

UVOD U R

STATISTIČKI PRAKTIKUM 2 (P. Lazić, V. Wagner)

1. VJEŽBE

Pošaljite (prazan) mail na

To: petralaz@math.hr
Subject: sp2

Materijali i obavijesti:

<http://web.math.pmf.unizg.hr/~wagner>

Početak rada u R-u I

R je besplatan i može se skinuti s web-stranice

<http://r-project.org>

Rad pod Linux/Unix operativnim sustavom:

Kako bi pokrenuli R pokrećemo terminal.

Sav rad koji ćemo napraviti na kolegiju bilo bi dobro spremiti u neki za to posebno stvoren direktorij. Stoga napravimo direktorij sp2 i uđimo u njega:

```
$ mkdir sp2
```

```
$ cd sp2
```

R pokrećemo tako da upišemo:

```
$ R
```

Početak rada u R-u II

Sada na komandnu liniju možemo upisivati naredbe (jednako kao pod Windowsima).

Iz R okruženja izlazimo:

```
> q()
```

Elementarni tipovi podataka

- ▶ Numerički tipovi:

- > 42
 - > 1.7+2.1
 - > (3.7-85/7)*6+91.2222

- ▶ Tekstualni tipovi:

- > "a"
 - > "duzi tekst"
 - > "posebni znakovi \n \t \\ "

- ▶ Logički tipovi:

- > TRUE
 - > FALSE
 - > 2 < 7
 - > (1 < 9) & ("a"=="b")

Jednostavne funkcije

Funkcije mogu imati razne tipove argumenata:

```
> sqrt(45)
[1] 6.708204
> exp(3)
[1] 20.08554
> nchar("sp2")
[1] 3
```

Funkcije se mogu koristiti u raznim izrazima:

```
> nchar("stat") + 15 > sqrt(265)
[1] TRUE
```

Pomoć o korištenju funkcija je lako dostupna:

```
> help("sqrt")
> ?sqrt
```

Funkcije s više argumenata

Funkcije mogu imati više argumenata. Primjerice funkcija `substr` koja ispisuje pod-string prima argumente `x` (string), `start` (odakle počinje pod-string) i `stop` (gdje završava pod-string). Realizacija može izgledati ovako:

```
> substr("abdce", 1, 3)
[1] "abd"
```

No isti rezultat možemo dobiti i na ove načine:

```
> substr(x="abdce", start=1, stop=3)
> substr("abdce", start=1, stop=3)
> substr("abdce", sto=3, sta=1)
> substr(sto=3, sta=1, x="abdce")
> substr(3, sta=1, x="abdce")
```

Funkcije s *default*-nim argumentima

Neke funkcije, ako im se ne kaže drukčije, mogu imati argumente koji po definiciji imaju neku *razumnu* vrijednost. Primjerice funkcija `log` je takva, ona ima argumente `x` (broj koji želim logaritmirat) i `base` (bazu po kojoj ćemo logaritmirat).

Tako `log(10)` daje vrijednost $\ln 10$, dok `log(100,10)` daje vrijednost $\log_{10} 100 = 2$.

```
> log(100)
[1] 4.60517
> log(100,10)
[1] 2
> log(base=2,x=32)
[1] 5
> log(exp(1))
[1] 1
```


Pohranjivanje vrijednosti u objekte

Izrazi se mogu dodijeliti slobodno imenovanim objektima (operatorima =, ->, <-).

```
> x = sqrt(34)-3  
> tekst <- "nesto"  
> TRUE -> uvjet
```

Objekti vrednuju izraze koji su im dodijeljeni.

```
> x  
[1] 2.830952  
> tekst  
[1] "nesto"  
> uvjet  
[1] TRUE
```

Korištenje objekata

Objekti se mogu, na očekivani način, koristiti u izrazima.

```
> abs(x/7 - 1)
[1] 0.5955783
> uvjet & (2 < 3)
[1] TRUE
> substr(tekst,2,4)
[1] "est"
```

Objekti i radni prostor

Svako dodjeljivanje vrijednosti nekom novom objektu stvara objekt u R-ovom *radnom prostoru*. Funkcija `ls` ispisuje popis svih (aktivnih) objekata.

```
> ls()  
[1] "tekst" "uvjet" "x"
```

Funkcija `rm` uklanja (briše) objekte iz radnog prostora.

```
> rm(uvjet)  
> ls()  
[1] "tekst" "x"  
> uvjet  
Error: object "uvjet" not found
```

Nizovi podataka iste vrste

Konačan niz podataka istog tipa najjednostavnije zapisujemo u *vektor*.

```
> c(4, 5, 12, 8, 9, 12)
[1] 4 5 12 8 9 12
> c("Rudi", "Kristina", "Ivana")
[1] "Rudi"      "Kristina"  "Ivana"
> c(TRUE, FALSE, TRUE, FALSE, TRUE, FALSE)
[1] TRUE FALSE TRUE FALSE TRUE FALSE
```

Kao i sve ostalo i vektori se mogu pohraniti kao objekti.

```
> tezine = c(67, 75, 81, 62, 65, 72, 73, 65, 87, 69)
> tezine
[1] 67 75 81 62 65 72 73 65 87 69
```

Svaki vektor ima svoju duljinu i možemo odabrati element na k -tom mjestu.

```
> length(tezine)
[1] 10
> tezine[4]
[1] 62
```

Korisne funkcije

Vektor je elementarna struktura podataka, pa možemo koristiti funkcije za rad s vektorima.

```
> sum(tezine)
[1] 716
> sum(tezine)/length(tezine)
[1] 71.6
```

Još neke korisne funkcije:

```
> sort(tezine)
[1] 62 65 65 67 69 72 73 75 81 87
> min(tezine)
[1] 62
```

Statističke funkcije

Neke česte statističke funkcije:

```
> mean(tezine)
```

```
[1] 71.6
```

```
> sd(tezine)
```

```
[1] 7.791734
```

```
> range(tezine)
```

```
[1] 62 87
```

```
> summary(tezine)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
62.0	65.5	70.5	71.6	74.5	87.0

Kraći načini kreiranja vektora

Slično kao u Matlab-u:

```
> 1:8  
[1] 1 2 3 4 5 6 7 8
```

Pravilni nizovi brojeva upisani u vektor:

```
> seq(1,10,by=2)  
[1] 1 3 5 7 9  
> seq(1,100, length=10)  
[1] 1 12 23 34 45 56 67 78 89 100
```

Ponavljanje vektora i dijelova vektora:

```
> rep(c(1,2,3), 4)  
[1] 1 2 3 1 2 3 1 2 3 1 2 3  
> rep(c(1,2,3), c(2,4,8))  
[1] 1 1 2 2 2 2 3 3 3 3 3 3 3 3
```


Dijelovi vektora 1

Možemo pojedini član vektora izvući.

```
> tezine[1]
[1] 67
> tezine[length(tezine)]
[1] 69
```

Negativni indeks izbacuje članove vektora.

```
> a=1:8
> a[-4]
[1] 1 2 3 5 6 7 8
```

Ispišimo dijelove ili izbacimo neke članove vektora `tezine`:

```
> tezine[3:7]
[1] 81 62 65 72 73
> tezine[-c(1,8)]
[1] 75 81 62 65 72 73 87 69
```

Dijelovi vektora 2

Zadatak

Unesite proizvoljan vektor težina (u rasponu od 40 do 100) i ispišite sve težine veće od 70.

Promjena vrijednosti u vektoru

Vrijednost pojedinog elementa mijenjamo na sljedeći način:

```
> x=1:10  
> x[1]=-1  
> x  
[1] -1  2  3  4  5  6  7  8  9 10
```

Slično radimo za vektore:

```
> x[1:5] = -x[1:5]  
> odd = 1:10 %% 2  
> odd  
[1] 1 0 1 0 1 0 1 0 1 0  
> x[odd==1] = x[odd==1] * 100
```

Isto se može napraviti za vektore s negativnim cijelim brojevima:

```
> x[-(1:5)] = rep(13, 5)
```

Vektorska aritmetika

Funkcije se primjenjuju na svaki pojedini element.

```
> visine = c(1.65, 1.56, 1.9, 1.32, 1.54,  
+ 1.78, 1.65, 1.5, 1.65, 2.03)  
> bmi = tezine/visine^2  
> round(bmi,1)  
[1] 24.6 30.8 22.4 35.6 27.4 22.7 26.8 28.9 32.0 16.7
```

Ako je jedan od vektora u binarnoj operaciji kraći, on će se ponoviti onoliko puta koliko je manji (*recycling*).

```
> x=seq(100,500, by=100)  
> y=1:2  
> x+y  
[1] 101 202 301 402 501
```

Warning message:

```
In x + y : longer object length is not a multiple  
of shorter object length
```

Nedostupne vrijednosti I

U nekim slučajevima vrijednosti elemenata vektora ne moraju biti poznate. (Npr. za neke davne godine podaci su izgubljeni, a za tekuću godinu se još ne znaju.) U takvim slučajevima na mjesto rezervirano za nepoznatu vrijednost upisujemo posebnu vrijednost NA (*marker*) (*Not Available*).

Postoji još jedna vrsta brojeva koji su nedostupni. To brojevi nastali izvođenjem (inače) nedefiniranih aritmetičkih operacija NaN (*Not a Number*). (Kao rezultat $0/0$ ili $\text{Inf}-\text{Inf}$ dobivamo NaN-ove.) Funkcija `is.na(x)` ispituje koje vrijednosti u vektoru nisu poznate, a `is.nan(x)` gdje se nalaze NaN-ovi. Ostale funkcije: `na.omit`, `complete.cases`.

Nedostupne vrijednosti II

```
> x=c(1,2,3,NA,0/0)
> x
[1] 1 2 3 NA NaN
> is.na(x)
[1] FALSE FALSE FALSE TRUE TRUE
> is.nan(x)
[1] FALSE FALSE FALSE FALSE TRUE
```

Zadatak

- (a) Ispišite vektor $x=(1,2,3,NA,5,NA)$ bez nedostupnih vrijednosti.
- (b) Učitajte podatke `airquality` i ispišite koliko redaka ima jedan ili više nedostupnih podataka.

Faktori za kategorizirane podatke

Kategorizirani podaci mogu se kodirati kao brojevi i *stringovi*.

```
> spol = rep(1:2, c(5,5))  
> tretman = rep(c("kontrola", "operacija"), c(5,5))
```

Općenito, bolje je te podatke pretvoriti u *faktore*.

```
> spol=factor(spol, level=c(1,2), labels=c("m","f"))  
> spol  
[1] m m m m m f f f f f  
Levels: m f
```

Za vektore stringova faktorizacija je još lakša.

```
> tretman=factor(tretman)  
> tretman  
[1] kontrola kontrola kontrola kontrola kontrola operacija  
[8] operacija operacija operacija  
Levels: kontrola operacija
```


Korisne stvari za rad s faktorima

Frekvencijska tablica:

```
> table(spol)
spol
m f
5 5
```

Računanje po grupama s funkcijom

```
tapply(x, INDEX, FUN, ...)
```

- na vektor vrijednosti x primjenjujemo funkciju FUN po kategorijama određenim faktorom $INDEX$.

Zadatak

Za deset slučajno odabranih osoba prikupili ste podatke o spolu i prihodima.

```
> prihod = c(12,13,14,12,9,11,23,6,11,17)
> spol=factor(spol, level=c(1,2), labels=c("m","f"))
```

Odredite srednju vrijednost i standardnu devijaciju prihoda po spolovima.

Matrice

Osnovne podatke možemo svrstati u matricu.

```
> matrix(1:9, nrow=3, ncol=3)
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Dovoljno je navesti samo broj stupaca ili sam broj redaka za formiranje matrice.

```
> matrix(c("a", "b", "c", "d"), nrow=2, byrow=TRUE)
      [,1] [,2]
[1,] "a"  "b"
[2,] "c"  "d"
```

Matrice kao kombinacije vektora

Matrice se mogu sastaviti po stupcima (ali nužno istog tipa!).

```
> mat1=cbind(visine, tezine, bmi)
```

```
> mat1
```

	visine	tezine	bmi
[1,]	1.65	67	24.60973
[2,]	1.56	75	30.81854
[3,]	1.90	81	22.43767
[4,]	1.32	62	35.58310
[5,]	1.54	65	27.40766
[6,]	1.78	72	22.72440
[7,]	1.65	73	26.81359
[8,]	1.50	65	28.88889
[9,]	1.65	87	31.95592
[10,]	2.03	69	16.74392

Indeksiranje matrica

Pojedini elementi iz matrice se mogu izvući korištenjem [] i brojeva koji označavaju mjesto u matrici.

```
> mat1[3,3]  
      bmi  
22.43767
```

2. red matrice dobivamo ovako:

```
> mat1[2,]  
visine  tezine      bmi  
1.56000 75.00000 30.81854
```

3. stupac ovako:

```
> mat1[,3]  
[1] 24.60973 30.81854 22.43767 35.58310 27.40766 22.72440  
26.81359 28.88889  
[9] 31.95592 16.74392
```

Indeksiranje matrica - imena redaka i stupaca

Tamo gdje je definirano možemo služiti imenima redaka i stupaca.

```
> mat1["tezine"]  
[1] 67 75 81 62 65 72 73 65 87 69
```

Aritmetika matrica

Slično kao kod vektora:

```
> mat2 = matrix(1:9, nrow=3)
> mat3 = matrix(1, nrow=3, ncol=3)
> mat2 + mat3
> mat2 + 10
```

Funkcije se mogu izvoditi po retcima i stupcima.

```
> apply(mat2, 1, min) # minimum elemenata po recima
> apply(mat2, 2, mean) # art. sred. elem. po stupcima
```

Standardne matrice operacije se mogu izvoditi.

```
> t(mat2)
> mat2 %*% mat3
```

Okviri podataka : kombiniranje različitih tipova podataka

Vektore svih tipova podataka možemo kombinirati.

```
> pred = bmi > 25  
> pusaci = factor(c(1,2,1,2,3,3,1,1,2,3), levels=1:3,  
+ labels=c("da","ne","nekoc"))  
> df1 = data.frame(tezine, visine, pred, pusaci)  
> df1[1:7,]
```

	tezine	visine	pred	pusaci
1	67	1.65	FALSE	da
2	75	1.56	TRUE	ne
3	81	1.90	FALSE	da
4	62	1.32	TRUE	ne
5	65	1.54	TRUE	nekoc
6	72	1.78	FALSE	nekoc
7	73	1.65	TRUE	da

Okviri podataka (*data frame*) su osnovna struktura podataka za svaku složeniju analizu podataka.

Okviri podataka : indeksiranje

Svako indeksiranje koje je vrijedilo za matrice vrijedi i za okvire podataka:

- ▶ vektori cijelih brojeva
- ▶ logički vektori
- ▶ imena stupaca

Posebno stupci/varijable mogu se doseći korištenjem znaka \$:

```
> df1$tezine  
[1] 67 75 81 62 65 72 73 65 87 69
```

Ime ne mora biti do kraja napisano.

```
> df1$tez  
[1] 67 75 81 62 65 72 73 65 87 69
```

Funkcije su također objekti

Definiciju svake funkcije dobivamo tako da napišemo njezino ime.

```
> log  
function (x, base = exp(1)) .Primitive("log")
```

Zadatak

Definirajte vlastitu funkciju za srednju vrijednost vektora brojeva (u definiciji možete koristiti sve funkcije osim `mean`).

Klase, objekti, metode

Iste funkcije mogu imati različite vrste rezultata ovisno o podacima.

```
> summary(tezine)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 62.0   65.5   70.5   71.6   74.5   87.0

> summary(pusaci)
  da    ne nekoc
  4     3     3
```

Ovisi o klasi (*class*) objekta:

```
> class(tezine)
[1] "numeric"
> class(pusaci)
[1] "factor"
```

Klase, objekti, metode

`summary` je generička metoda koja ima različite *metode* za različite objekte.

```
> summary  
function (object, ...)  
UseMethod("summary")  
<environment: namespace:base>
```

Pohranjivanje radnog prostora

Kako bi pohranili sve objekte u radnom prostoru to radimo naredbom

```
> save.image()
```

koja pohranjuje radni prostor u datoteku `.RData`. (Može se izabrati i drugo ime i lokacija.)

Učitavanje pohranjenog prostora u tekući radni prostor radimo naredbom

```
> load(".RData")
```

Tekući direktorij i njegov sadržaj dobivamo sljedećim naredbama.

```
> getwd()  
> dir(getwd())
```

A direktorij mijenjamo ovako:

```
> setwd("~/")
```

Čitanje datoteka u R-u

Podaci su često pohranjeni u raznim datotekama, pa kako ne bi ručno unosili podatke, učitati ćemo ih iz datoteke.

Za *tekstualne datoteke* imamo sljedeće mogućnosti

- ▶ Fleksibilna naredba `read.table`.
- ▶ Za velike količine podataka `scan`.

Mnogi podaci se nalaze u Excel datotekama. Za takve na raspolaganju su nam sljedeće mogućnosti.

- ▶ izvoz u format *tab-delimited/csv text files*
- ▶ korištenje paketa `xlsReadWrite` (Windows)
- ▶ paket `gdata` (traži Perl - lako pod Unix-om)

Postoje još paketi unošenje podataka iz baza podataka (MySQL) i drugog statističkog software-a (Matlab, Ocatve, SAS,...).

Primjer unošenja podataka iz Excel datoteke

Prvo treba podatke izvesti u csv-format (File → Save As →...). (Ovo možemo napraviti i pod Linux-om putem *Open Office-a*.)
Sada podatke unosimo na sljedeći način:

```
podaci <- read.csv("podaci.csv", header=T, dec=".", sep=";")
```

(Uočimo da Excel koristi decimalni zarez umjesto točke, pa smo to i naznačili. Podaci su prilikom izvoza u csv-datoteku (ako drukčije ne specificiramo) razdvojeni s ';' , pa je i to spomenuto.)

Spremanje rezultata

Cut & paste za manje stvari.

save i load za spremanje pojedinih objekata.

```
> save(df1,file="df1.RData")  
> load("df1.RData")
```

Preusmjeravanje numeričkih rezultata u datoteku.

```
> sink("rezultati.txt")  
> summary(tezine)  
> sink()  
> file.show("rezultati.txt")
```


Spremanje koda

Naredba `history` ispisuje zadnje naredbe, a pomoću `savehistory` spremamo te naredbe.

```
> history()
> savehistory()
> load(".Rhistory")
```

Ime datoteke se može promijeniti.

Ako imamo datoteku s naredbama koje želimo izvršiti to radimo putem naredbe `source`.

```
> source("naredbe.txt")
```

Grafički prikazi - uvod

Grafika je vrlo značajan dio R okruženja.

Moguće je koristiti brojne vrste statističkih grafova kao i napraviti potpuno nove tipove grafova.

R omogućava interaktivno korištenje grafike.

R prilikom pokretanja ima spreman grafički *device driver*, koji će prilikom izdavanja naredbe pokrenuti poseban grafički prozor. Makar se to radi automatski, dobro je znati da se pritom koristi naredba `X11()` (pod UNIX-om) ili `windows()` (pod Windows-om).

Grafički prikazi - tipovi naredbi

Naredbe se dijele u tri tipa:

- ▶ Funkcije za crtanje **visokog nivoa** (*high-level*) crtaju grafove, po mogućnosti s osima, oznakama, nazivima ...
- ▶ Funkcije za crtanje **niskog nivoa** (*low-level*) dodaju na postojeći graf dodatne informacije poput novih točaka, linija i oznaka.
- ▶ **Interaktivne** grafičke funkcije koje omogućavaju dodavanje i uzimanje podataka na grafu.

Bez dodatnih paketa u R-u koristimo se samo *osnovnom* grafikom. Dodatni paketi koji mogu biti korisni su `grid` i `lattice`.

Grafičke funkcije visokog nivoa - plot

Jedna od najosnovnijih grafičkih funkcija visokog nivoa je plot. Funkcija plot može kao argument primiti jedan ili više argumenata, moguće različitog tipa.

Zadatak

Provjerite output sljedećih poziva funkcije plot

```
> plot(x,y) \\ x i y numerički vektori  
> plot(m) \\ m nx2 matrica  
> plot(x) \\ x numerički vektor  
> plot(f) \\ f faktor  
> plot(f,x) \\ x numerički vektor, f faktor
```

plot za okvire podataka i liste objekata

Za okvir podataka (data frame) *df1* naredba `plot(df1)` crta usporedbu svih vektora podataka u tom okviru.

```
>plot(df1)
```

Neka je *expr* lista objekata spojena s znakom '+'. Tada `plot(~expr)` crta usporedbe objekata u listi, a `plot(y~expr)` usporedbu *y* sa objektima iz *expr*.

```
>plot(~tezine+visine)
>plot(bmi~tezine+visine)
```

Normalni vjerojatnosni graf i histogram

```
> qqnorm(x)  
> qqline(x)
```

Prva naredba crta točke, a druga pravac kod normalnog vjerojatnosnog grafa.

```
> hist(tezine)
```



Crta histogram podataka. Mogući argument `nclass=n` ili `breaks=b`.

Grafičke funkcije niskog nivoa - dodavanje točaka i linija

```
> plot(visine,tezine)
> points(visine,tezine,col='red')
> lines(visine,tezine,lty=2,lwd=2)
```

Prva naredba dodaje crvene točke na crtež tako da prvu koordinatu uzme iz vektora x , a drugu iz vektora y . Druga naredba dodaje deblju isprekidanu liniju koja spaja točke.

```
> abline(a,b)
```

Dodaje pravac s koeficijentom smjera b i slobodnim članom a na postojeći crtež.

Vjerojatnosne funkcije

R može zamijeniti cijelu zbirku statističkih tablica. Od vjerojatnosnih funkcija R zna odrediti vrijednosti funkcije distribucije, gustoće, funkciju kvantila i zna simulirati uzorak iz zadane distribucije.

Distribucija	Oznaka u R-u	dodatni argumenti
beta	beta	shape1, shape2, ncp
binomna	binom	size, prob
Cauchyjeva	cauchy	location, scale
χ^2	chisq	df, ncp
eksponencijalna	exp	rate
Fisherova	f	df1, df2, ncp
Γ	gamma	shape, scale
geometrijska	geom	prob
hipergeometrijska	hyper	m, n, k
normalna	norm	mean, sd
Poissonova	pois	lambda
Studentova (t)	t	df, ncp
uniformna	unif	min, max
Wilcoxonova	wilcox	m, n

Simulacija uzorka duljine n

Kod simulacija ispred imena distribucije lijepimo 'r'. Prvi argument funkcije `xxxxx` je duljina uzorka, a onda slijede argumenti distribucije.

Zadatak

Simulirajte uzorak duljine 200 iz $B(5, 0.3)$ i nacrtajte histogram relativnih frekvencija.

Funkcija gustoće

Kako bi dobili vrijednost funkcije gustoće iz neprekidne razdiobe x , ispred oznake stavljamo 'd'. Prvi argument funkcije $dxxxx$ je x koji označava točku u kojoj želimo izračunati vrijednost funkcije gustoće.

Zadatak

Nacrtajte na prethodnom histogramu graf funkcije gustoće razdiobe $N(1.5, 1.05)$.

Funkcije distribucije i kvantila

Za dobivanje vrijednosti funkcije distribucije razdiobe `xxxx`, ispred oznake stavljamo 'p'. Prvi argument funkcije `pxxxx` je točka u kojoj želimo izračunati vrijednost funkcije.

Za dobivanje vrijednosti q -kvantila radimo ispred oznake razdiobe pišemo 'q', a prvi argument funkcije `qxxxx` je q .

Funkcije imaju ugrađen logički argument `lower.tail`, koji ispituje gledamo od početka ili od kraja. Odnosno (ako je vrijednost `TRUE`) vjerojatnost da vrijednost sl. varijable bude manja ili jednaka od x , ili ako je argument `FALSE` vjerojatnost da je vrijednost veća od x . Slično je i za gornji odnosno donji kvantil.

Zadatak

- (a) Odredite p -vrijednost dvostranog t -testa s 13 stupnjeva slobode ako je realizacija testne statistike 2.34.
- (b) Odredite 90% pouzdani interval za distribuciju $F(2, 3)$.

Kolmogorov - Smirnovljev test

Jedan od najpoznatijih testova kojima se ispituje pripadnost slučajnog uzorka nekoj distribuciji provodi se relativno lako u R-u.

```
ks.test(uzorak,"ime_distribucije", argumenti_distribucije)
```

Zadatak

Promatrajmo simulirani uzorak duljine 200 iz $B(5, 0.3)$. Testirajmo pripada li on distribuciji $N(1.5, 1.05)$.

Uvjetne naredbe, grupirani izrazi i petlje

R ima mogućnost zadavanja uvjetnih naredbi. Oblik tih naredbi je standardan (slično kao u C-u)

```
> if (iz_1) iz_2 else iz_3
```

Naredbe se mogu grupirati u vitičastim zagradama ovako {iz_1; ... iz_n;}. Dakle, R podržava grupirane izraze.

U R-u postoji mogućnost zadavanja for petlji. One su oblika

```
> for (name in iz_1) iz_2
```

ime je varijabla, iz_1 je vektor (obično niz poput 1:10), a iz_2 je grupirani izraz.

Drugi oblici petlji i prekidi petlji

Drugi oblici petlji su

```
> repeat  iz
```

```
i
```

```
> while (uvjet) iz
```

Naredba `break` prekida petlju i izlazi iz nje, dok naredba `next` prekida trenutni ciklus i prelazi na sljedeći. `break` je jedini način prekidanja `repeat` petlje.