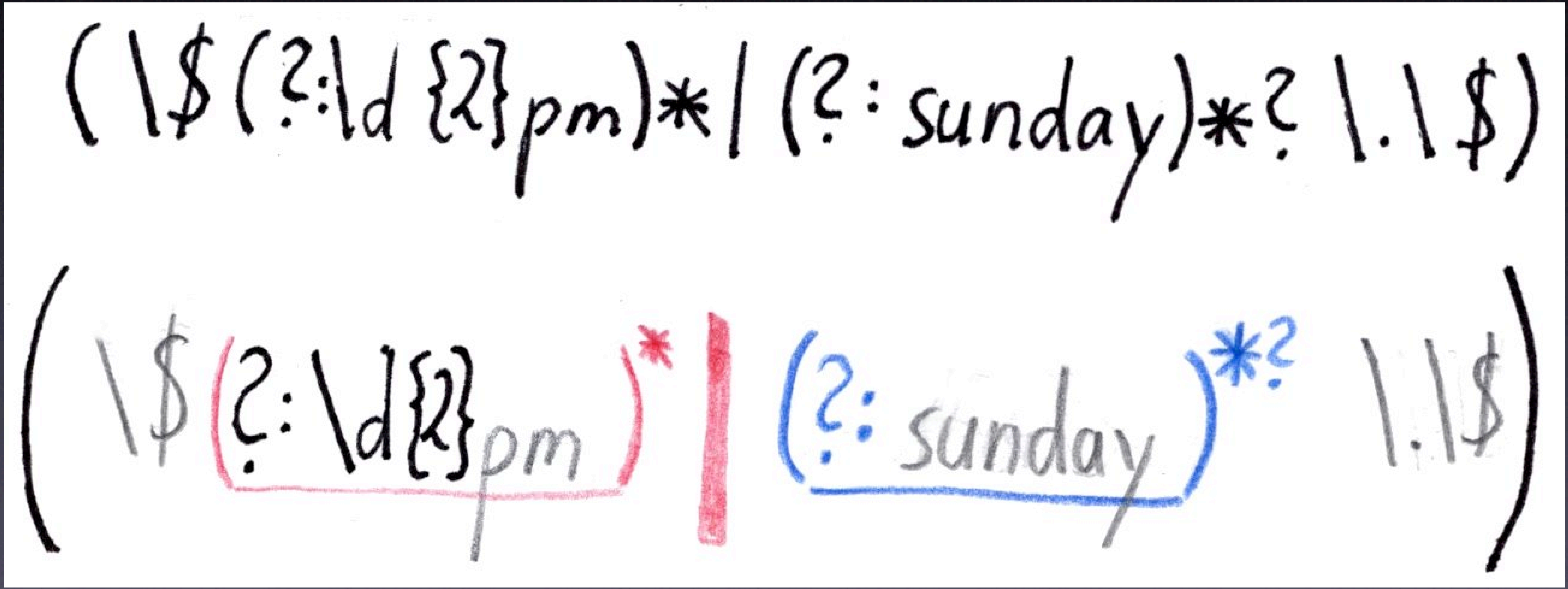
- Code-Visualisierung
- Ablauf-Visualisierung
- Ergebnis-Visualisierung
- Synergie-Effekte
- Profiling

Syntax-Highlighting



```
(\$(?:\d{2}pm)*|(?:sunday)*?\.\?)
(\$ (?:\d{2}pm)* | (?:sunday)*? \.\?)
```

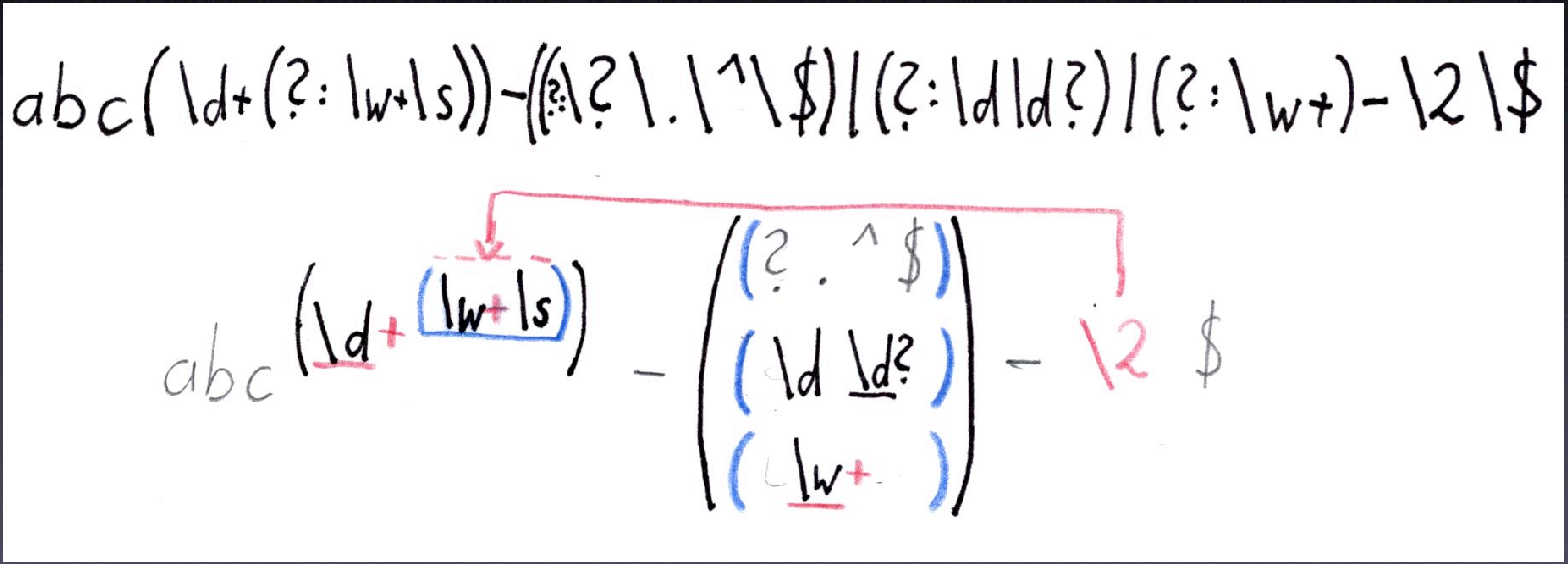
Syntax-Styling



```
(\$(?:\d{2}pm)*|(?:sunday)*?\.\$)
```

```
(\$(?:\d{2}pm)*|(?:sunday)*?\.\$)
```

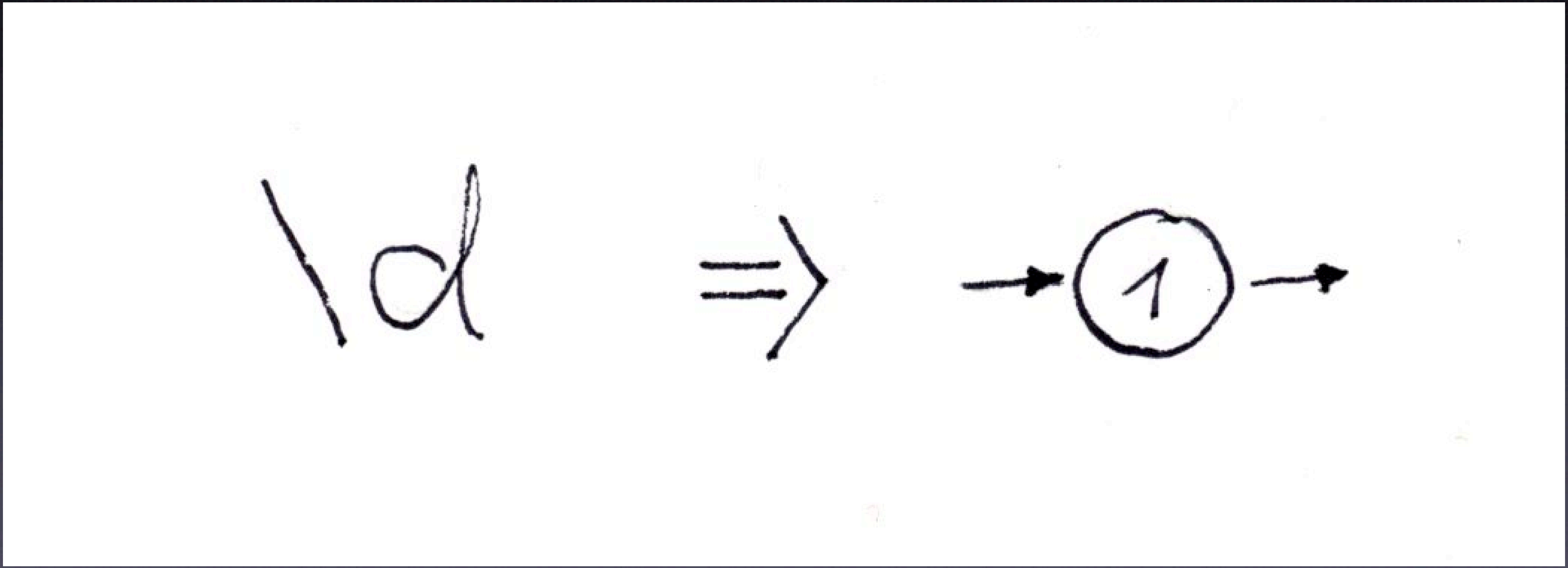
Visualisierung über Muster



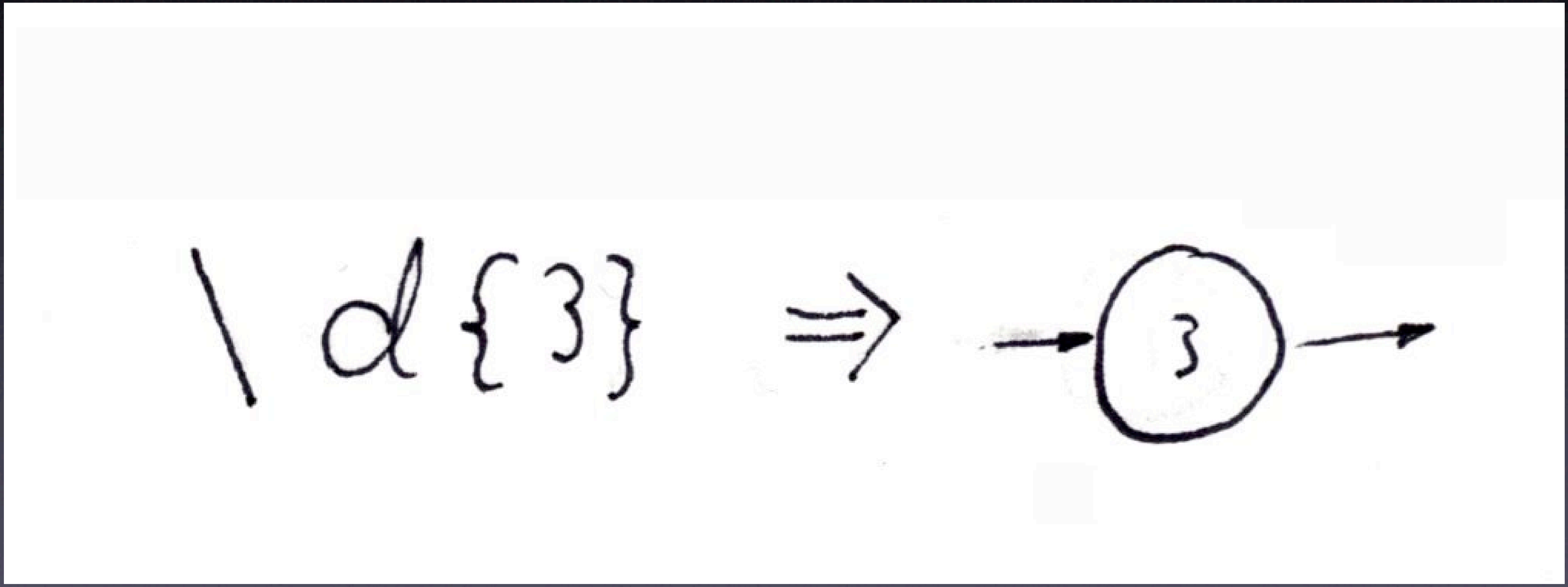
```
abc(\d+(?:\w+\s))-((?:\?\.\^\$)|(?:\d\d?)|(?:\w+)-\2\$
```

Alien-Highlighting

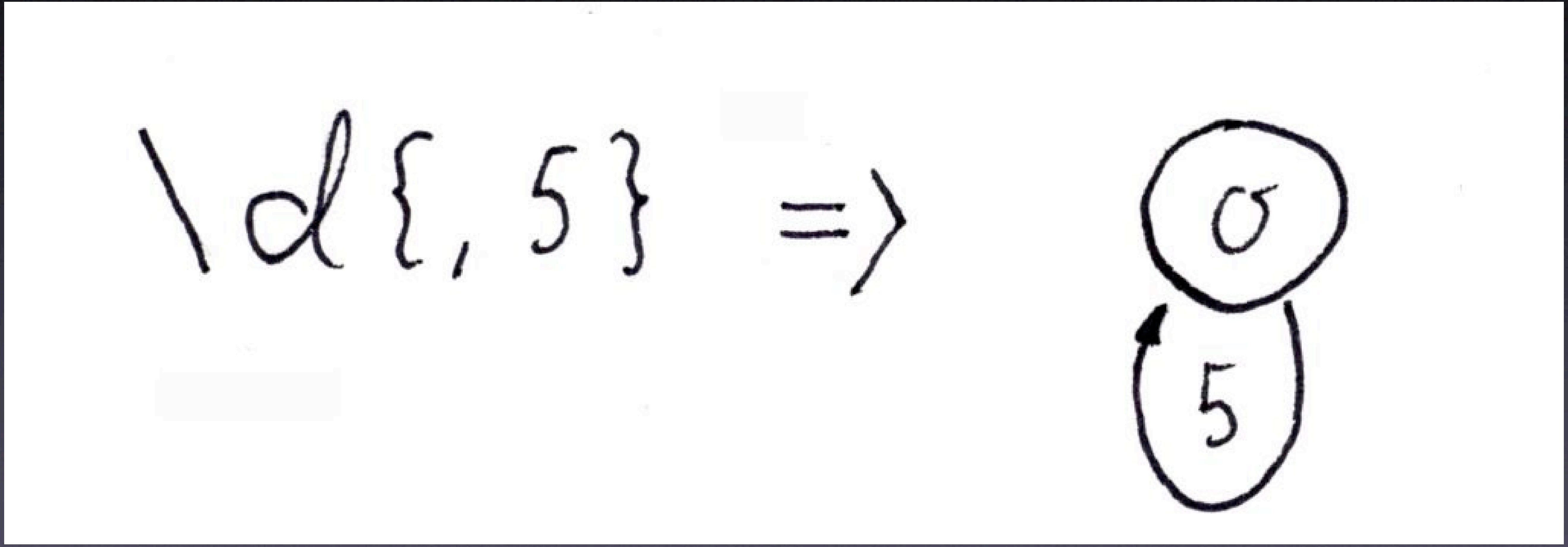
Alien-Highlighting



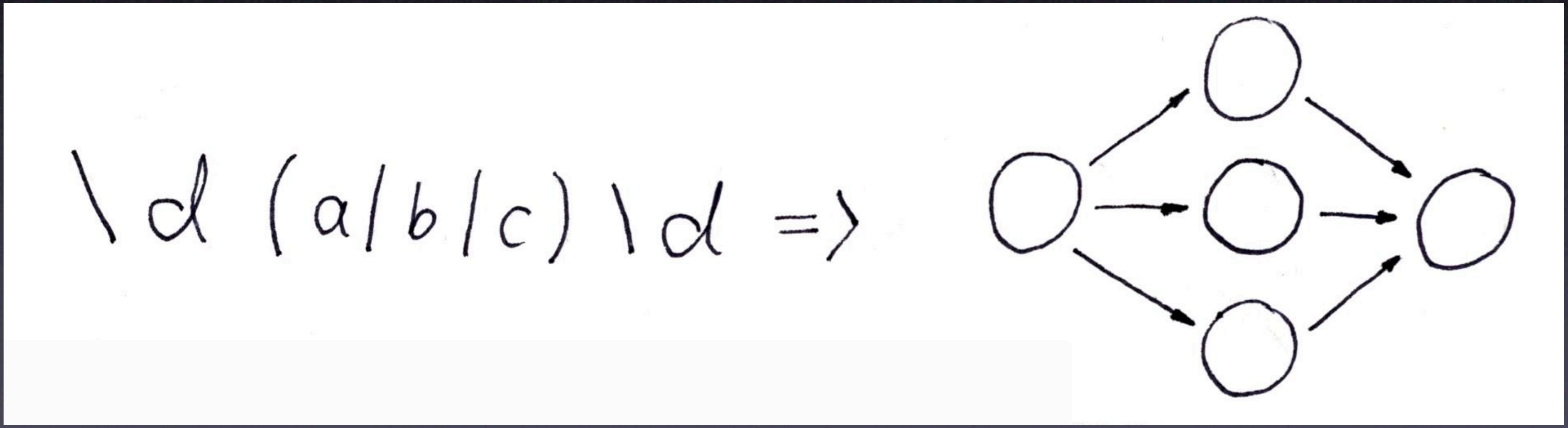
Alien-Highlighting



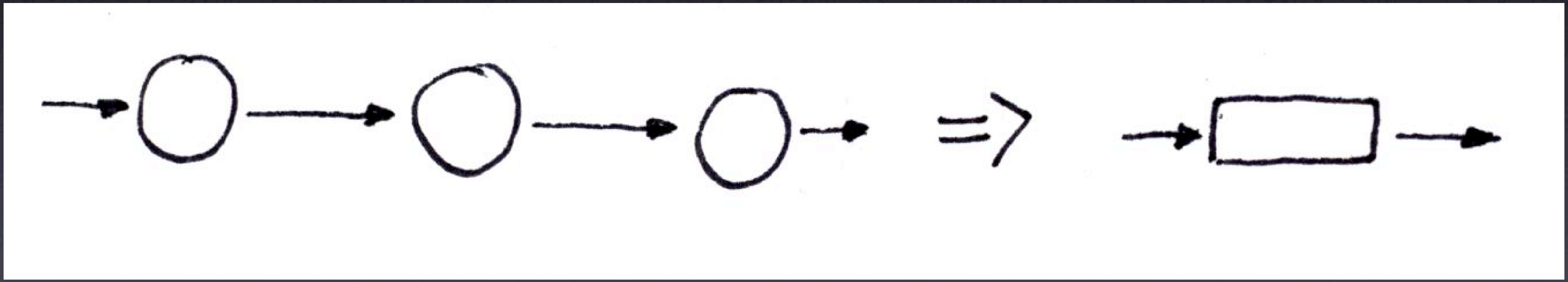
Alien-Highlighting



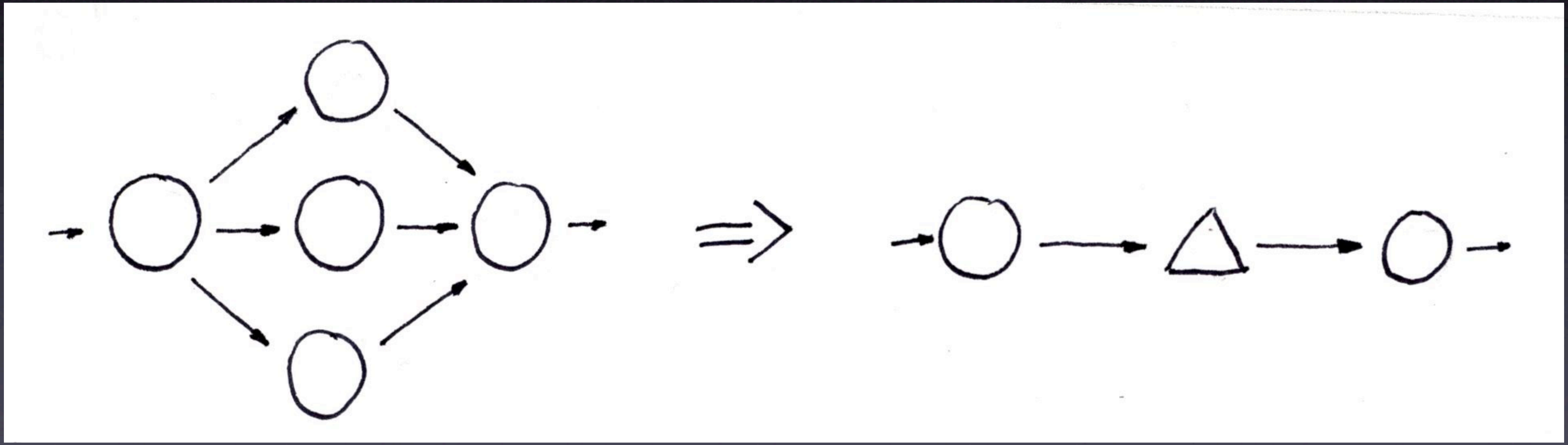
Alien-Highlighting



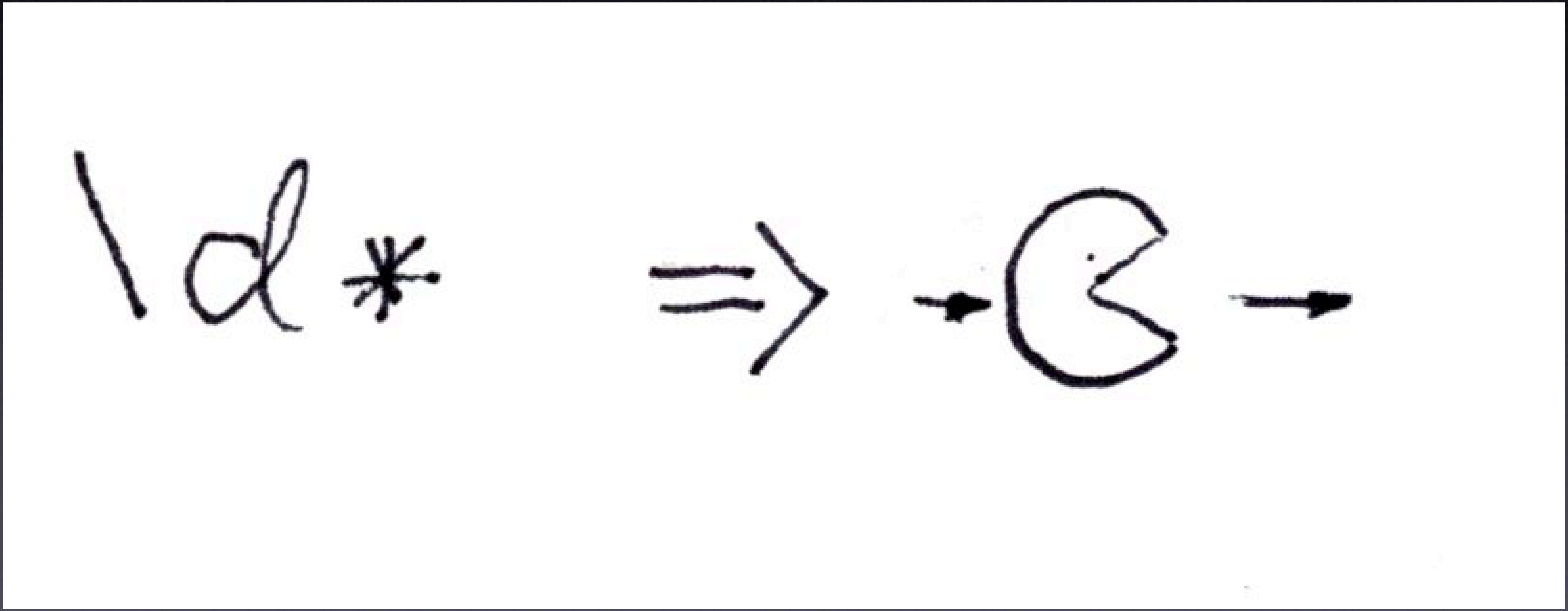
Alien-Highlighting



Alien-Highlighting



Alien-Highlighting

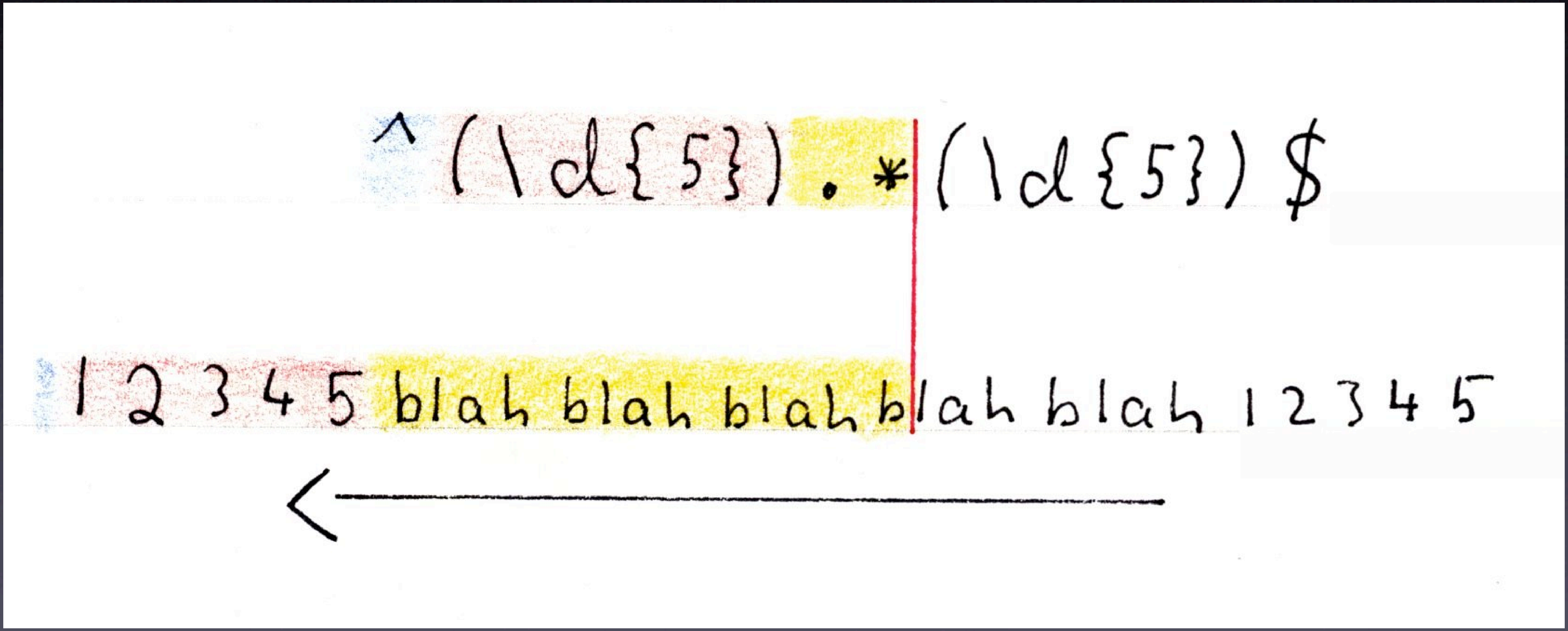


Übersetzung in natürliche Sprachen

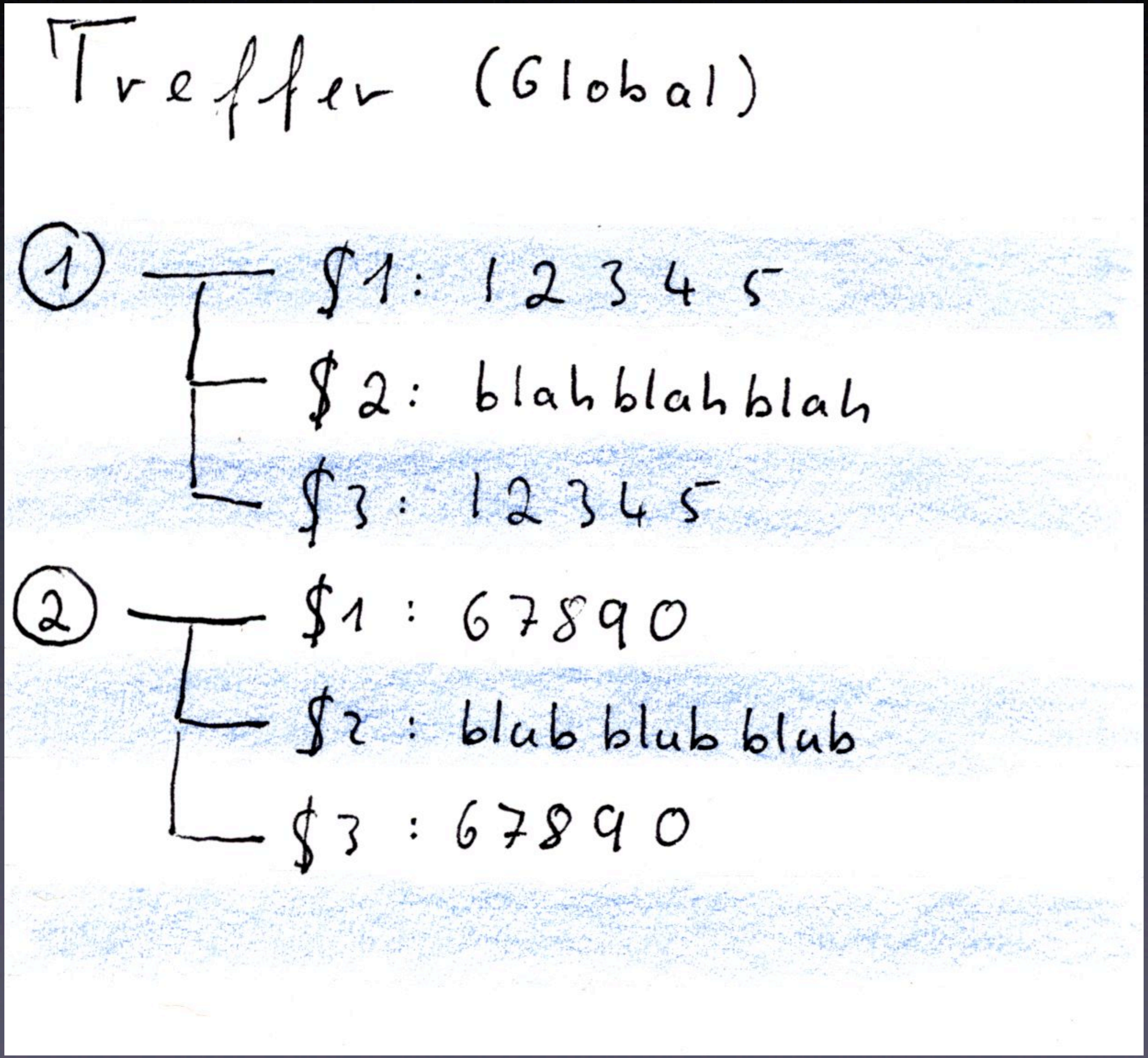
- \d => 0 1 2 3 4 5 6 7 8 9
- \x21-\x25 => ! " # $ %
- ^\d{3,5}$ => Eine Zeile die nur eine drei- bis fünfstellige Zahl enthält

Visuelle Programmierung

Ablauf-Visualisierung



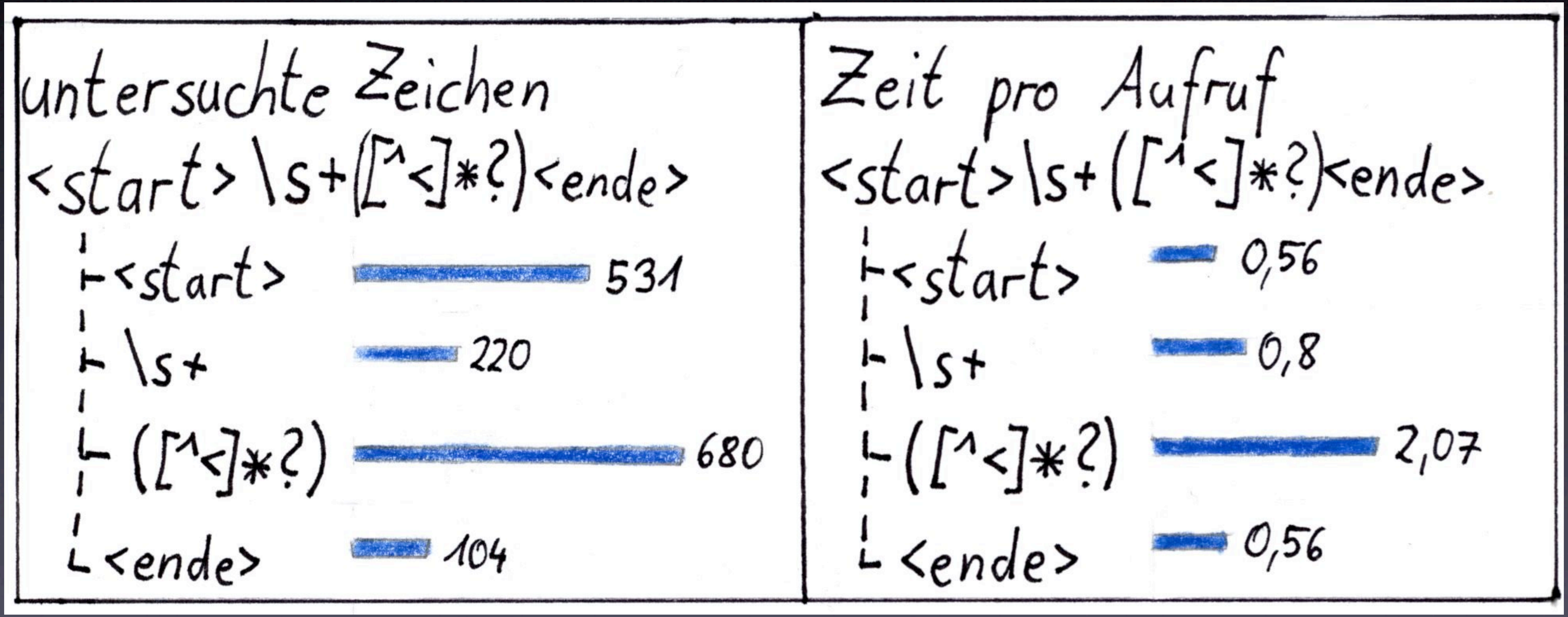
Ablauf-Visualisierung



Ergebnis-Visualisierung

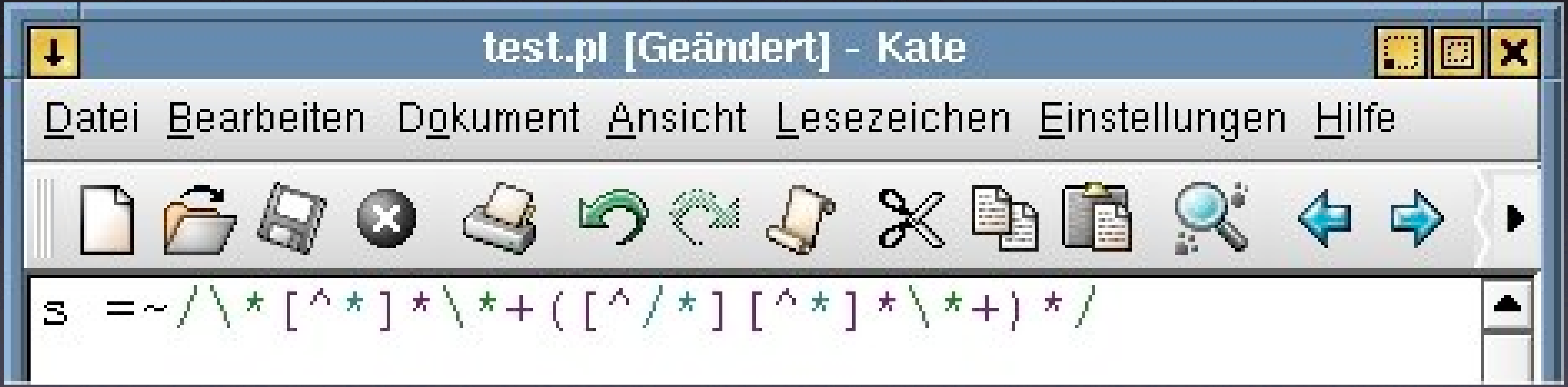
Synergie-Effekte

Profiling


untersuchte Zeichen
<start>\s+([^<]*?)<ende>
<start> 531
\s+ 220
([^<]*?) 680
<ende> 104
Zeit pro Aufruf
<start>\s+([^<]*?)<ende>
<start> 0,56
\s+ 0,8
([^<]*?) 2,07
<ende> 0,56

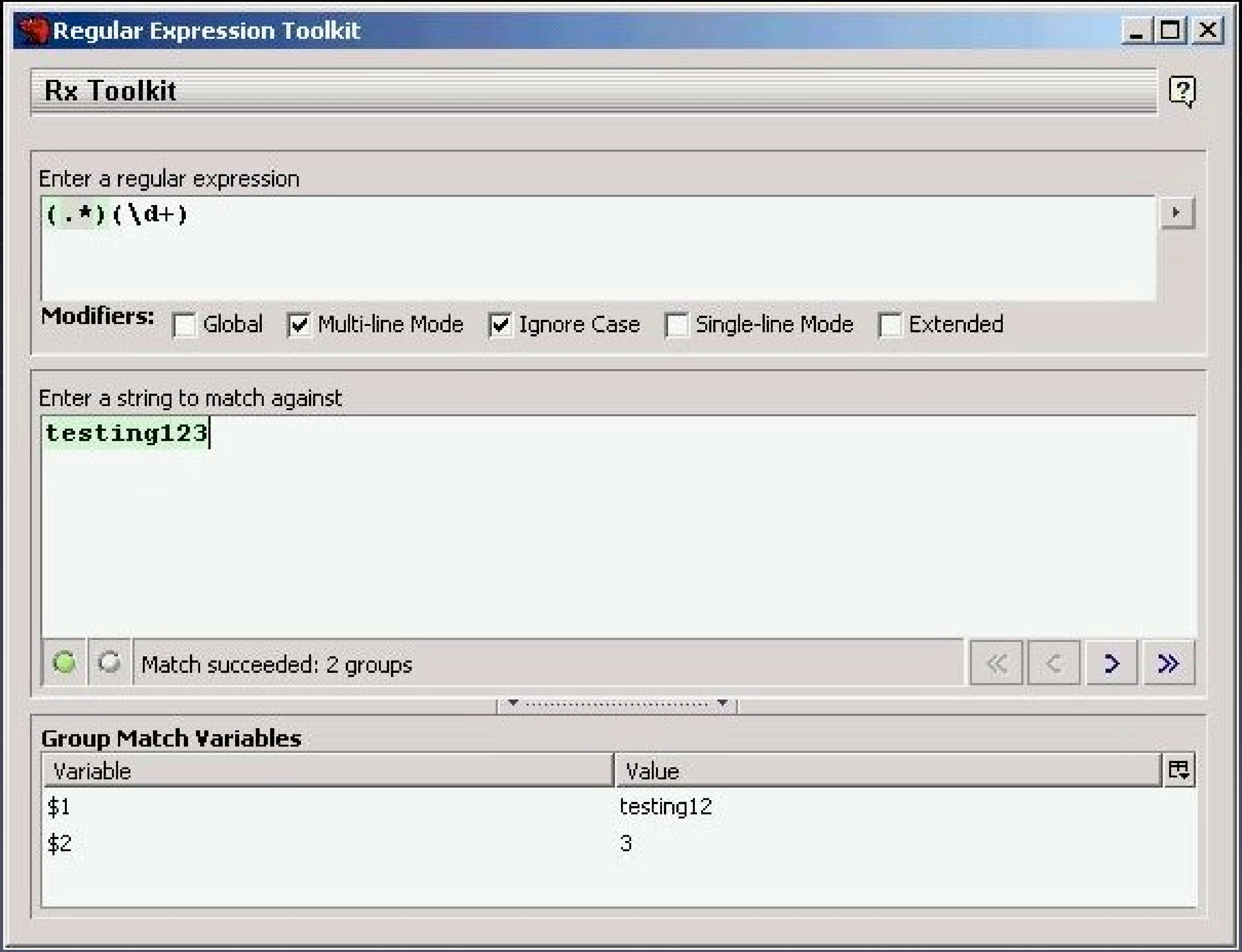
Bewertung existierender Werkzeuge

Kate


test.pl [Geändert] - Kate
Datei Bearbeiten Dokument Ansicht Lesezeichen Einstellungen Hilfe

```
s =~/\*[^*]*\*+([^/*][^*]*\*+)*/
```

Komodo



Enter a regular expression

```
(.*)(\d+)
```

Modifiers: [ ] Global [✓] Multi-line Mode [✓] Ignore Case [ ] Single-line Mode [ ] Extended

Enter a string to match against

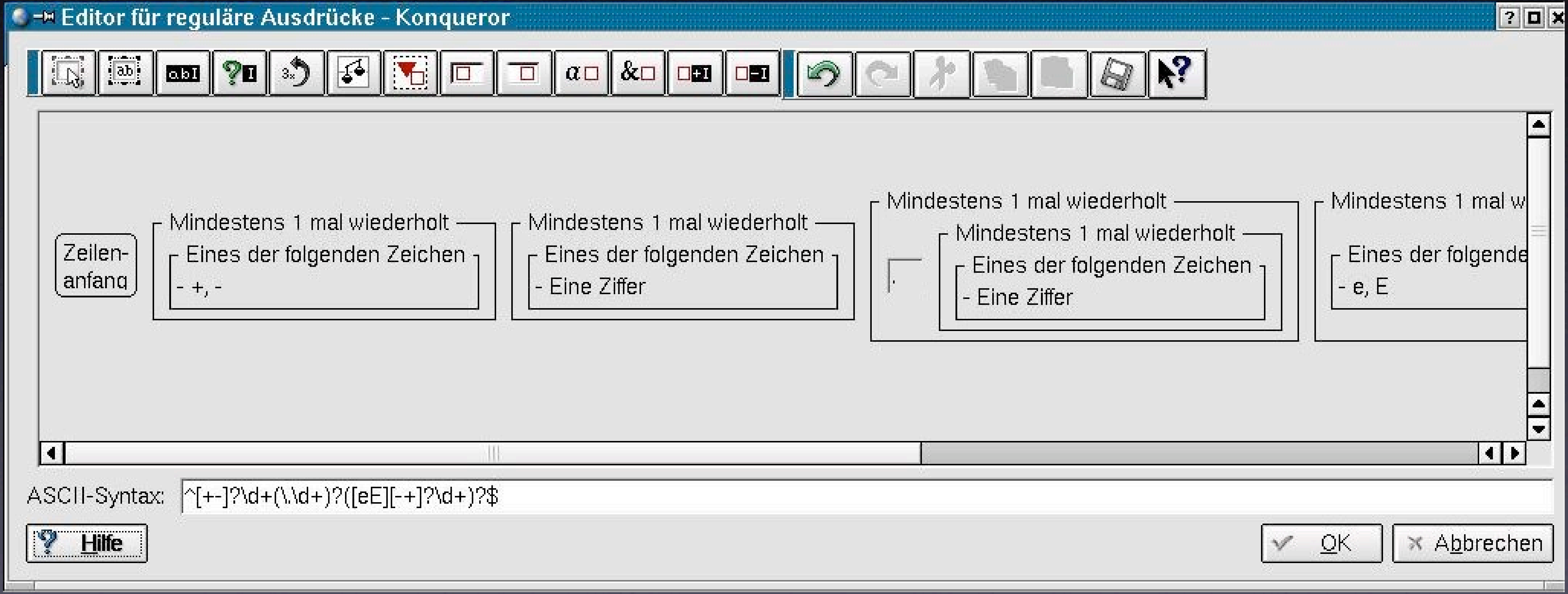
```
testing123
```

Match succeeded: 2 groups

Group Match Variables

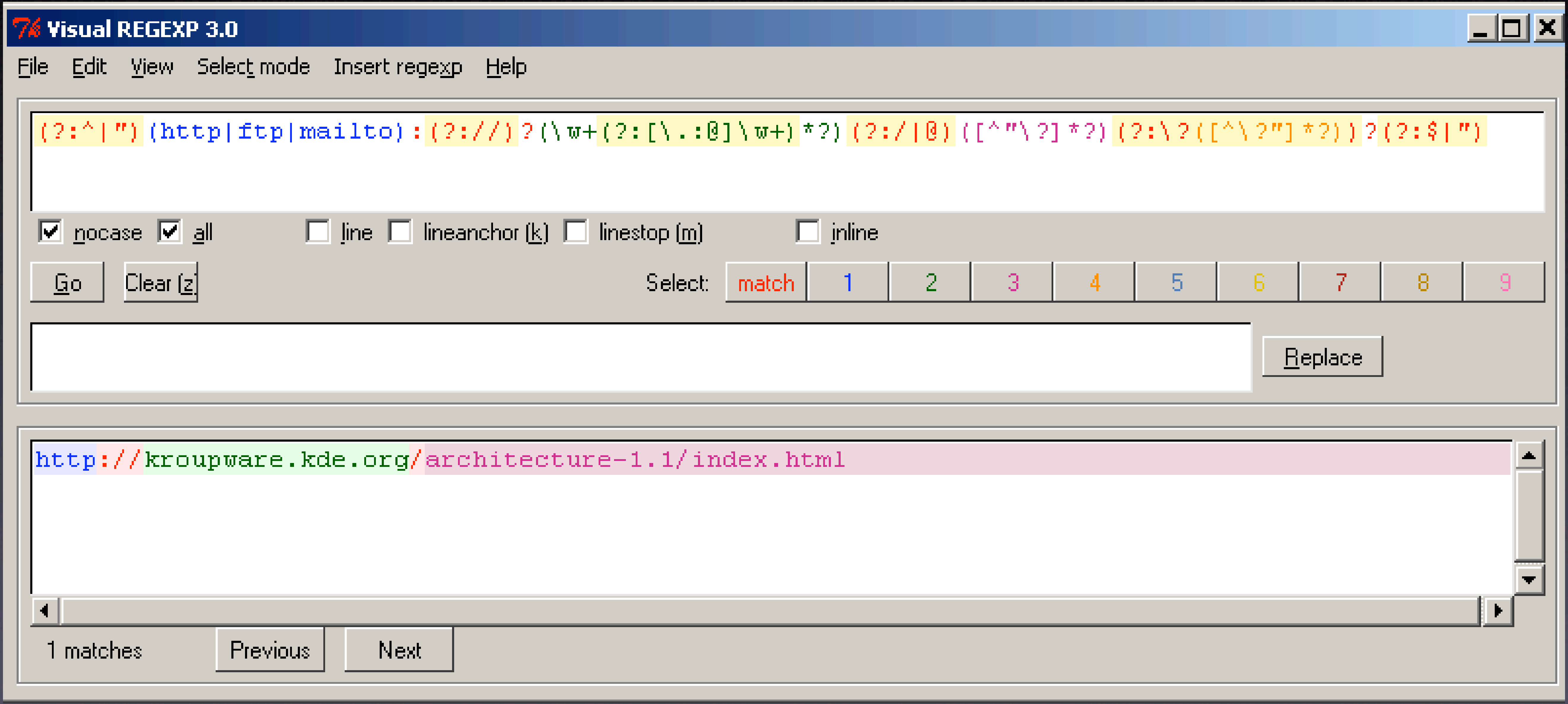
Variable	Value
$1	testing12
$2	3

kregexpeditor


Editor für reguläre Ausdrücke - Konqueror
Zeilen-anfang
Mindestens 1 mal wiederholt
Eines der folgenden Zeichen
- +, -
Mindestens 1 mal wiederholt
Eines der folgenden Zeichen
- Eine Ziffer
Mindestens 1 mal wiederholt
.
Mindestens 1 mal wiederholt
Eines der folgenden Zeichen
- Eine Ziffer
Mindestens 1 mal w
Eines der folgende
- e, E
Hilfe
OK
Abbrechen

ASCII-Syntax: `^[+-]?\d+(\.\d+)?([eE][-+]?\d+)?$`

Visual Regexp



```
(?:^|")(http|ftp|mailto):(?://)?(\w+(?:[\.:@]\w+)*?)(?:/|@)([^"\?]*?)(?:\?([^\?"]*?))?(?:$|")
```

[✓ nocase] [✓ all] line lineanchor (k) linestop (m) inline

```
http://kroupware.kde.org/architecture-1.1/index.html
```

1 matches

Fazit

Was hier keinen Platz mehr hatte…

Herzlichen Dank!