

## 2 Wprowadzenie do R cd.

### 2.1 Listy

- Kolejnym podstawowym typem danych jest lista. Najlepiej postrzegać ją jako ciąg złożony z elementów o dowolnych typach (a więc już niekoniecznie tych samych jak w przypadku wektorów atomowych). W skład listy mogą wchodzić wektory logiczne, liczbowe i napisów, a nawet funkcje, czy też same listy.
- Listy tworzymy zazwyczaj za pomocą funkcji `list()`.

```
(x <- list(TRUE, 3.5, "DSTA"))
```

```
## [[1]]  
## [1] TRUE  
##  
## [[2]]  
## [1] 3.5  
##  
## [[3]]  
## [1] "DSTA"
```

```
(x <- list(logiczna = TRUE, liczba = 3.5, napis = "DSTA"))
```

```
## $logiczna  
## [1] TRUE  
##  
## $liczba  
## [1] 3.5  
##  
## $napis  
## [1] "DSTA"
```

```
x[[1]]
```

```
## [1] TRUE
```

```
x$logiczna
```

```
## [1] TRUE
```

```
str(x)
```

```
## List of 3
```

```
## $ logiczna: logi TRUE
```

```
## $ liczba : num 3.5
```

```
## $ napis : chr "DSTA"
```

```
x[1] <- NULL
```

```
str(x)
```

```
## List of 2
```

```
## $ liczba: num 3.5
```

```
## $ napis : chr "DSTA"
```

## 2.2 Macierze

- Macierz (typ złożony `matrix` ) i, ogólniej, tablice (typ złożony `array` ) są reprezentowane w R przez wektory atomowe. Mają one jednak ustawiony atrybut specjalny `dim` . Jako wartość może mieć on przypisany jedynie wektor liczb całkowitych dodatnich o długości nie mniejszej niż dwa.
- Wykorzystując atrybut `dim` , macierz możemy utworzyć „ręcznie”.
- Macierze zazwyczaj łatwiej utworzyć korzystając z funkcji `matrix()` .

```
(x <- matrix(1:8, nrow = 2, ncol = 4))
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    3    5    7  
## [2,]    2    4    6    8
```

```
dim(x)
```

```
## [1] 2 4
```

```
nrow(x)
```

```
## [1] 2
```

```
ncol(x)
```

```
## [1] 4
```

```
(x <- matrix(1:8, nrow = 2))
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    3    5    7  
## [2,]    2    4    6    8
```

```
(x <- matrix(1:8, ncol = 4))
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    3    5    7  
## [2,]    2    4    6    8
```

```
(x <- matrix(1:8, ncol = 4, byrow = TRUE))
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    2    3    4  
## [2,]    5    6    7    8
```

- Skoro macierze są wektorami atomowymi, do ich indeksowania możemy użyć nawiasów kwadratowych `[]`. Jednak możemy korzystać z nich na „większą” liczbę sposobów.

```
(x <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]  
## [1,]    1    3    5  
## [2,]    2    4    6
```

```
x[2, 3]
```

```
## [1] 6
```

```
x[2, ]
```

```
## [1] 2 4 6
```

```
x[, 3]
```

```
## [1] 5 6
```

```
x[, 2:3]
```

```
##      [,1] [,2]
## [1,]    3    5
## [2,]    4    6
```

```
x[1:2, c(1, 3)]
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
```

- Oczywiście, w R zaimplementowanych jest wiele funkcji wykonujących operacje specyficzne dla macierzy.

```
t(matrix(1:6, ncol = 3))
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6
```

```
(A <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
(B <- matrix(7:12, ncol = 2))
```

```
##      [,1] [,2]
## [1,]    7   10
## [2,]    8   11
## [3,]    9   12
```

```
A %*% B
```

```
##      [,1] [,2]  
## [1,]   76  103  
## [2,]  100  136
```

```
B %*% A
```

```
##      [,1] [,2] [,3]  
## [1,]   27   61   95  
## [2,]   30   68  106  
## [3,]   33   75  117
```

```
(A <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]  
## [1,]    1    3    5  
## [2,]    2    4    6
```

```
(B <- matrix(7:12, ncol = 3))
```

```
##      [,1] [,2] [,3]  
## [1,]    7    9   11  
## [2,]    8   10   12
```

```
A * B
```

```
##      [,1] [,2] [,3]  
## [1,]    7   27   55  
## [2,]   16   40   72
```

- `rankMatrix()` z pakietu `Matrix` - rząd macierzy
- `det()` - wyznacznik macierzy
- `kronecker(A, B)` - iloczyn Kroneckera macierzy
- `solve(A, b)` - rozwiązuje układy równań liniowych, jako pierwszy parametr podajemy macierz współczynników, a jako drugi wektor wyrazów wolnych. Jeśli nie podamy drugiego parametru funkcja obliczy macierz odwrotną.
- `ginv()` z pakietu `MASS` - pseudoodwrotność Moore'a-Penrose'a
- `eigen()` - wartości oraz wektory własne (rozkład spektralny macierzy symetrycznej)

## 2.3 Czynniki

- Czynniki (ang. factors) można postrzegać jako wektory zawierające elementy ze zbioru o z góry określonej, najczęściej względnie niewielkiej liczbie możliwych wartości. Zatem czynniki służą do reprezentowania danych jakościowych.
- W praktyce analizy danych, zmienne typu czynnikowego zazwyczaj kodują informację o zmiennych niemierzalnych, takich jak płeć, kolor oczu czy wykształcenie. Można je oczywiście zakodować za pomocą liczb.

```
(plec <- rep(c("F", "M"), c(2, 3)))
```

```
## [1] "F" "F" "M" "M" "M"
```

```
(plec_factor <- factor(plec))
```

```
## [1] F F M M M
```

```
## Levels: F M
```

```
levels(plec_factor)
```

```
## [1] "F" "M"
```

```
nlevels(plec_factor)
```

```
## [1] 2
```

```
table(plec_factor)
```

```
## plec_factor
```

```
## F M
```

```
## 2 3
```

## 2.4 Ramki danych

- Ramki danych (ang. data frames) to obiekty przechowujące informacje w postaci macierzowej, najczęściej takie, które są np. wynikiem eksperymentów (także numerycznych). Wiersze ramki danych odpowiadają reprezentowanym obiektom, tzw. obserwacjom (ang. observations), bądź przypadkom (ang. cases), np. badanym osobom. Kolumny z kolei podają informacje na temat wartości różnych zmiennych (ang. variables) opisujących ich wybrane własności (mieralne lub nie).
- W R, ramki danych są reprezentowane przez listy zawierające wektory atomowe o tej samej długości. Każdy element tej szczególnej listy odpowiada kolumnie ramki danych.

```
ramka <- data.frame(  
  plec = c("K", "K", "M", "M", "K"),  
  wykształcenie = c("s", "w", "w", "p", "s"),  
  waga = c(60, 55, 80, 75, 62)  
)  
ramka
```

```
##   plec wykształcenie waga  
## 1    K              s   60  
## 2    K              w   55  
## 3    M              w   80  
## 4    M              p   75  
## 5    K              s   62
```



```
nrow(ramka)
```

```
## [1] 5
```

```
ncol(ramka)
```

```
## [1] 3
```

```
rownames(ramka)
```

```
## [1] "1" "2" "3" "4" "5"
```

```
colnames(ramka)
```

```
## [1] "plec"          "wyksztalcenie" "waga"
```

```
ramka[[3]] # Lub ramka$waga lub ramka[, 3]
```

```
## [1] 60 55 80 75 62
```

```
ramka$waga <- c(58, 54, 78, 72, 60)
```

```
ramka[ramka$plec == "M", ]
```

```
##   plec wyksztalcenie waga
```

```
## 3    M              w    78
```

```
## 4    M              p    72
```

```
rbind(ramka[1:2, ], ramka[1:2, ])
```

```
##   plec   wyksztalcenie   waga
## 1     K               s     58
## 2     K               w     54
## 3     K               s     58
## 4     K               w     54
```

```
cbind(ramka[1:2, ], wyksztalcenie_2 = as.integer(ramka$wyksztalcenie[1:2]))
```

```
##   plec   wyksztalcenie   waga   wyksztalcenie_2
## 1     K               s     58               2
## 2     K               w     54               3
```

## 2.5 Odczytywanie i zapisywanie danych

- `read.table()` , `load()` , `read.csv()` , `read.csv2()` - wczytanie zbioru danych, odpowiednio z pliku tekstowego, pliku w formacie programu R (z rozszerzeniem `RData` ), pliku `csv` , odpowiednio
- `write.table()` , `save()` , `write.csv()` , `write.csv2()` - zapis zbioru danych, odpowiednio do pliku tekstowego, pliku w formacie programu R (z rozszerzeniem `RData` ), plików `csv` , odpowiednio
- Przy odczytywaniu i zapisywaniu danych, wygodnie jest najpierw ustalić katalog bieżący na ten, w którym znajdują się lub mają znaleźć się pliki z danymi. Aktualny katalog bieżący sprawdzamy za pomocą funkcji `getwd()` , natomiast zmieniamy go używając funkcji `setwd()` .

```
getwd()
```

```
## [1] "/home/ls/MEGA/DYDAKTYKA/STA/DSTA_LIO/DSTA_LIO_cwiczenia_bookdown"
```

```
# setwd("/home/ls/MEGA/DYDAKTYKA/STA/DSTA_LIO")
```

```
# (odczyt_1 <- read.table("odczyt_1.txt"))
```

```
(odczyt_1 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_1.txt"))
```

```
##          V1          V2          V3
## 1 zmienna1 zmienna2 zmienna3
## 2      1.2      1.3      1.4
## 3      2.1      2.2      2.3
## 4      3.1      3.2      3.3
```

```
(odczyt_1 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_1.txt",
                        header = TRUE))
```

```
##  zmienna1 zmienna2 zmienna3
## 1      1.2      1.3      1.4
## 2      2.1      2.2      2.3
## 3      3.1      3.2      3.3
```

```
(odczyt_2 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_2.txt",
                        header = TRUE))
```

```
##  zmienna1.zmienna2.zmienna3
## 1              1,2;1,3;1,4
## 2              2,1;2,2;2,3
## 3              3,1;3,2;3,3
```

```
(odczyt_2 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_2.txt",
                        header = TRUE, sep = ";", dec = ","))
```

```
##  zmienna1 zmienna2 zmienna3
## 1      1.2      1.3      1.4
## 2      2.1      2.2      2.3
## 3      3.1      3.2      3.3
```

- Pliki z danymi do powyższych przykładów: `odczyt_1.txt`, `odczyt_2.txt`

- Można też zaimportować dane klikając na `Import Dataset` w RStudio i w otworzonym okienku ustawić potrzebne parametry.
- Podgląd danych w edytorze kodu źródłowego otrzymujemy za pomocą funkcji `view()`.
- Zapisywanie danych:

```
dane_1 <- data.frame(1:10, 5:14)
write.table(dane_1, "dane_1.txt")
save(dane_1, file = "dane_1.RData")
dane_1 <- read.table("dane_1.txt")
load("dane_1.RData")
```

## 2.6 Zadania

**Zadanie 1.** Skonstruuj listę o nazwie `moja_lista`, której pierwszym elementem będzie dwuelementowy wektor napisów zawierający Twoje imię i nazwisko, drugim elementem będzie liczba  $\pi$ , trzecim funkcja służąca do obliczania pierwiastka kwadratowego, a ostatni element listy to wektor złożony z liczb 0,02; 0,04; ...; 1. Następnie usuń elementy numer jeden i trzy z tej listy. Na zakończenie, wyznacz listę zawierającą wartości funkcji gamma Eulera dla elementów listy `moja_lista`.

```
## List of 4
## $ : chr [1:2] "Łukasz" "Smaga"
## $ : num 3.14
## $ :function (x)
## $ : num [1:50] 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2 ...

## List of 2
## $ : num 3.14
## $ : num [1:50] 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2 ...
```

```
## [[1]]
## [1] 2.288038
##
## [[2]]
## [1] 49.442210 24.460955 16.145727 11.996566 9.513508 7.863252 6.688686
## [8] 5.811269 5.131821 4.590844 4.150482 3.785504 3.478450 3.216852
## [15] 2.991569 2.795751 2.624163 2.472735 2.338256 2.218160 2.110371
## [22] 2.013193 1.925227 1.845306 1.772454 1.705844 1.644773 1.588641
## [29] 1.536930 1.489192 1.445038 1.404128 1.366164 1.330884 1.298055
## [36] 1.267473 1.238954 1.212335 1.187471 1.164230 1.142494 1.122158
## [43] 1.103124 1.085308 1.068629 1.053016 1.038403 1.024732 1.011947
## [50] 1.000000
```

**Zadanie 2.** Wyznacz rząd, wyznacznik, odwrotność, wartości własne, wektory własne oraz sumy i średnie arytmetyczne dla kolejnych wierszy i kolumn dla następującej macierzy:

$$\begin{bmatrix} 1 & 5 & 3 \\ 2 & 0 & 5 \\ 1 & 2 & 1 \end{bmatrix}$$

Ponadto, pomnóż tę macierz przez jej odwrotność.

```
## [1] 3
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 6.661338e-16

## [1] 17

##           [,1]      [,2]      [,3]
## [1,] -0.5882353  0.05882353  1.47058824
## [2,]  0.1764706 -0.11764706  0.05882353
## [3,]  0.2352941  0.17647059 -0.58823529
```

```
## eigen() decomposition
## $values
## [1]  6.0790256 -3.2070365 -0.8719891
##
## $vectors
##           [,1]      [,2]      [,3]
## [1,] -0.7537024  0.7058088 -0.9275678
## [2,] -0.5472752 -0.6900345  0.1392322
## [3,] -0.3638991  0.1602696  0.3467453
```

```
## [1] 9 7 4
```

```
## [1] 3.000000 2.333333 1.333333
```

```
## [1] 4 7 9
```

```
## [1] 1.333333 2.333333 3.000000
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.000000e+00 -1.110223e-16 2.220446e-16
## [2,] 0.000000e+00 1.000000e+00 0.000000e+00
## [3,] -2.775558e-17 -2.775558e-17 1.000000e+00
```

**Zadanie 3.** Utwórz wektor kwadratów 100 pierwszych liczb naturalnych. Następnie zlicz, które cyfry oraz jak często występują na pozycji jedności w kolejnych elementach tego wektora.

```
## [1] 1 4 9 16 25 36 ...
```

```
##
```

```
## 0 1 4 5 6 9
```

```
## 10 20 20 10 20 20
```

**Zadanie 4.** Za pomocą funkcji `outer()` wyznacz tabliczkę mnożenia dla liczb mniejszych od 6.

```
##      [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] "1 * 1 = 1" "1 * 2 = 2" "1 * 3 = 3" "1 * 4 = 4" "1 * 5 = 5"
## [2,] "2 * 1 = 2" "2 * 2 = 4" "2 * 3 = 6" "2 * 4 = 8" "2 * 5 = 10"
## [3,] "3 * 1 = 3" "3 * 2 = 6" "3 * 3 = 9" "3 * 4 = 12" "3 * 5 = 15"
## [4,] "4 * 1 = 4" "4 * 2 = 8" "4 * 3 = 12" "4 * 4 = 16" "4 * 5 = 20"
## [5,] "5 * 1 = 5" "5 * 2 = 10" "5 * 3 = 15" "5 * 4 = 20" "5 * 5 = 25"
```

**Zadanie 5.** Odczytaj zbiór danych `dane1.csv` a następnie:

1. Z odczytanej ramki danych wyświetl tylko parzyste wiersze.
2. Korzystając z operatorów logicznych wyświetl tylko wiersze odpowiadające pacjentkom starszym niż 50 lat z przerzutami do węzłów chłonnych ( `Wezly.chlonne = 1` ).

```
##   Wiek Rozmiar.guza Wezly.chlonne Nowotwor Receptory.estrogenowe
## 1   29           1             0         2                (-)
## 2   29           1             0         2                (++)
## 3   30           1             1         2                (-)
## 4   32           1             0         3                (++)
## 5   32           2             0        NA                (-)
## 6   33           1             1         3                (-)
##   Receptory.progesteronowe Niepowodzenia Okres.bez.wznowy VEGF
## 1                        (++)           brak           22  914
## 2                        (++)           brak           53 1118
## 3                        (+)           brak           38  630
## 4                        (++)           brak           26 1793
## 5                        (++)           brak           19  963
## 6                        (++)         wznowa           36 2776

## ...
```

##	Wiek	Rozmiar.guza	Wezly.chlonne	Nowotwor	Receptory.estrogenowe
## 2	29	1	0	2	(++)
## 4	32	1	0	3	(++)
## 6	33	1	1	3	(-)
## 8	35	2	1	2	(+)
## 10	36	1	1	2	(-)
## 12	37	1	0	3	(-)

##	Receptory.progesteronowe	Niepowodzenia	Okres.bez.wznowy	VEGF
## 2	(++)	brak	53	1118
## 4	(++)	brak	26	1793
## 6	(++)	wznowa	36	2776
## 8	(++)	brak	38	3827
## 10	(++)	brak	37	834
## 12	(+)	wznowa	40	3331

## ...

##	Wiek	Rozmiar.guza	Wezly.chlonne	Nowotwor	Receptory.estrogenowe
## 78	51	1	1	2	(++)
## 79	51	1	1	2	(+)
## 81	51	2	1	2	(+++)
## 84	51	2	1	NA	(-)
## 88	52	2	1	2	(+)
## 95	55	1	1	2	(++)

##	Receptory.progesteronowe	Niepowodzenia	Okres.bez.wznowy	VEGF
## 78	(++)	brak	33	629
## 79	(+)	brak	36	2879
## 81	(++)	brak	52	1098
## 84	(-)	brak	30	8064
## 88	(+)	wznowa	48	1927
## 95	(++)	brak	29	373

## ...

**Zadanie 6.** Poniższe dane są średnimi miesięcznymi temperaturami (w °F) w Nowym Yorku.



Styczeń – 32	Kwiecień – 52	Lipiec – 77	Październik – 58
Luty – 33	Maj – 62	Sierpień – 75	Listopad – 47
Marzec – 41	Czerwiec – 72	Wrzesień – 68	Grudzień – 35

1. Wprowadź te dane do ramki danych o jednej zmiennej `NY_F` .
2. Utwórz nową zmienną `NY_C` podającą temperaturę w stopniach Celsjusza (zaokrągloną do dwóch miejsc po przecinku). **Wskazówka:**  $(x^{\circ}\text{F}) = (x - 32) \cdot 5/9(^{\circ}\text{C})$
3. Zamień nazwy kolumn na `NY_Fahrenheit` i `NY_Celsiusz` .
4. Usuń kolumnę z temperaturą w Fahrenheitach.
5. Zapisz otrzymane dane w pliku `NY_temp.RData` .

```
##           NY_F
## Styczeń      32
## Luty         33
## Marzec       41
## Kwiecień     52
## Maj          62
## Czerwiec     72
## Lipiec       77
## Sierpień     75
## Wrzesień     68
## Październik  58
## Listopad     47
## Grudzień     35
```

```
##           NY_F  NY_C
## Styczeń      32  0.00
## Luty         33  0.56
## Marzec       41  5.00
## Kwiecień     52 11.11
## Maj          62 16.67
## Czerwiec     72 22.22
## Lipiec       77 25.00
## Sierpień     75 23.89
## Wrzesień     68 20.00
## Październik  58 14.44
## Listopad     47  8.33
## Grudzień     35  1.67
```

##	NY_Fahrenheit	NY_Celsiusz
## Styczeń	32	0.00
## Luty	33	0.56
## Marzec	41	5.00
## Kwiecień	52	11.11
## Maj	62	16.67
## Czerwiec	72	22.22
## Lipiec	77	25.00
## Sierpień	75	23.89
## Wrzesień	68	20.00
## Październik	58	14.44
## Listopad	47	8.33
## Grudzień	35	1.67

##	NY_Celsiusz
## Styczeń	0.00
## Luty	0.56
## Marzec	5.00
## Kwiecień	11.11
## Maj	16.67
## Czerwiec	22.22
## Lipiec	25.00
## Sierpień	23.89
## Wrzesień	20.00
## Październik	14.44
## Listopad	8.33
## Grudzień	1.67