Statistische Software (R)

Paul Fink, M.Sc.

Institut für Statistik
Ludwig-Maximilians-Universität München

Operationen mit Zeichenketten

Formatierung und Ausgabe

Die wichtigsten Befehle zur Ausgabe und Formatierung sind `print()`, `cat()` und `format()`.

`print()` ist dabei ein sog. *generischer Befehl*, der für jede Klasse zur Verfügung steht.

```
> X <- matrix(data = sqrt(2:7), nrow = 3, ncol = 2, byrow = TRUE)
> print(X)
         [,1]     [,2]
[1,] 1.414214 1.732051
[2,] 2.000000 2.236068
[3,] 2.449490 2.645751
```


Formatierung und Ausgabe – print()

Damit wird die Matrix in die Konsole oder Ausgabedatei (Batch-Modus) geschrieben. Das Argument `digits` erlaubt beispielsweise die Angabe der auszugebenden Stellen.

Gibt das Objekt zurück, das reingesteckt wird!

```
> print(sqrt(2))
[1] 1.414214
> print(sqrt(2), digits = 2)
[1] 1.4
> print(sqrt(2), digits = 4)
[1] 1.414
> print(sqrt(2) + 100, digits = 2)
[1] 101
> print(sqrt(2) + 100, digits = 4)
[1] 101.4
```


Formatierung und Ausgabe – format()

Die Funktion `format()` erlaubt eine umfangreichere Formatierung. Die Rückgabe ist eine Zeichnekette!

```
> print(format(sqrt(2), digits = 3, nsmall = 5))
[1] "1.41421"
> print(format(0.5, digits = 10, nsmall = 5))
[1] "0.50000"
> print(format(0.5, digits = 1, nsmall = 5), quote = FALSE)
[1] 0.50000
```

Formatierung und Ausgabe – cat()

Die Funktion `cat()` wandelt alle übergebenen Argumente in Zeichenketten um, hängt diese zusammen und gibt die gesamte Zeichenkette auf der Konsole aus.

Das Argument `sep` ist die Zeichenkette, die als Trennung zwischen den Eingabe-Zeichenketten dient.

```
> v <- 7
> cat("Quadrat von", v, ":", v^2, "!\n")
Quadrat von 7 : 49 !
> cat("Quadrat von ", v, ": ", v^2, "!\n", sep = "")
Quadrat von 7: 49!
> cat("Wurzel von", v, ": ungefaehr", format(sqrt(v), digits = 3), "\n")
Wurzel von 7 : ungefaehr 2.65
```

Achtung: `cat()` hat als Rückgabewert immer `NULL`!

Operationen mit Zeichenketten – paste()

Funktion `paste()` zum (vektorwertigen) Zusammenfügen von Zeichenketten und Zahlen mit Argument `sep` wie bei `cat()`.

```
> paste("Aufgabe", 1, "a)", sep = "_")
[1] "Aufgabe_1_a)"
> (pa <- paste("Aufgabe", 1:5, sep = "_"))
[1] "Aufgabe_1" "Aufgabe_2" "Aufgabe_3" "Aufgabe_4" "Aufgabe_5"
> pa[1]
[1] "Aufgabe_1"
```

Über das Argument `collapse`, kann man einen Vektor von Zeichenketten in eine einzige zusammenführen.

```
> (pac <- paste(pa, collapse = "*"))
[1] "Aufgabe_1*Aufgabe_2*Aufgabe_3*Aufgabe_4*Aufgabe_5"
```

`paste0()` ist eine Abkürzung für `paste( , sep = "")`

Operationen mit Zeichenketten – strsplit()

Zerlegung einer Zeichenkette in Teile mit Funktion `strsplit()`:

```
> x <- "Die Syntax#!von paste#!findet man!#wie immer!#in der Hilfe"
> strsplit(x,"!")
[[1]]
[1] "Die Syntax#"   "von paste#"    "findet man"    "#wie immer"
[5] "#in der Hilfe"
> strsplit(x,"#!")
[[1]]
[1] "Die Syntax"
[2] "von paste"
[3] "findet man!#wie immer!#in der Hilfe"
> spl <- strsplit(x,"!#")
> spl
[[1]]
[1] "Die Syntax#!von paste#!findet man" "wie immer"
[3] "in der Hilfe"
```


Operationen mit Zeichenketten – strsplit()

Achtung:

- Es liefert eine Liste zurück!

```
> is.list(spl)
[1] TRUE
```

- Die "Split"-Zeichenkette wird standardmäßig als sog. *regulärer Ausdruck* interpretiert! Verhindern mit `fixed = TRUE`

```
> y <- "Dieser?Ausdruck?geht?schief"
> strsplit(y, "?")
[[1]]
 [1] "D" "i" "e" "s" "e" "r" "?" "A" "u" "s" "d" "r" "u" "c" "k" "?" "g"
[18] "e" "h" "t" "?" "s" "c" "h" "i" "e" "f"
> strsplit(y, "?", fixed = TRUE)
[[1]]
[1] "Dieser"   "Ausdruck" "geht"     "schief"
```

Suchen und Ersetzen in Zeichenketten

Reines Suchen geht mit `grep()` und `grepl()`

```
> c(x,y)
[1] "Die Syntax#!von paste#!findet man!#wie immer!#in der Hilfe"
[2] "Dieser?Ausdruck?geht?schief"

> grep("#!", c(x,y) )
[1] 1
> # "#!" ist im 1-ten Element, also in x
> grepl("#!", c(x,y) )
[1]  TRUE FALSE
```

Suchen und Ersetzen mit `sub()` und `gsub()`

```
> sub("#!", " ", x) # Nur das erste Vorkommen
[1] "Die Syntax von paste#!findet man!#wie immer!#in der Hilfe"
> gsub("#!", " ", x) # Alle Vorkommen
[1] "Die Syntax von paste findet man!#wie immer!#in der Hilfe"
```


Reguläre Ausdrücke

Beim Suchen (und Ersetzen) von Zeichenketten werden standardmässig reguläre Ausdrücke verwendet. Verhindern wie bei `strsplit`

```
> gsub("?", " ", y)
[1] " D i e s e r ? A u s d r u c k ? g e h t ? s c h i e f "
> gsub("?", " ", y, fixed = TRUE)
[1] "Dieser Ausdruck geht schief"
> (z <- "           Viele Leerzeichen am Anfang")
[1] "           Viele Leerzeichen am Anfang"
> gsub("^\\s*", "", z) # Loeschen der Leerzeichen am Anfang
[1] "Viele Leerzeichen am Anfang"
```

Zeichen mit spezieller Bedeutung bei regulären Ausdrücken:

`"."`, `"?"`, `"^"`, `"$"`, `"*"`, `"+"`, Klammern aller Art

Übersicht

`cat()`	nur für Ausgabe in Konsole und Dateien
`formatC()`	Formatierung im C Stil
`grep()`, `grepl()`	Suchen von Zeichenfolgen
`nchar()`	Anzahl Zeichen in einer Zeichenkette
`paste()`, `paste0()`	Zusammensetzen von Zeichenketten
`strsplit()`	Zerlegen von Zeichenketten
`sub()` , `gsub()`	Ersetzen von (Teil-)Zeichenfolgen
`toupper()` , `tolower()`	Umwandlung in Groß- bzw. Kleinbuchstaben
`\t`, `\n`	Tabstopp-Einrückung, Zeilenumbruch

Aufgaben

1. Laden Sie den Vektor `gplpreamble` aus der Datei `gplpreamble.RData` direkt von der Homepage — Codeabschnitt *aufgabe-laden-daten* in `5-Rfolien.R`.
2. Entfernen Sie alle Leerzeilen (Zeilen mit weniger als einem Zeichen) aus `gplpreamble`.
3. Zählen Sie in wie vielen Zeilen die **Zeichenkette** `"you"` vorkommt. Ergeben sich Unterschiede, wenn man die Groß- und Kleinschreibung nicht berücksichtigt?
4. Ersetzen Sie alle Vorkommen des **Wortes** `"you"` mit `"YOU"`.