

Uvod u R

Prirodoslovno-matematički Fakultet, Sveučilište u Zagrebu

2. listopada 2026.

Uvod u R

Neke generalne funkcije koje ćemo koristiti

- `ls()`
- `rm()`
- `getwd()`
- `setwd("../")`
- `help()`

Tipovi podataka

R ima klasične tipove podataka:

- `numeric` (10.55, 34, -2.3)
- `integer` (10L, -12L) - slovo "L" označava podatak cijelim brojem
- `complex`
- `character` ("jutro", "dan", "10.55")
- `logical` (TRUE, FALSE)

Pozivom funkcije `class()` možemo otkriti koji tip podataka je spremljen u varijabli.

Tipovi podataka

R ima klasične tipove podataka:

- `numeric` (10.55, 34, -2.3)
- `integer` (10L, -12L) - slovo "L" označava podatak cijelim brojem
- `complex`
- `character` ("jutro", "dan", "10.55")
- `logical` (TRUE, FALSE)

Pozivom funkcije `class()` možemo otkriti koji tip podataka je spremljen u varijabli.

Zadatak

Kako biste dobili vektor s 2., 3. i 5. elementom od `a`?

Vektori

Vektor kreiramo koristeći funkciju `c()`.

Zadatak

Što ispisuju zadnje dvije naredbe u donjem kodu?

```
x <- c(4, 7, 7, 9, 12)
smjena <- c("jutro", "noc", "jutro", "popodne", "jutro")
length(x)
class(x)
```

Vektori

Vektor kreiramo koristeći funkciju `c()`.

Zadatak

Što ispisuju zadnje dvije naredbe u donjem kodu?

```
x <- c(4, 7, 7, 9, 12)
smjena <- c("jutro", "noc", "jutro", "popodne", "jutro")
length(x)
class(x)
```

Dozvoljene su sljedeće operacije

```
y <- c(2, -3, 1, 7, -1)
x-y
x*y
x+2
x/4
x^2
```

Zadatak

Provjerite što će biti spremljeno u a, b i c.

```
a <- -4:5
```

```
b <- seq(from = 0, to = 1, by = 0.1)
```

```
c <- seq(from = 0, to = 1, length.out = 11)
```

Zadatak

Provjerite što će biti spremljeno u a, b i c.

```
a <- -4:5  
b <- seq(from = 0, to = 1, by = 0.1)  
c <- seq(from = 0, to = 1, length.out = 11)
```

Ako želimo pristupiti 3. elementu ili 2. – 5. ili svima osim 3. i 6. pišemo sljedeće:

```
a[3]  
a[2:5]  
a[c(-3, -6)]
```

Vektori

Zadatak

Provjerite što će biti spremljeno u a, b i c.

```
a <- -4:5  
b <- seq(from = 0, to = 1, by = 0.1)  
c <- seq(from = 0, to = 1, length.out = 11)
```

Ako želimo pristupiti 3. elementu ili 2. – 5. ili svima osim 3. i 6. pišemo sljedeće:

```
a[3]  
a[2:5]  
a[c(-3, -6)]
```

Korisne funkcije za vektore: `length()`, `sum()`, `prod()`, `sort()`.

Zadaci

Rješavanje zadataka 1., 2., 4., 5., 6., 9., 10., 12. i 13. za vektore.

Podatkovni okvir (*data frame*)

Ako imamo više podataka za koje ima smisla da su objedinjeni (npr. pratimo letove, kašnjenje i broj putnika na pojedinom letu), spremamo ih kao `data.frame`.

Pozivom sljedeće funkcije dobit ćemo podatkovni okvir sa stupcima pod nazivom `id`, `smjena` i `broj_poziva`, a podaci će biti brojevi od 1 do 5, podaci iz vektora `smjena` te podaci iz vektora `x`, redom.

```
uzorak <- data.frame(  
  id = 1:5,  
  smjena = smjena,  
  broj_poziva = x  
)  
uzorak
```

Matrice

Ako su svi podaci istog tipa (npr. svi podaci su tipa `numeric`) možemo koristiti matrice. Koristimo funkciju `matrix()`.

Na primjer, ako želimo stvoriti matricu $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$, možemo to učiniti izvršavanjem sljedećih linija koda

```
podaci <- c(1,2,3,4,5,6)
matrica <- matrix(podaci, nrow=2, ncol=3, byrow=True)
```

Matrice

Ako su svi podaci istog tipa (npr. svi podaci su tipa `numeric`) možemo koristiti matrice. Koristimo funkciju `matrix()`.

Na primjer, ako želimo stvoriti matricu $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$, možemo to učiniti izvršavanjem sljedećih linija koda

```
podaci <- c(1,2,3,4,5,6)
matrica <- matrix(podaci, nrow=2, ncol=3, byrow=True)
```

Učitavanje i prvi pregled

Za učitavanje obično koristimo sljedeće funkcije

```
read.table(file, header = FALSE, sep = "", dec = ".")  
read.csv(file, header = TRUE, sep = ",", dec = ".")
```

Učitavanje i prvi pregled

Za učitavanje obično koristimo sljedeće funkcije

```
read.table(file, header = FALSE, sep = "", dec = ".")  
read.csv(file, header = TRUE, sep = ",", dec = ".")
```

Zadatak

Analizirajte podatke `pozivi.csv`. Kako biste ih unijeli? Spremite ih u varijablu `pozivi`.

Učitavanje i prvi pregled

Za učitavanje obično koristimo sljedeće funkcije

```
read.table(file, header = FALSE, sep = "", dec = ".")  
read.csv(file, header = TRUE, sep = ",", dec = ".")
```

Zadatak

Analizirajte podatke `pozivi.csv`. Kako biste ih unijeli? Spremite ih u varijablu `pozivi`.

Rješenje.

```
pozivi <- read.csv("podaci/pozivi.csv", sep=";", dec=",")
```



Učitavanje i prvi pregled

Za učitavanje obično koristimo sljedeće funkcije

```
read.table(file, header = FALSE, sep = "", dec = ".")  
read.csv(file, header = TRUE, sep = ",", dec = ".")
```

Zadatak

Analizirajte podatke `pozivi.csv`. Kako biste ih unijeli? Spremite ih u varijablu `pozivi`.

Rješenje.

```
pozivi <- read.csv("podaci/pozivi.csv", sep=";", dec=",")
```



Za početni dojam o podacima možemo koristiti sljedeće funkcije

```
head(pozivi)  
str(pozivi)  
summary(pozivi)  
dim(pozivi)
```

Dohvat i izmjena podataka u podatkovnom okviru

Ako želimo promijeniti naziv stupca, radimo sljedeće:

```
names(pozivi) <- c("s1", "s2", "s3", "s4", "s5")
```

Dohvat i izmjena podataka u podatkovnom okviru

Ako želimo promijeniti naziv stupca, radimo sljedeće:

```
names(pozivi) <- c("s1", "s2", "s3", "s4", "s5")
```

Stupcu u podatkovnom okviru pristupamo operatorom `$` (ako stupac ima naziv, npr. `pozivi$dan`) ili uglatim zagradama (`pozivi[redci, stupci]`).

Dohvat i izmjena podataka u podatkovnom okviru

Ako želimo promijeniti naziv stupca, radimo sljedeće:

```
names(pozivi) <- c("s1", "s2", "s3", "s4", "s5")
```

Stupcu u podatkovnom okviru pristupamo operatorom `$` (ako stupac ima naziv, npr. `pozivi$dan`) ili uglatim zagradama (`pozivi[redci, stupci]`).

Zadatak

Izvršite sljedeće linije koda u R-u i komentirajte što dobijete.

```
pozivi$broj_poziva  
pozivi[, c("smjena", "broj_poziva")]  
pozivi[1:4, ]
```

Dohvat i izmjena podataka u podatkovnom okviru

Ako želimo promijeniti naziv stupca, radimo sljedeće:

```
names(pozivi) <- c("s1", "s2", "s3", "s4", "s5")
```

Stupcu u podatkovnom okviru pristupamo operatorom `$` (ako stupac ima naziv, npr. `pozivi$dan`) ili uglatim zagradama (`pozivi[redci, stupci]`).

Zadatak

Izvršite sljedeće linije koda u R-u i komentirajte što dobijete.

```
pozivi$broj_poziva  
pozivi[, c("smjena", "broj_poziva")]  
pozivi[1:4, ]
```

- **Dodavanje novog stupca** `pozivi$ime <- novi_stupac`
- **Brisanje stupca** `pozivi$dan <- NULL`

Logički operatori i provjera uvjeta

Usporedbe vraćaju logičke vrijednosti TRUE i FALSE. Logički operatori su

- == (provjerava jesu li dva objekta jednaka)
- != (provjerava jesu li dva objekta različita)
- & (i)
- | (ili)
- ! (ne)

Logički operatori i provjera uvjeta

Usporedbe vraćaju logičke vrijednosti TRUE i FALSE. Logički operatori su

- == (provjerava jesu li dva objekta jednaka)
- != (provjerava jesu li dva objekta različita)
- & (i)
- | (ili)
- ! (ne)

U R-u možemo provjeravati je li vektor jednak nekom elementu. Na primjer ako napišemo

```
x == 7
```

Logički operatori i provjera uvjeta

Usporedbe vraćaju logičke vrijednosti TRUE i FALSE. Logički operatori su

- `==` (provjerava jesu li dva objekta jednaka)
- `!=` (provjerava jesu li dva objekta različita)
- `&` (i)
- `|` (ili)
- `!` (ne)

U R-u možemo provjeravati je li vektor jednak nekom elementu. Na primjer ako napišemo

```
x == 7
```

kao rezultat ćemo dobiti vektor s logičnim vrijednostima

```
FALSE TRUE TRUE FALSE FALSE.
```

Provjera uvjeta

Ako nas zanima pripada li vrijednost zadanom skupu, koristimo `%in%`. Koji rezultat će dati prva, a koji druga linija koda? Provjerite u R-u.

```
x %in% c(7,12)  
x[x %in% c(7,12)]
```

Provjera uvjeta

Ako nas zanima pripada li vrijednost zadanom skupu, koristimo `%in%`. Koji rezultat će dati prva, a koji druga linija koda? Provjerite u R-u.

```
x %in% c(7,12)
x[x %in% c(7,12)]
```

Zadatak

Kreirajte novi data frame s nazivom `pozivi_jutro` koji će sadržavati informacije samo o pozivima u jutarnjoj smjeni. Iz nje izbrišite stupac `smjena`.

Provjera uvjeta

Ako nas zanima pripada li vrijednost zadanom skupu, koristimo `%in%`. Koji rezultat će dati prva, a koji druga linija koda? Provjerite u R-u.

```
x %in% c(7,12)
x[x %in% c(7,12)]
```

Zadatak

Kreirajte novi data frame s nazivom `pozivi_jutro` koji će sadržavati informacije samo o pozivima u jutarnjoj smjeni. Iz nje izbrišite stupac `smjena`.

Rješenje.

Uvjeta na retke pišemo kao prvi argument u uglatim zagradama, a koje stupce želimo pišemo nakon zareza.

```
pozivi_jutro <- pozivi[pozivi$smjena == "jutro", ]
View(pozivi_jutro)
pozivi_jutro$smjena = NULL
```

Provjera uvjeta

Zadatak

Koja informacija je spremljena u varijabli `zauzeto_jutro`, a koja u varijabli `dan`?

```
zauzeto_jutro <- pozivi[  
  pozivi$smjena == "jutro" & pozivi$broj_poziva >= 10, ]  
dan <- pozivi[pozivi$smjena %in% c("jutro", "popodne"), ]
```

Nedostajuće vrijednosti

NA označava nepoznatu ili nezabilježenu vrijednost. Ona **nije nula** i **nije prazan tekst**. Nije jednaka ni samoj sebi, ne tražimo je izrazom `x == NA`, nego funkcijom `is.na()`.

```
is.na(pozivi$trajanje_min)
sum(is.na(pozivi$trajanje_min))
```

Nedostajuće vrijednosti

NA označava nepoznatu ili nezabilježenu vrijednost. Ona **nije nula** i **nije prazan tekst**. Nije jednaka ni samoj sebi, ne tražimo je izrazom `x == NA`, nego funkcijom `is.na()`.

```
is.na(pozivi$trajanje_min)
sum(is.na(pozivi$trajanje_min))
```

`complete.cases()` odabire retke koji nemaju NA ni u jednom stupcu.

Zadatak

Pomoću R-a odredite koliko svaka od sljedećih tablica podataka ima redaka.

```
potpuni <- pozivi[complete.cases(pozivi), ]
bez_NA_u_trajanju <- pozivi[!is.na(pozivi$trajanje_min), ]
```

Petlja for

Petlju for koristimo kada unaprijed znamo po kojim vrijednostima ili objektima želimo prolaziti. Opći oblik je:

```
for (element in skup_vrijednosti) {  
  naredbe  
}
```

Ako želimo proći kroz vektor a duljine 10, zvat ćemo

```
for (i in 1:10) {  
  naredbe s a[i]  
}
```

Petlja while

Petlja `while` ponavlja naredbe dok je zadani logički uvjet `TRUE`. sljedeći postupak zbraja uzastopne prirodne brojeve dok zbroj ne postane veći od 20:

```
i <- 0
zbroj <- 0
while (zbroj < 20) {
  i <- i + 1
  zbroj <- zbroj + i
}
```

Naredba if

Često (u petljama, a i izvan njih) trebamo funkciju if. Jedan primjer korištenja if je sljedeći.

```
kasnjenje<-45.6
if(kasnjenje<15){
  koliko_kasnjenje <- "malo"
} else if (kasnjenje < 30){
  koliko_kasnjenje <- "srednje"
} else {
  koliko_kasnjenje <- "veliko"
}
```