Datové struktury v R

Základní datové struktury v R jsou vektor (*vector*), faktor (*factor*), matice (*matrix*), pole (*array*), datová tabulka (*data.frame*) a seznam (*list*). Všechny tyto struktury si představíme dále. Pro začátek prozradíme, že všechny jsou tvořeny výčtem jednoduchých prvků určitých datových typů. Tento výčet může být jednorozměrný (třeba řada čísel) nebo vícerozměrný (třeba „tabulka“ dat pacienta). Na datových strukturách je možné provádět matematické operace.

Datové typy, které jsme si představili v předchozí kapitole, jsou vlastně ve své podstatě nejjednodušší vektory o jednom prvku. Dokážeme si to později, nejprve si přehledně zobrazíme vlastnosti jednotlivých struktur.

struktura	počet rozměrů	počet datových typů	další vlastnosti	ukázka
vector	1 *(řada prvků)*	1 *(prvky stejného typu)*	vektor může nést prvky typu *numeric*, *character* nebo *logical*	$3, 18, 9, 126$
factor	1 *(řada prvků)*	1 *(prvky stejného typu)*	faktory se používají k výčtu prvků, které mohou nabývat jednoznačně definovaného rozsahu hodnot (např. pohlaví)	"muž", "žena", "žena", "muž"
matrix	2 *(dvourozměrné pole prvků)*	1 *(prvky stejného typu)*	matice může nést prvky typu *numeric*, *character* nebo *logical*	$\begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$
array	n *(libovolný počet rozměrů)*	1 *(prvky stejného typu)*	objekt s vlastnostmi matice, ale libovolným počtem dimenzí	viz třeba vektor nebo matice
data.frame	2 *(dvourozměrné pole prvků)*	n *(prvky různých datových typů)*	jeden sloupec datové tabulky obsahuje vždy prvky stejného typu, sloupce se ale mezi sebou liší	$\begin{pmatrix} \text{Pavel} & \text{mu} & 35 & \text{Plze} \\ \text{Ren ta} & & \text{ena} & 82 \ \text{Bratislava} \end{pmatrix}$
list	n *(libovolný počet rozměrů)*	n *(prvky různých datových typů)*	rozměry jednotlivých polí nebo rozměrů se mohou lišit	hm... pro představu třeba řada účtenek, každá obsahuje různý počet položek nákupu, ceny, názvy a podobně

Vektor

Vektor (*vector*) je jednorozměrný výčet prvků stejného datového typu. Například řada čísel nebo řada textových řetězců.

Vektory zachovávají pořadí svých prvků. Vytvoříme je pomocí funkce `c()`. Pokud jste již v nějakém jazyce programovali, tak dávejte pozor, R indexuje prvky od jedničky, nikoli od nuly jako jiné jazyky.

Příklady

```
# vytvoříme prázdný vektor
prazdny <- c()

# jeho délka (počet prvků) je samozřejmě 0
length(prazdny)
#> [1] 0

# vytvoříme lepší vektor
jedenprvek <- c(1)

# vzpomínáte, že jednotlivé datové typy jsou vlastně vektory o jednom prvku?
jedenprvek == 1
#> [1] TRUE

# jiný vektor s řetězci
text <- c("ahoj", "světe", "!")
text
#> [1] "ahoj"  "světe" "!"

# vnořený vektor
c(c(1, 2, 3), c(4, 5)) == c(1, 2, 3, 4, 5)
#> [1] TRUE TRUE TRUE TRUE TRUE
# vidíme, že operátor porovnání (==) porovnává každý prvek zvlášť
# a vektor složený z více vektorů je zase jenom vektor
```

🔎 *Podrobnější informace naleznete na stránce Práce s vektory v R.*

Faktor

Faktory (*factor*) jsou vektory (tj. jednorozměrné řady) textových hodnot, kde každá hodnota patří do své kategorie. Faktory se používají k výčtu prvků, které mohou nabývat jednoznačně definovaného rozsahu hodnot (např. pohlaví).

Faktory se tvoří příkazem `factor()`.

Příklady

```
# vytvoříme faktor šesti lidí ve skupině
skupina <- factor(c("muž", "žena", "muž", "muž", "žena", "žena"))

# zobrazíme skupinu
skupina
#> [1] muž  žena muž  muž  žena žena
#> Levels: muž žena

# pořadí kategorií je abecední, pokud chceme změnu, musíme si o ni říct
levels(skupina) <- c("žena", "muž")

# teď je žena na prvním místě
skupina
#> [1] žena muž  žena žena muž  muž
#> Levels: žena muž

# je to stejná skupina!
```

Nad faktory můžeme jednoduše sestavit kontingenční tabulku. Ta se ve statistice užívá k přehlednému zobrazení vztahů mezi dvěma statistickými znaky. Kontingenční tabulku vytvoříme příkazem `table()`. Ten nám vypíše název faktoru a následně tabulku.

```
table(skupina)
skupina
#> žena  muž
#>    3    3
```

Matice

Matice (*matrix*) je dvojrozměrným polem, které obsahuje prvky stejného datového typu. Všechny sloupce matice mají shodnou délku a také všechny řádky matice mají shodnou délku.

Příklady

Matice mohou vypadat třeba takto:

$$A = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}, B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Jak si je vytvoříme? Použijeme příkaz `matrix()`. Jako první parametr je čtyřprvkový vektor tak, jak jdou prvky ve sloupcích. Parametr `nrow` udává počet řádků matice, parametr `ncol` udává počet sloupců matice.

```
# vytvoříme matici A
A <- matrix(c(1, 2, 3, 4), nrow = 2, ncol = 2)

# jak to vypadá na výstupu?
A
#>      [,1] [,2]
#> [1,]    1    3
#> [2,]    2    4

# je to stejné, jako v příkladu nahoře!

# teď zkusíme matici B
B <- matrix(c(1, 3, 2, 4), nrow = 2, ncol = 2)

# je to tak?
B
#>      [,1] [,2]
#> [1,]    1    2
#> [2,]    3    4
```

Matici můžeme definovat i vektorem, kde prvky vyjmenujeme po řádcích. V tom případě musíme uvést parametr `byrow` s logickou hodnotou TRUE.

Mimochodem: vidíte, jak se nám tu hezky kombinují datové typy a datové struktury? Tvoříme *matrix*, jako první parametr poskytujeme *vector*, další dva parametry jsou *numeric* a poslední parametr je *logical*.

```
# vyjmenujeme prvky matice B po řádcích a uvedeme parametr "byrow"
B <- matrix(c(1, 2, 3, 4), nrow = 2, ncol = 2, byrow = TRUE)

# zkontrolujeme, že je to pořád stejná matice B, jak jsme si ji načrtli nahoře
B
#>      [,1] [,2]
#> [1,]    1    2
#> [2,]    3    4
```

🔎 *Podrobnější informace naleznete na stránce Práce s maticemi v R.*

Pole

Pole (*array*) je objekt s vlastnostmi matice, ale libovolným počtem rozměrů. Matice je jen dvourozměrná, zatímco pole může mít libovolný počet rozměrů. Nicméně se pole se příliš nepoužívají, protože vstupem pro analytické funkce jazyka R je obvykle matice či datová tabulka.

Pole vytvoříme příkazem `array()` s výčtem prvků po sloupcích v prvním parametru (coby vektorem) a rozměry v druhém parametru (jako druhý vektor: řádky, sloupce, hloubka… a další rozměry klidně až do paralelních vesmírů).

Příklady

```
# vytvoříme pole
# c(1:24) je zkratka pro c(1, 2, 3, 4, 5, …, 23, 24)
# parametr dim() udává rozměry 4 řádky, 3 sloupce a 2 "hloubky"
mojepole <- array(c(1:24), dim = c(4, 3, 2))

# přečteme mojepole
# další rozměry kromě řádků a sloupců jsou uvedeny jako "nadpis" s úvodními čárkami,
# kde chybí první dvě hodnoty (což jsou ty řádky a sloupce ;-))
mojepole
#> , , 1
#>
#>      [,1] [,2] [,3]
#> [1,]    1    5    9
#> [2,]    2    6   10
#> [3,]    3    7   11
#> [4,]    4    8   12
#>
#> , , 2
#>
#>      [,1] [,2] [,3]
#> [1,]   13   17   21
#> [2,]   14   18   22
#> [3,]   15   19   23
#> [4,]   16   20   24

# když se chceme v poli pohybovat, použijeme hranaté závorky
mojepole[, , 1]        # první vrstva
mojepole[, , 2]        # druhá vrstva
mojepole[3, , 1]      # třetí řádek první vrstvy
mojepole[, 2, ]        # druhé sloupce všech vrstev

# a třeba si zobrazíme hodnotu na druhém řádku třetího sloupce v první vrstvě
mojepole[2, 3, 1]
#> [1] 10

# je to tak!!!
```

Tabulka dat

Datová tabulka (*data.frame*) je dvojrozměrným polem, které obsahuje v každém sloupci prvky stejného datového typu, ale jednotlivé sloupce se mohou datovým typem lišit. Všechny sloupce datové tabulky mají shodnou délku a samozřejmě i všechny řádky datové tabulky mají shodnou délku.

Datové tabulky už se nejlépe podobají tomu, s čím se budeme setkávat. Každý sloupec může nést prvky jiného datového typu. Podívejte níže: jméno a pohlaví jsou textové řetězce (vlastně ty sloupce tvoří faktor, vidíte to?), zatímco sloupec s věkem nese číselné hodnoty.

Jméno	Věk	Pohlaví
Irena	27	žena
Pavel	38	muž
Anastázie	67	žena

Příklady

Datovou tabulku můžeme vytvořit příkazem `data.frame()`, kde každý sloupec tvoří samostatný vektor (nebo faktor). Dávejte pozor, každý textový vektor je automaticky převeden na faktor, tedy hodnoty jsou převedeny do kategorií.

Vyzkoušejte:

```
# vytvoříme výše uvedenou tabulku
jmeno <- c("Irena", "Pavel", "Anastázie")
vek <- c(27, 38, 67)
pohlavi <- c("žena", "muž", "žena")

skupina <- data.frame(jmeno, vek, pohlavi)

# vypíšeme tabulku
skupina
#>       jmeno vek pohlavi
#> 1     Irena  27    žena
#> 2     Pavel  38     muž
#> 3 Anastázie  67    žena
```

```
# to sedí!
```

Tabulku dat můžeme velice jednoduše přečíst z textové tabulky. Vlastně tím importujeme data. Cítíte to napětí? Dostáváme se do datových otáček!

```
# přečtu data příkazem read.table()
# první řádek je záhlaví tabulky, proto uvedu header=TRUE
skupina <- read.table(header=TRUE, text='
    jmeno vek pohlavi
    Irena  27    žena
    Pavel  38     muž
Anastázie  67    žena
 ')

# vypíšu skupinu
skupina
#>       jmeno vek pohlavi
#> 1     Irena  27    žena
#> 2     Pavel  38     muž
#> 3 Anastázie  67    žena

# je to ono!!!
```

V R je řada vestavěných datových tabulek, např. `mtcars`, se kterou lze pracovat.

```
# mtcars je strašně dlouhá vestavěná datová tabulka s parametry aut
# výpis si zkuste sami ve svém počítači
mtcars

# když je to tak dlouhé, vyplatí se příkaz str(), který nám poskytne náhled na celou tabulku
str(mtcars)

#> 'data.frame':    32 obs. of  11 variables:
#>  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
#>  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
#>  $ disp: num  160 160 108 258 360 ...
#>  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
#>  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
#>  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
#>  $ qsec: num  16.5 17 18.6 19.4 17 ...
#>  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
#>  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
#>  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
#>  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...

# rozměry této tabulky
dim(mtcars)
#> [1] 32 11

# je to vlastně vektor c(32, 11), který můžeme použít při výpočtech jinde
is.vector(dim(mtcars))
#> [1] TRUE
```

🔎 *Podrobnější informace naleznete na stránce Práce s datovými tabulkami v R.*

Seznam

Seznam (*list*) je výčtem prvků, které mohou být různorodého datového typu, včetně listu samotného.

Zní to složitě, ale představte si třeba několik seznamů úkolů do práce nebo do školy seřazených za sebou. Každý z nich může být jinak dlouhý, jeden má třeba jen dva body, kdežto další může mít dvě stránky. Nebo dalším příkladem může být štos účtenek, z nichž každá obsahuje tabulku položek s názvy a cenami.

Vidíme, že délky jednotlivých prvků seznamu se mohou lišit a seznam může tedy obsahovat i další vnořené seznamy.

Příklady

Seznam se tvoří příkazem `list()`.

```
# seznam bude složený ze čtyř položek
    # položka "a" bude vektor čísel od 1 do 10 - c(1:10) je zkratkou c(1, 2, 3, 4, …, 9, 10)
    # položka "b" bude vestavěná [[#Tabulky dat|datová tabulka]] "mtcars"
    # položka "c" bude matice čísel od 1 do 8 se dvěma řádky
    # položka "z" bude textový řetězec, pozdrav "ahoj"
mujseznam <- list("a" = c(1:10), "b" = mtcars, "c" = matrix(1:8, 2), "z" = "ahoj")

# teď si seznam prozkoumáme příkazem str(),
# který nám poskytne zběžný náhled na obsah seznamu
str(mujseznam)

#> List of 4
#>  $ a: int [1:10] 1 2 3 4 5 6 7 8 9 10
#>  $ b:'data.frame':   32 obs. of  11 variables:
#>   ..$ mpg : num [1:32] 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
#>   ..$ cyl : num [1:32] 6 6 4 6 8 6 8 4 4 6 ...
#>   ..$ disp: num [1:32] 160 160 108 258 360 ...
#>   ..$ hp  : num [1:32] 110 110 93 110 175 105 245 62 95 123 ...
#>   ..$ drat: num [1:32] 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
#>   ..$ wt  : num [1:32] 2.62 2.88 2.32 3.21 3.44 ...
#>   ..$ qsec: num [1:32] 16.5 17 18.6 19.4 17 ...
#>   ..$ vs  : num [1:32] 0 0 1 1 0 1 0 1 1 1 ...
```

```
#>   ..$ am  : num [1:32] 1 1 1 0 0 0 0 0 0 0 ...
#>   ..$ gear: num [1:32] 4 4 4 3 3 3 3 4 4 4 ...
#>   ..$ carb: num [1:32] 4 4 1 1 2 1 4 2 2 4 ...
#>  $ c: int [1:2, 1:4] 1 2 3 4 5 6 7 8
#>  $ z: chr "ahoj"

# zkontroluji, že je to fakt seznam
class(mujseznam)
#> [1] "list"

# ještě jednou to zkontroluji
is.list(mujseznam)
#> [1] TRUE

# a teď se podívám na položku "z", co je zač;
# můžu na ni odkázat buď hranatými závorkami
mujseznam["z"]
#> $z
#> [1] "ahoj"

# nebo znakem dolaru mezi jménem seznamu a jménem jeho položky
mujseznam$z
#> [1] "ahoj"
```

🔎 *Podrobnější informace naleznete na stránce Práce se seznamy v R.*

Souhrn

V následující tabulce si shrneme základní práci s datovými strukturami v prostředí R.

struktura	definice	class(x)	mode(x)	potvrzení typu
vector	`x <- c(…)`	*podle typu položek (character, numeric, logical)*	*podle typu položek (character, numeric, logical)*	`is.vector(x)`
factor	`x <- factor(…, levels = c(…))`	factor	*podle typu položek (character, numeric, logical)*	`is.factor(x)`
matrix	`x <- matrix(…, nrow = …, ncol = …, byrow = …)`	matrix	*podle typu položek (character, numeric, logical)*	`is.matrix(x)`
array	`x <- array(…, dim = c(…))`	array	*podle typu položek (character, numeric, logical)*	`is.array(x)`
data.frame	`x <- data.frame(…)`	data.frame	list	`is.data.frame(x)`
list	`x <- list(…)`	list	list	`is.list(x)`