RA - Einführung

- **Reguläre Ausdrücke** sind eine (*privimitive?*) **Programmiersprache**, die das **Aussehen von Texten** (*Zeilen, Strings*) beschreibt, indem sie festlegen
  - welche **Zeichen**
  - in welcher **Reihenfolge**
  - mit welcher **Häufigkeit**
  - unter welchen **Randbedingungen**

  darin vorkommen sollen.
- Mit Regulären Ausdrücken können im wesentlichen **Texte gesucht**, **umgeformt** und **gelöscht** werden.

RA - Einführung

- Neben **normalen Zeichen** kennen sie noch **Metazeichen** (*z.B.* `. * ? [ ] ^ $`), die nicht für sich selbst, sondern für **spezielle Funktionen** stehen.
- Viele **UNIX-Programme** verwenden Reguläre Ausdrücke zur Textsuche (*und Textbearbeitung*):
  - Suchprogramme: `grep`, `egrep`, `find`
  - Editoren: `ed`, `ex`, `vi`, `emacs`
  - Stream-Editor: `sed`
  - Skript- und Programmiersprachen: `awk`, `perl`, `php`, `python`, `ruby`, `tcl/tk`, `lua`, `java`, `C`, `C++`, ...
  - Server: `apache`, `sendmail`, `postfix`

RA - Einführung

- Da die Programme, die Reguläre Ausdrücke verwenden, **zu verschiedenen Zeitpunkten entstanden**, gibt es (*leider*) unterschiedliche Arten:
  - **Standard / Basic RA** (**BRE**): Mindestsatz, überall gleich.
  - **Erweiterte /Extended RA** (**ERE**): Erweiterungen, nicht überall vorhanden, nicht überall gleich.
  - **Perl** bietet eine **Obermenge** aller bekannten Möglichkeiten.
  - Dokumentation dazu erhält man durch:

    `man 7 regex` (**Linux**)

    `man -s 5 regex` (**Solaris**, **HP-UX**)

RA - Einführung

Achtung

- Bitte das sogenannte "**Wildcard Pattern Matching**" der Shell für **Dateinamen** nicht mit den Regulären Ausdrücken verwechseln:
  - Einfacher und weniger leistungsfähig.
  - Für **Dateinamen** gedacht, nicht für **Texte**.
  - **Metazeichen**: `* ? [abc] [^abc] [!abc]`
  - Teilweise die gleichen Metazeichen.
  - *Aber teilweise eine völlig andere Wirkung.*
  - Suchen automatisch **vollständige** Dateinamen.

RA - Einführung

Begriffe

- Reguläre Ausdrücke (*engl.* **Regular Expressions** = **RegExp**, **RE**, **RA**) heißen auch **Pattern** (**Muster**).
- Wenn ein Regulärer Ausdruck auf einen Text **paßt**, dann "**matcht**" er diesen Text.
- **Konkatenation** (**Sequenz**, **Verkettung**) hängt Zeichen zu einer Zeichenkette aneinander `a•b•c` (***impliziter Operator*** *in Regulären Ausdrücken*).
- Ein **Literal** ist ein Zeichen, das für sich selbst steht.

RA - Besonderheiten

Achtung!

- Wird ein Kommando mit Argumenten aufgerufen, dann werden die **Metazeichen** in den Parametern **zuerst von der Shell interpretiert** und erst danach das Ergebnis an das Kommando übergeben.

*Sollen also **Metazeichen** (wie sie z.B. in Regulären Ausdrücken vorkommen) an ein Kommando übergeben werden, so sind sie vor der Shell durch **Quotieren** mit \C, "..." und '...' zu **schützen**.*

RA - Besonderheiten

Beispiel

– Das Kommando (*allgemein* `grep` ***`MUSTER`*** `DATEI...`)

```
grep [abc]* kap*
```

wird von der Shell folgendermaßen **expandiert**, wenn die Dateien `array`, `bug`, `comp`, `kap1` und `kap2` im aktuellen Verzeichnis vorhanden sind:

```
grep array bug comp kap1 kap2
```

– D.h. es wird das **Muster** `array` in 4 Dateien gesucht.

– Um das **Muster** daher **Anführungszeichen** setzen:

```
grep "[abc]*" kap*     oder
grep '[abc]*' kap*
```


RA - Besonderheiten

Besonderheiten von Regulären Ausdrücken

- **Jedes** Zeichen hat eine Bedeutung (*auch Leerzeichen und Tabulator*).
- Der **erste** passende Text wird gefunden (**left-most**).

  `xx*` findet `abcxxxdefxxghixxxxxjklmn`

  – Kann in Perl auch umgekehrt werden (**right-most**).
- Der **längste** passende Text wird gefunden (**greedy**).

  `xx*` findet nicht `abcxxxxdef` sondern `abcxxxxdef`

  – Kann in Perl auch umgekehrt werden (**non-greedy**)

RA - Besonderheiten

Besonderheiten von Regulären Ausdrücken

- Normalerweise muß nur ein **Teil des Textes** passen (*Unterschied zur* ***Dateinamen-Expansion*** *der Shell*).

  – D.h. sind nicht automatisch "**verankert**" (*durch spezielle Metazeichen aber erzwingbar*).

  – Trifft ein Regulärer Ausdruck **nicht am Textanfang** zu, dann wird ein Zeichen weiter **erneut probiert** bis zum Ende des Textes (*aufwendige Suche!*)

  `xx*` findet `abcxxxxxxxxxdef`

  `^xx*` findet in `abcxxxxxxxxxdef` nichts

RA - Besonderheiten

Besonderheiten von Regulären Ausdrücken

- **Zeilenvorschub** "`\n`" kann *nicht* gesucht werden (*außer durch* `perl`*-Tricks*).
- **Quotierung** eines Metazeichens *C* ist nur durch `\`*C* möglich, aber *nicht* mit "..." oder '...' (*z.B.* `\\`).
- Einige **normale Zeichen** werden durch ein "`\`" davor zu einem Metazeichen (*z.B.* `\s` `\d` `\w` `\b`).
- **Keine vollständige Programmiersprache:**
  – "**Klammerngebirge**" nicht erkennbar (*kann nicht rechnen*).
  – **Negation** eines Musters oft nicht ausdrückbar.

RA - Standard-Metazeichen

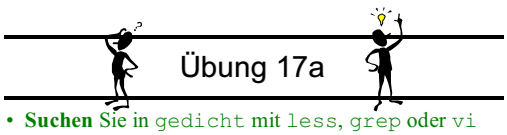
Standard-Metazeichen (*immer verfügbar*)

`.`	***Ein* beliebiges Zeichen** (*außer Zeilenende* `\n`)
`C*`	**0-unendlich** Wiederholungen des Zeichens `C`
`^`	**Zeilenanfang** (*nur am Anfang des Musters*)
`$`	**Zeilenende** (*nur am Ende des Musters*)
`[abc]`	*Ein* Zeichen aus **Zeichenmenge** `abc` (*kein , !*)
`[a-z]`	*Ein* Zeichen aus **Zeichenmenge** `a, b, ..., z`
`[^abc]`	**Ein** Zeichen **nicht** aus der Zeichenmenge `abc`. (*d.h. eines der anderen 253 ASCII-Zeichen*)
`\C`	Sonderbedeutung des Metazeichens `C` aufheben

RA - Standard-Metazeichen

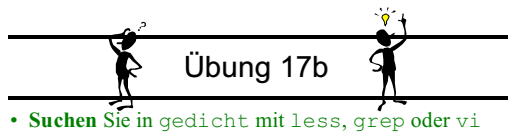
Beispiel

`made`	Zeile enthält die Zeichenfolge `made`
`.ade`	Zeile enthält `ade` mit mind. einem Zeichen davor
`xx*y`	Zeile enthält bel. viele `x` (*mind. 1*×) direkt vor `y`
`^made`	Zeile **beginnt** mit `made`
`made$`	Zeile **endet** mit `made`
`^...$`	Zeile enthält **genau** 3 beliebige Zeichen
`[A-Za-z]`	Zeile enthält einen Buchstaben
`[0-9]`	Zeile enthält eine Ziffer
`[^aeiou]`	Zeile enthält ein Zeichen, das **kein** Vokal ist



Übung 17a

- **Suchen** Sie in `gedicht` mit `less`, `grep` oder `vi` per Regulären Ausdrücken nach Zeilen, die:
  - Text `made` enthalten.
  - Text `made` am **Anfang** / **Ende** enthalten.
  - **Nur** aus dem Text `made` / `MADE` bestehen.
  - Text `made` in beliebiger **Groß/Kleinschreibung** enthalten.
  - **2**× Text `made` enthalten.
  - Text `made` am **Anfang und am Ende** enthalten.
  - **Genau 6 Buchstaben** umfassen.



Übung 17b

- **Suchen** Sie in `gedicht` mit `less`, `grep` oder `vi` per Regulären Ausdrücken nach Zeilen, die:
  - Ein (*oder mehr*) **Leerzeichen** am Ende enthalten.
  - **Leer** sind (*erst eine Definition dafür überlegen!*).
  - **Nur Großbuchstaben** enthalten.
  - **Kein einziges** `a` oder `e` enthalten.
    - Analog Zeilen, die auch kein `A` oder `E` enthalten.
  - **Genau ein** `o` (*Oh*) enthalten.
    - Analog Zeilen, die **genau zwei** `o` enthalten.
    - Analog Zeilen, die **genau drei** `o` enthalten.

RA - Erweiterte Metazeichen

Erweiterte Metazeichen (*nicht immer verfügbar*)

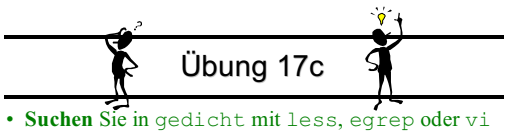
C`?`	**0** / **1** Wiederholung des Zeichens *C* **(optional)**
C`+`	**1-unendlich** Wiederholungen des Zeichens *C*
L`|`*R*	Linker Teil *L oder* rechter Teil *R* **(Alternative)**
`(...)`	Klammerung mehrerer Zeichen (*für* `*` `+` `?`)
C`\{`*M*`,`*N*`\}`	***M-N*-malige** Wiederholung des Zeichens *C*
C`\{`*M*`,\}`	***M*-unendlich** Wiederholungen des Zeichens *C*
C`\{`*M*`\}`	***M*-malige** Wiederholung des Zeichens *C*

- Der Backslash vor `\{` und `\}` ist eventuell wegzulassen.

RA - Erweiterte Metazeichen

Beispiel

`abc?ba`	Zeile mit `abba` oder `abcba` paßt
`x+y`	Zeile enthält bel. viele `x` (*mind. 1*×) direkt vor `y`
`(ab|yz)`	Paßt auf Zeilen mit `ab` oder `yz`
`x\{5,7\}`	**Zwischen** 5 und 7 `x` hintereinander
`x\{5,\}`	**Mindestens** 5 `x` hintereinander
`x\{5\}`	**Genau** 5 `x` hintereinander
`xxxxx`	**Genau** 5 `x` hintereinander



Übung 17c

- **Suchen** Sie in `gedicht` mit `less`, `egrep` oder `vi` per **erweiterten** Regulären Ausdr. nach Zeilen, die:
  - Das Wort `katze` oder `kate` enthalten.
  - Ein oder mehrere "=" direkt hintereinander enthalten.
  - Mindestens **zwei gleiche Vokale** `aa`, `ee`, `ii`, `oo`, `uu` **direkt hintereinander** enthalten.
  - Text `made` am **Anfang oder am Ende** enthalten.
  - **60**× das Zeichen "=" direkt hintereinander enthalten.

RA - Weitere Metazeichen

Weitere Metazeichen (*nicht immer verfügbar*)

`\<`	**Wortanfang** (*Übergang Leerzeichen → Zeichen*)
`\>`	**Wortende** (*Übergang Zeichen → Leerzeichen*)

Metazeichen für Suchen + Ersetzen

`\(...\)`	Muster zur späteren Verwendung **merken** (*bis zu 9 Speicher möglich, die Reihenfolge der Klammer* `\(` ***numeriert*** *die Speicher durch*)
`\`*N*	*N*-ten über `\(...\)` gemerkten Text **einfügen**
`&`	Gesamter gefundener Text (*im Ersetzungsteil*)

RA - Weitere Metazeichen

Beispiel

`\<ma`	man, made, macht, aber nicht normal
`ade\>`	schade, made, ade, aber nicht maden
`\<[Mm]ade\>`	Die Worte Made oder made, aber nicht maden und nomaden
`im \(Theater|Museum\) ist es schön`	Speichert passendes Textstück in \1
`\1`	Ergibt dann Theater oder Museum
`\(\<[A-Za-z]+\>\)[ ]+\1`	Zwei gleiche Worte hintereinander

RA - Weitere Metazeichen

- Nicht alle Programme verstehen **alle Metazeichen**:
  - ~~grep, sed und vi kennen keine erweiterten Metazeichen.~~
  - ~~vi kann zusätzlich \< \> und \{n,m\}.~~
  - ~~sed und vi können zusätzlich \(\) und \N.~~
  - ~~egrep und awk können alle erw. Metaz. bis auf \< \>.~~
- Viele Programme unterscheiden die **Groß/Klein-schreibung**, einige tun dies nicht (*z.B.* less).
  - Durch eine **Option** oder Einträge in **Konfigurationsvariablen/dateien** einstellbar.


Übung 17d

- **Suchen** Sie in gedicht mit less, egrep oder vi per **erweiterten** Regulären Ausdr. nach Zeilen, die:
  - Mit "m" beginnende **Wörter** enthalten.
  - Mit "ade" endende **Wörter** enthalten.
  - Zwei **gleiche Wörter** hintereinander enthalten.
  - Zwei **gleiche Wörter** mit einem Wort dazwischen enthalten.
  - Zwei **gleiche Wörter** am Zeilenanfang und am Zeilenende enthalten.

RA - Suchen und Ersetzen

Suchen und Ersetzen

- Neben Textsuche kann bei manchen Werkzeugen (vi, sed) oder Skriptsprachen (perl) der gefundene Text auch **ersetzt** werden (*unter Verwendung des Suchergebnisses,* **substitute**, **global**).
- Dazu müssen allerdings **Suchmuster** und **Ersatztext** irgendwie getrennt werden (*meist durch "/"*):

```
s/made/MARMELADE/g
```

  - Das Zeichen "/" ist dann durch "\/" darzustellen

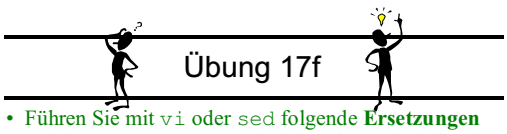
RA - Suchen und Ersetzen

Beispiel

`s/[Uu]nix/UNIX/g`	Text Unix oder unix durch UNIX ersetzen
`s/.*/(&)/`	Klammern um Zeile setzen
`s/^/ /`	Zeile einrücken
`s/ *$//`	Leerzeichen am Zeilenende entf.
`s/TAB/ /`	Tabs durch 4 Leerzeichen ersetz.
`s/^\(...\)\(..\)/\2\1/`	Spalten 1-3 mit 4-5 vertauschen


Übung 17e

- Führen Sie mit vi oder sed folgende **Ersetzungen** auf der Datei gedicht durch (*mit Hilfe des Befehls* [:%]**s**/*SUCH*/*ERSATZ*/**g**):
  - Das Wort "made" überall durch MARMELADE **ersetzen**.
    - Analog, nur **Groß/Kleinschreibung** ignorieren.
  - Die **ersten 3 Zeichen** jeder Zeile **löschen**.
  - Alle **Leerzeichen** entfernen.
  - Alle **Vokale** entfernen.
  - Jedes Wort "made" im Text **verdoppeln** (*egal ob es groß oder klein geschrieben ist*).

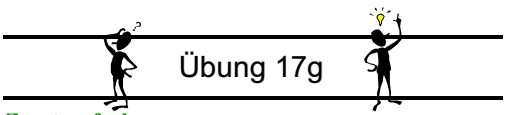


Übung 17f

- Führen Sie mit `vi` oder `sed` folgende **Ersetzungen** auf der Datei `columns` mit folgendem Inhalt durch:

```
000000000111111111122222222223...
123456789012345678901234567890...
```

  - Spalten 1-10 **löschen**.
  - Spalten 11-20 **löschen**.
  - Die letzten 5 Spalten **löschen**.
  - Spalten 1-5 mit den Spalten 6-10 **vertauschen**.
  - Spalten 1-5 mit den Spalten 11-15 **vertauschen**.
  - Spalten 1-5 mit den letzten 5 Spalten **vertauschen**.



Übung 17g

Zusatzaufgabe

- Welche **Ersatzdarstellungen** der Metazeichen

```
* ? + [] \ SPACE
```

  lassen sich mit Hilfe folgender Metazeichen angeben:

```
{} () | * []
```

- Vergleichen Sie **Reguläre Ausdrücke** und die Muster zur **Dateinamen-Expansion** miteinander.
  - Welche Gemeinsamkeiten bzw. Unterschiede gibt es?
  - Warum sind diese Unterschiede vorhanden?