LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Statistische Software (R)

Paul Fink, M.Sc., Eva Endres, M.Sc.

Institut für Statistik
Ludwig-Maximilians-Universität München

Operationen mit Zeichenketten

Ausgabe

Die wichtigsten Funktionen zur Ausgabe: `print()`, `cat()`

`print()` ist dabei eine sog. *generische Funktion*, die für jedes R-Objekt 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
```

Ausgabe – `print()`

Ausgabe des Objekts auf die Konsole

Rückgabewert: Das Objekt, mit dem die Funktion aufgerufen wurde

Argument `digits`: Angabe der relevanten Stellen.

```
> 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
```

Ausgabe – cat()

Umwandlung und Verknüfpung aller übergebenen Objekte in eine Zeichenkette und Ausgabe auf die Konsole oder in eine Datei

Rückgabewert: immer NULL!

Argument sep: Zeichenkette zwischen den einzelnen Elemente
Argument file: Name einer Ausgabe-Datei (Falls leer: Konsole)

```
> 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!
```

Operationen mit Zeichenketten – `paste()`

(Vektorwertiges) Zusammenfügen von Zeichenketten

Rückgabewert: Vektor von zusammengefügten Zeichenketten

Argument `sep`: wie bei `cat()`.
Argument `collapse`: Vektor von Zeichenketten als eine Einzige

```
> pa <- paste("Aufgabe", 1:5, "a", sep = "_")
> pa
[1] "Aufgabe_1_a" "Aufgabe_2_a" "Aufgabe_3_a" "Aufgabe_4_a" "Aufgabe_5_a"
> pa[1]
[1] "Aufgabe_1_a"

> pac <- paste(pa, collapse = "?")
> pac
[1] "Aufgabe_1_a?Aufgabe_2_a?Aufgabe_3_a?Aufgabe_4_a?Aufgabe_5_a"
```

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

Operationen mit Zeichenketten – `strsplit()`

(Vektorwertige) Zerlegung einer Zeichenkette in Teile

Rückgabewert: **Liste** von Vektoren der getrennten Zeichenketten

Argument `split`: Trennzeichen

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

Operationen mit Zeichenketten – `strsplit()`

Achtung:
Interpretation der "Split"-Zeichenkette als sog. *regulärer Ausdruck*!

```
> strsplit(pac, "?")
[[1]]
 [1] "A" "u" "f" "g" "a" "b" "e" "_" "1" "_" "a" "?" "A" "u" "f" "g" "a"
[18] "b" "e" "_" "2" "_" "a" "?" "A" "u" "f" "g" "a" "b" "e" "_" "3" "_"
[35] "a" "?" "A" "u" "f" "g" "a" "b" "e" "_" "4" "_" "a" "?" "A" "u" "f"
[52] "g" "a" "b" "e" "_" "5" "_" "a"
```

Operationen mit Zeichenketten – `strsplit()`

Achtung:
Interpretation der "Split"-Zeichenkette als sog. *regulärer Ausdruck*!

```
> strsplit(pac, "?")
[[1]]
 [1] "A" "u" "f" "g" "a" "b" "e" "_" "1" "_" "a" "?" "A" "u" "f" "g" "a"
[18] "b" "e" "_" "2" "_" "a" "?" "A" "u" "f" "g" "a" "b" "e" "_" "3" "_"
[35] "a" "?" "A" "u" "f" "g" "a" "b" "e" "_" "4" "_" "a" "?" "A" "u" "f"
[52] "g" "a" "b" "e" "_" "5" "_" "a"
```

Verhindern mit `fixed = TRUE`

```
> strsplit(pac, "?", fixed = TRUE)
[[1]]
[1] "Aufgabe_1_a" "Aufgabe_2_a" "Aufgabe_3_a" "Aufgabe_4_a" "Aufgabe_5_a"
```

Zeichen mit spezieller Bedeutung bei regulären Ausdrücken:

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

Suchen und Ersetzen in Zeichenketten

Nur Muster-Suche: grep() und grepl()

```
> c(x,pac)
[1] "Die Syntax#!von strsplit#!findet man!#wie immer!#in der Hilfe"
[2] "Aufgabe_1_a?Aufgabe_2_a?Aufgabe_3_a?Aufgabe_4_a?Aufgabe_5_a"

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

Muster-Suche und Ersetzung: sub() und gsub()

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

Suchen und Ersetzen in Zeichenketten

Interpretation des Suchmusters als regulärer Ausdruck:
Verhindern wie bei `strsplit`

```
> gsub("?", " ", pac)
[1] " A u f g a b e _ 1 _ a ? A u f g a b e _ 2 _ a ? A u f g a b e _ 3 _ a ? A
> gsub("?", " ", pac, fixed = TRUE)
[1] "Aufgabe_1_a Aufgabe_2_a Aufgabe_3_a Aufgabe_4_a Aufgabe_5_a"
```

Beispiel für Verwendung eines regulären Ausdrucks:

```
> z <- "           Viele Leerzeichen am Anfang"
> z
[1] "           Viele Leerzeichen am Anfang"
> gsub("^\\s*", "", z) # Loeschen der Leerzeichen am Anfang
[1] "Viele Leerzeichen am Anfang"
```

Übersicht

`cat()`	nur für Ausgabe in Konsole und Dateien
`format()`	Formatierung von Zahlen als Zeichenkette
`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"`.