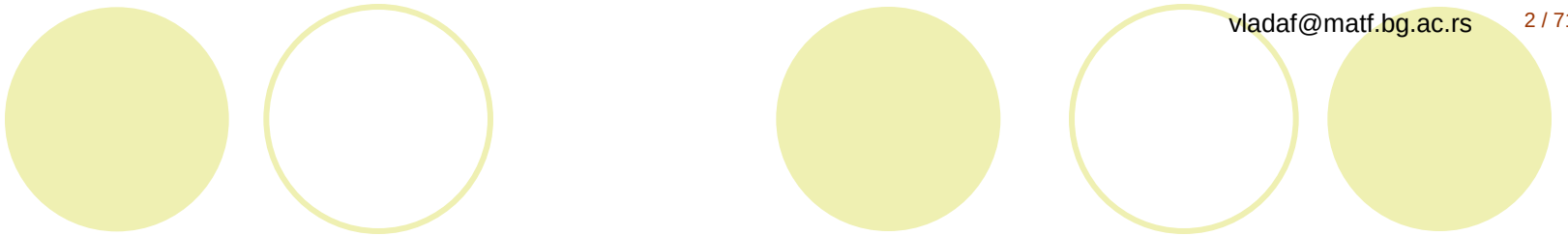
Primjena računara u biologiji

Vladimir Filipović

vladaf@matf.bg.ac.rs



Elementarna statisika
i
R

Predmet i ciljevi statistike

Predmet statistike je prikupljanje podataka, njihova interpretacija i, na osnovu tih podataka, donošenje zaključaka o fenomenu koji je predmet proučavanja.

Predmet proučavanja statistike obuhvata i sve ostale procese prikupljanja znanja o fenomenu kao što je pronalaženje obrazaca u podacima i sl.

Statistika je naučni pristup analizi podataka kako bi se obezbedio maksimum razumevanja, interpretacije i korisnosti podataka.

Pod podacima se najčešće podrazumevaju numerički, kvantitativni podaci, a zadatak statistike je da ih preradi i pretvori u informacije.

Predmet i ciljevi statistike (2)

Neka od pitanja na koja principi i metodologija statistike daje odgovore su:

- Kakve podatke i koliko podataka treba prikupiti?
- Kako organizovati i interpretirati podatke?
- Kako analizirati podatke i doneti zaključak?
- Kako proceniti snagu donetog zaključka?

Glavni ciljevi statistike su da omogući:

- donošenje zaključaka o populaciji na osnovu informacija o uzorku, što uključuje i procenu stepena nepouzdanosti ovakvog zaključka i
- dizajniranje procesa i određivanje obima uzorkovanja tako da se na osnovu tih podataka može doneti valjan zaključak o populaciji.

Predmet i ciljevi statistike (3)

Statistika je autonomna matematička nauka koja je od svojih početaka povezana sa demografijom, ekonomijom, kao i sa prirodnim naukama (fizikom, hemijom, biologijom).

Preko naučnog metoda i induktivnog zaključivanja statistika je povezana sa filozofijom nauke, a naglasak statistike na učenju o podacima i predviđanju povezuje je sa naukom o odlučivanju.

Informatika i statistika imaju zajedničku oblast, a to su podaci.

Savremena statistika je nezamisliva bez intenzivne upotrebe računara, dok s druge strane razvoj statističkih metoda u velikoj meri doprinosi razvoju informatičke teorije i prakse.

Primena statistike u nauci

Savremeni naučni metod u mnogim naukama je nezamisliv bez primene statistike u svim fazama, bez obzira na vrstu nacrta istraživanja:

- Dizajn istraživanja
- Prikupljanje podataka
- Opisivanje podataka
- Analiziranje podataka
- Donošenje zaključaka

Primena statistike u nauci (2)

Dizajn istraživanja

Pošto se rezultati istraživanja, bez obzira na vrstu dizajna, baziraju na prikupljenim podacima, uloga statistike počinje još u fazi dizajna istraživanja.

Potrebno je planirati na koji način će se prikupiti validni podaci koji će kasnije moći da se obrade, kako bi se dobio validan rezultat naučnog istraživanja.

Uloga statistike u planiranju istraživanja je toliko značajna da se često kaže kako istraživanje koje nije uključilo statistiku ili statističara od samog početka može da bude samo post mortem konstatacija o razlozima propasti istraživanja.

Primena statistike u nauci (3)

Prikupljanje podataka

Prikupljanje podataka najčešće podrazumeva uzimanje uzoraka iz neke veće populacije.

Da rezultat ne bi bio deformisan, uzorak mora da bude nezavistan, što se postiže slučajnim izborom individualnih elemenata populacije.

Uzorak ne sme da obuhvati samo deo populacije koji je značajan za istraživanje, jer bi u suprotnom dobili iskrivljenu sliku populacije.

Primena statistike zahteva i dovoljan stepen varijabiliteta između elemenata uzoraka.

Primena statistike u nauci (4)

Opisivanje podataka

Nakon što su podaci prikupljeni, potrebno ih je organizovati i sumirati, tako da se stekne globalna slika o procesu koji je predmet istraživanja.

U statistici se to radi na dva načina

- grafički i
- deskriptivnom statistikom

Grafički prikaz podataka vrši se putem dijagrama koji imaju neke standardne karakteristike. To mogu biti:

- stubičasti dijagram,
- histogram,
- poligon,
- pita i dr.

Primena statistike u nauci (5)

Opisivanje podataka

Grafički prikaz često može na prvi pogled da otkrije osobine analiziranih podataka, koje se kasnije preciznije kvantifikuju putem deskriptivne statistike.

Deskriptivna statistika opisuje podatke tako što pronalazi njihove važne karakteristike, kao što su:

- mere centralne tendencije,
- mere disperzije i
- povezanost sa drugim podacima

Ako su podaci kategorički, oni se obično grupišu, a svakoj grupi se pridodaju frekvencije pojavljivanja elemenata u grupi. To se radi zbog njihove lakše upotrebe u narednim fazama obrade podataka.

Primena statistike u nauci (6)

Analiziranje podataka

Da bi se doneo ispravan zaključak, neophodno je analizirati podatke. Postoji mnogo načina za analizu podataka, a neki od njih su:

- utvrđivanje povezanosti dve varijable (npr. aritmetičke sredine dvaju uzoraka) putem regresione analize;
- utvrđivanje oblika distribucije podataka i dr.

Pre nego što se pređe na donošenje zaključaka o istraživanju, neophodno je definisati i interval poverenja, kao meru pouzdanosti donetog zaključka.

U društvenim naukama obično se za stepen pouzdanosti uzima 95%, što znači da postoji šansa od 5% da je doneti zaključak pogrešan. Iz toga se izračunava interval poverenja za konkretan slučaj.

Primena statistike u nauci (7)

Donošenje zaključaka

Suština i krajnji rezultat istraživanja i proučavanja podataka je donošenje zaključka. U tu svrhu se koriste statistički testovi koji treba da daju odgovor na pitanje šta jeste, a šta nije osobina fenomena koji je predmet istraživanja.

Celokupne proračune danas obavljaju računari, ali oni sami nisu sposobni da protumače dobijeni zaključak. To može samo istraživač koji je uspešno realizovao sve faze istraživačkog rada.

Da bi došao do ispravnog zaključka i uspešno završio svoje istraživanje, istraživač mora da postavi hipoteze, izabere odgovarajući statistički test, da izračuna test statistiku i na osnovu dobijenih vrednosti odluči o tome da li se nulta hipoteza prihvata ili odbacuje.

Uvod u R

Instalacija

Programski paket R se (za operativni sistem Windows) može dovući sa lokacije: https://cran.r-project.org/bin/windows/base/ .

R Sesija

Kada se pokrene R, pojavi se konzola koja očekuje unos tj. naredbe.
Iza odzivnog znaka > korisnik unosi brojeve i realizuje izračunavanja:

```
> 1 + 2
[1] 3
```

Dodela vrednosti promenljive

Vrednosti se dodeljuju promenljivima pomoću naredbi dodele = (češće se koristi <- kao operator naredba dodele).
Prikaz vrednosti promenljive se dobija unosom imena te promenljive iza odzivnog znaka R sistema:

```
> x = 1
> x
[1] 1
```

Uvod u R (2)

Funkcije

Funkcija se poziva tako što se navede njeno ime, iza koga slede zagrade, a u zagradama nula, jedan ili više argumenata.

Primer. Funkcija c kombinuje tri numeričke vrednosti u jedan vektor

```
> c(1, 2, 3)
[1] 1 2 3
```

Komentari

Tekst iza znaka # je komentar i njega prilikom izvršavanja R sistem ignoriše.

```
> 1 + 1      # this is a comment
[1] 2
```

Paketi sa proširenjima

Ponekad je potrebna dodatna funkcionalnost, pored one koju obezbeđuje jezgro R sistema. Ako treba instalirati paket sa proširenjima, poziva se funkcija install.packages i prate uputstva koje daje sistem:

```
> install.packages()
```

Uvod u R (3)

Sistem pomoći

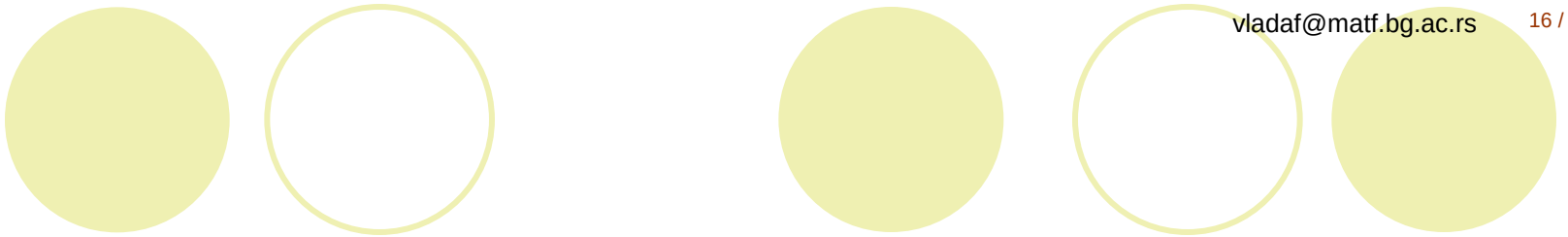
Sistem R obezbeđuje detaljnu dokumentaciju.

Primer. Naredbom ?c ili help(c) se dobija dokumentacija o funkciji c u sistemu R.

```
> help(c)
```

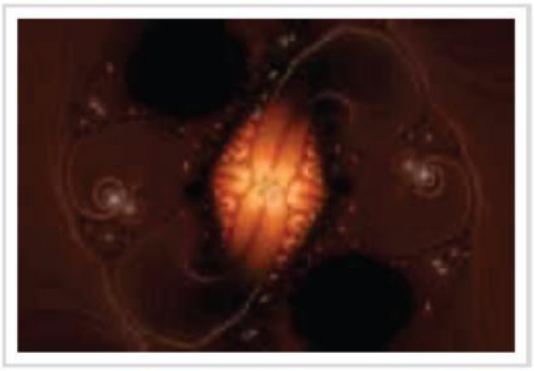
Ako korisnik nije siguran kako se tačno naziva ono što ga interesuje, može izvršiti heurističku pretragu kroz sistem pomoći, funkcijom apropos, i na taj način dobiti spisak pojmova iz dokumentacije koji „liče“ na ono što interesuje korisnika

```
> apropos("nova")
[1] "anova"                 "anova.glm"
   ....
```



Osnovni tipovi podataka

Osnovni tipovi podataka



Postoji nekoliko osnovnih tipova podataka u R-u, koji se često sreću prilikom izračunavanja.

Uako liče na tipove u klasičnim programskim jezicima, ovi tipovi ponekad mogu da iznenade.

Osnovni tipovi će biti bolje objašnjeni kroz primere, a ne prolaskom kroz suvoparnu specifikaciju.

Radi očuvanja jednostavnosti, razmatranje pojma vektora se odlaže za kasnije, a na početku će se koristiti samo vektori dužine jedan.

Osnovni tipovi su:

- Brojevni tip (Numeric)
- Celobrojni tip (Integer)
- Kompleksni tip (Complex)
- Logički tip (Logical)
- Znakovni tip (Character)

Brojevni tip

Ovo je podrazumevani tip za izračunavanje u R-u. Brojevi se zapisuju kao decimalne vrednosti.

Primer. Sledećom naredbom se promenljivoj x dodeljuje decimalna vrednost, pa će promenljiva x po izvršenju naredbe sadržati vrednost numeričkog tipa:

```
> x = 10.5        # assign a decimal value
> x               # print the value of x
[1] 10.5
> class(x)        # print the class name of x
[1] "numeric"
```

Primer. Ako celobrojnu vrednost dodelimo promenljivoj k, vrednost promenljive će, po izvršenju naredbe, biti brojevnog tj. numeričkog tipa:

```
> k = 1
> k               # print the value of k
[1] 1
> class(k)        # print the class name of k
[1] "numeric"
```

Na sledeći način se može potvrditi da promenljiva k nije celobrojnog tipa:

```
> is.integer(k)  # is k an integer?
[1] FALSE
```

Celobrojni tip

Da bi se u R-u dobila celobrojna vrednost, poziva se funkcija **as.integer**. Provera da li je argument celobrojan se realizuje korišćenjem **is.integer** funkcije.

Primer. Sledećim naredbama se dodeljuje celobrojna vrednost i proverava da li je argument tipa integer:

```
> y = as.integer(3)
> y               # print the value of y
[1] 3
> class(y)        # print the class name of y
[1] "integer"
> is.integer(y)  # is y an integer?
[1] TRUE
```

Primer. Decimalna numerička vrednost se može konvertovati u celobrojni tip korišćenjem funkcije as.integer:

```
> as.integer(3.14)    # coerce a numeric value
[1] 3
```

Primer. Od niske znakova koja predstavlja decimalni broj se funkcijom as.integer može dobiti ceo broj koji se sadrži u toj nisci:

```
> as.integer("5.27")  # coerce a decimal string
[1] 5
```

Celobrojni tip (2)

Sa druge strane, pokušaj da se od niske koja ne predstavlja decimalni broj dobije celobrojna vrednost dovodi do problema.

Primer. Pretvaranje niske „Joe“ u ceo broj dovodi do sledećeg rezultata:

```
> as.integer("Joe")   # coerce an non-decimal string
[1] NA
Warning message:
NAs introduced by coercion
```

Često se javlja potreba da se logičke vrednosti pretvaraju u cele brojeve. U tom slučaju, u jeziku R, vrednosti TRUE odgovara 1, a vrednosti FALSE odgovara 0.

Primer. Ilustruje kako se logičke vrednosti pretvaraju u cele brojeve:

```
> as.integer(TRUE)    # the numeric value of TRUE
[1] 1
> as.integer(FALSE)   # the numeric value of FALSE
[1] 0
```

Kompleksni tip

Podatak kompleksnog tipa se definiše korišćenjem i, koje predstavlja imaginarnu jedinicu.

Primer. Kreiranje kompleksnog broja i provera tipa za kompleksnu vrednost:

```
> z = 1 + 2i      # create a complex number
> z               # print the value of z
[1] 1+2i
> class(z)        # print the class name of z
[1] "complex"
```

Primer. U okviru brojevnog tipa, ne može se izračunati kvadratni koren negativnog broja:

```
> sqrt(-1)        # square root of -1
[1] NaN
Warning message:
In sqrt(-1) : NaNs produced
```

Primer. Ako se pređe u kompleksni tip, onda se može izračunati kvadratni koren negativnog broja:

```
> sqrt(-1+0i)     # square root of -1+0i
[1] 0+1i
```

Kompleksni tip (2)

Korišćenjem funkcije **as.complex** se na osnovu vrednosti argumenta određuje odgovarajući kompleksni broj.

Primer. Kvadratni koren negativnog broja se može izračunati tako što se prvo odredi odgovarajući kompleksni broj, a potom se izračuna njegov kvadratni koren:

```
> sqrt(as.complex(-1))
[1] 0+1i
```

Logički tip

Vrednost logičkog tipa se često dobija tako što se uporede vrednosti dve promenljive.

Primer. Vrednost promenljive z se dobija poređenjem vrednosti promenljivih x i y:

```
> x = 1; y = 2   # sample values
> z = x > y      # is x larger than y?
> z              # print the logical value
[1] FALSE
> class(z)       # print the class name of z
[1] "logical"
```

Standardne logičke operacije su:

konjukcija (i) - **&**, disjunkcija (ili) – **|**, negacija (ne) - **!**.

Primer. Ilustruje izvršavanje logičkih operacija:

```
> u = TRUE; v = FALSE
> u & v          # u AND v
[1] FALSE
> u | v          # u OR v
[1] TRUE
> !u             # negation of u
[1] FALSE
```

Logički tip (2)

Primer. Dodatni detalji i informacije o odgovarajućim logičkim operacijama mogu se naći u dokumentaciji sistema R:

```
> help("&")
```

Znakovni tip

Podatak znakovnog tipa u jeziku R predstavlja znakovnu nisku.

Funkcijom **as.character** se na osnovu date vrednosti dobija niska koja predstavlja tu vrednost.

Primer. Kreiranje niske na osnovu datog broja:

```
> x = as.character(3.14)
> x             # print the character string
[1] "3.14"
> class(x)      # print the class name of x
[1] "character"
```

Dve niske se nadovezuju jedna na drugu korišćenjem funkcije **paste**.

Primer. Ilustruje kako se jedna niska nadovezuje na drugu.

```
> fname = "Joe"; lname ="Smith"
> paste(fname, lname)
[1] "Joe Smith"
```

Znakovni tip (2)

Formiranje niski na komplikovaniji način, kombinovanjem raznovrsnih argumenata se obično postiže funkcijom **sprintf** (čija je sintaksa slična sintaksi funkcije printf programskog jezika C).

Primer. Ilustruje kako se kreira niska kombinovanjem argumenata koji mogu biti različitog tipa:

```
> sprintf("%s has %d dollars", "Sam", 100)
[1] "Sam has 100 dollars"
```

Za izdvajanje podniske date niske, koristi se funkcija **substr**.

Primer. Ilustruje kako se iz date niske izdvaja podniska od treće do dvanaeste pozicije:

```
> substr("Mary has a little lamb.", start=3, stop=12)
[1] "ry has a l"
```

Funkcijom sub se realizuje zamena jedne podniske drugom u datoj nisci.

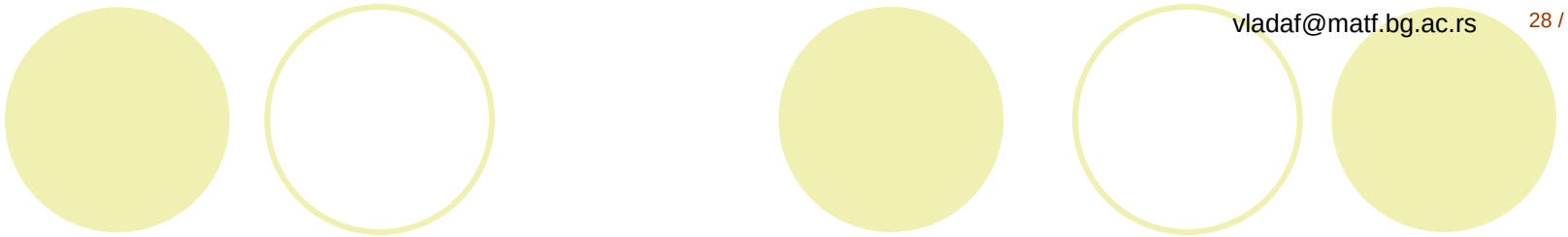
Primer. Ilustruje kako se niska „little” menja sa niskom „big”:

```
> sub("little", "big", "Mary has a little lamb.")
[1] "Mary has a big lamb."
```

Znakovni tip (3)

Primer. Informacije o prethodno pobrojanim i o dodatnim funkcijama za manipulaciju niskama mogu se naći u dokumentaciji sistema R:

```
> help("sub")
```



Vektori

Vektori

Vektor je sekvenca podataka istog osnovnog tipa.

Elementi tj. podaci u vektoru se nazivaju ***komponentama*** vektora, ili ***članovima*** vektora.

Primer. Formiranje vektora od tri broja 2,3 i 5:

```
> c(2, 3, 5)
[1] 2 3 5
```

Primer. Formiranje vektora od tri logičke vrednosti:

```
> c(TRUE, FALSE, TRUE, FALSE, FALSE)
[1]  TRUE FALSE  TRUE FALSE FALSE
```

Primer. Formiranje vektora od pet niski:

```
> c("aa", "bb", "cc", "dd", "ee")
[1] "aa" "bb" "cc" "dd" "ee"
```

Broj elemenata vektora daje funkcija **length**.

Primer. Ilustruje određivanje dužine vektora:

```
> length(c("aa", "bb", "cc", "dd", "ee"))
[1] 5
```

Kombinovanje vektora

Vektor se kombinuju korišćenjem funkcije **c**.

Primer. Ilustruje kako se kombinuju dva vektora različite dužine, čiji su članovi različitih tipova:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> c(n, s)
[1] "2"  "3"  "5"  "aa" "bb" "cc" "dd" "ee"
```

Uočava se da su komponente rezultujućeg vektora tipa niske.

Dakle, prilikom kombinovanja vektora, da bi se postiglo da sve komponente rezultujućeg vektora budu istog tipa, vrši se ***konverzija tipa*** komponenti rezultujućeg vektora u najopštiji tip argumenata.

Aritmetika sa vektorima

Mogu se izvršavati aritmetičke operacije nad vektorima. One se izvode član-po-član, tj. tako što se data aritmetička operacija izvrši nad svakim parom odgovarajućih članova vektora.

Primer. Ilustruje kako se sabiraju vektori:

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> a + b
[1]  2  5  9 15
```

Primer. Ilustruje kako se vektor množi skalarom (brojem):

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> 5 * a
[1]  5 15 25 35
```

Primer. Ilustruje kako se oduzima jedan vektor od drugog:

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> a - b
[1]  0  1  1 -1
```

Aritmetika sa vektorima (2)

Primer. Ilustruje kako se množe dva vektora:

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> a * b
[1]  1  6 20 56
```

Primer. Ilustruje kako se jedan vektor deli sa drugim:

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> a / b
[1] 1.000 1.500 1.250 0.875
```

Ako su vektori-operandi različite dužine, tada se prilikom izvršenja operacije, kada se kraći vektor isprazni, komponente kraćeg vektora ***ciklično ponavljaju***, od početka prema kraju.

Primer. Ilustruje sabiranje dva vektora različite dužine:

```
> u = c(10, 20, 30)
> v = c(1, 2, 3, 4, 5, 6, 7, 8, 9)
> u + v
[1] 11 22 33 14 25 36 17 28 39
```

Indeksi kod vektora

Komponentama vektora se pristupa korišćenjem operatora srednjih zagrada **[ ]**, u okviru kojih se smešta brojčana vrednost, tzv. ***indeks***. Indeks prvog člana niza je jedan (a ne nula, kao npr. kod programskog jezika C).

Primer. Ilustruje kako se pristupa trećem članu vektora:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[3]
[1] "cc"
```

Za razliku od drugih programskih jezika, indeks vektora ne mora biti jedan broj, već može biti sekvenca brojeva ili interval. Stoga, rezultat pristupa nizu preko indeksa nije jedna vrednost, već više vrednosti tj. vektor komponenti originalnog niza, tzv. ***isečak***.

Primer. U prethodnom primeru, dobijeni rezultat nije jedna vrednost „cc“, već isečak tj. vektor dužine jedan, čiji je prvi član niska „cc“.

Indeksi kod vektora (2)

Ako je ***indeks negativan***, to dovodi do uklanjanja iz rezultujućeg isečka onih elemenata vektora čija je pozicija jednaka apsolutnoj vrednosti negativnog indeksa.

Primer. Ilustruje kako se dobija rezulat sa uklonjenim trećim članom vektora:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[-3]
[1] "aa" "bb" "dd" "ee"
```

Pokušaj da se pri isecanju pristupa komponenti vektora preko indeksa koji je van opsega dovodi do rezulatata NA (not avaliable – nije dostupno).

Primer. Ilustruje šta se dobija kada se pokuša isecati vektor preko indeksa koji je van opsega:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[10]
[1] NA
```

Vektori celih brojeva kao indeksi

Za indeksiranje vektora prilikom isecanja vektora se često koristi ***vektor celih brojeva***.

Primer. Ilustruje kako se određuje isečak vektora koji sadrži njegov drugi i treći član:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[c(2, 3)]
[1] "bb" "cc"
```

U vektoru koji služi za indeksiranje je dozvoljeno da se neke njegove vrednosti ***ponavljaju*** – u tom slučaju bivaju ponovljene vrednosti u rezultujućem vektoru isečku.

Primer. Ilustruje isecanje gde se indeksiranje vrši pomoću vektora kod koga se pojedini članovi ponavljaju:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[c(2, 3, 3)]
[1] "bb" "cc" "cc"
```

Vektori celih brojeva kao indeksi (2)

Članovi vektora koji služi za indeksiranje ***ne moraju biti uređeni***. U rezultujućem vektoru-isečku će članovi biti poređani onim redom kojim su poređani njihovi indeksi u vektoru koji služi za indeksiranje.

Primer. Ilustruje isecanje pomoću vektora za indeksiranje radi izdvajanja prva tri člana, pri čemu će biti obrnut redosled prvog i drugog člana:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[c(2, 1, 3)]
[1] "bb" "aa" "cc"
```

Vektor koji služi za indeksiranje može biti predstavljen kao ***interval***.
Za predstavljanje intervala koristi se operator dvotačka **:** .
Ovo može biti korisno kada se određuje isečak velikog vektora.

Primer. Ilustruje isecanje u kom se indeksira korišćenjem intervala:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[2:4]
[1] "bb" "cc" "dd"
```

Vektori celih brojeva kao indeksi (3)

Primer. Više informacija o operatoru dvotačka može se naći u dokumentaciji sistema R:

```
> help(":")
```

Vektori logičkih vrednosti kao indeksi

Za isecanje vektora se, u svhu indeksiranja, može koristiti i ***vektor logičkih vrednosti***, koji ima istu dužinu kao i vektor-argument (iz kog se vrši isecanje). Ako je član vektora za indeksiranje TRUE, tada se prilikom isecanja odgovarajući član vektora- argumenta prepisuje u rezultujući isečak. Ako je dati član vektora za indeksiranje FALSE, onda nema prepisivanja.

Primer. Ilustruje kako se određuje isečak vektora s tako da ne budu prepisani drugi i četvri član vektora-argumenta:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> L = c(FALSE, TRUE, FALSE, TRUE, FALSE)
> s[L]
[1] "bb" "dd"
```

Primer. Zadatak iz prethodnog primera se može realizovati korišćenjem kraćeg zapisa:

```
> s = c("aa", "bb", "cc", "dd", "ee")
> s[c(FALSE, TRUE, FALSE, TRUE, FALSE)]
[1] "bb" "dd"
```

Imenovanje komponenti vektora

Komponentama vektora se mogu dodeliti ***imena***.

Primer. Neka je dat vektor niski koji se sastoji od dve komponente (dva člana):

```
> v = c("Mary", "Sue")
> v
[1] "Mary" "Sue"
```

Sada se mogu imenovati komponente - prva komponenta vektora v dobija naziv First, a druga komponenta dobija naziv Last:

```
> names(v) = c("First", "Last")
```

Prikaz vektora v je posle izvršenja naredbe imenovanja drugačiji nego što je ranije bio:

```
> v
 First   Last
"Mary"  "Sue"
```

Sada se može pristupiti komponenti vektora preko imena te komponente:

```
> v["First"]
[1] "Mary"
```

Imena komponenti se mogu koristiti i prilikom indekstiranja vektora:

```
> v[c("Last", "First")]
  Last  First
 "Sue" "Mary"
```

Matrice

Matrice

Matrica je kolekcija podataka koji su organizovani u obliku dvodimenzionalne pravougaone tablice.

Elementi u jednosj horizontali se zovu ***vrste matrice***, a elementi u jednoj vertikali se zovu ***kolone matrice***.

Primer. Matrica sa dve vrste i tri kolone:

$$A = \begin{bmatrix} 2 & 4 & 3 \\ 1 & 5 & 7 \end{bmatrix}$$

Matrica se u R-u kreira korišćenjem funkcije **matrix**.

Primer. Matrica iz prethodnog primera bi se mogla formirati na sledeći način:

```
> A = matrix(
+   c(2, 4, 3, 1, 5, 7), # the data elements
+   nrow=2,              # number of rows
+   ncol=3,              # number of columns
+   byrow = TRUE)        # fill matrix by rows
```

Matrice (2)

Primer. Prikaz matrice iz prethodnog primera se realizuje na isti način kao prikaz bilo koje druge promenljive:

```
> A                     # print the matrix
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7
```

Isecanje tj. pristup elementima matrice se postiže korišćenjem operatora **[ ]**:

Primer. Elemenat matrice A u m-toj vrsti i n-toj koloni dobija se izrazom A[m,n]:

```
> A[2, 3]       # element at 2nd row, 3rd column
[1] 7
```

Cela m-ta vrsta matrice A se dobija izrazom A[m,]:

```
> A[2, ]        # the 2nd row
[1] 1 5 7
```

Cela n-ta kolona matrice A se dobija izrazom A[,n]:

```
> A[ ,3]        # the 3rd column
[1] 3 7
```

Matrice (3)

Ako ima potrebe, moguće je ***istovremeno isećii*** više od jedne vrste ili kolone matrice.

Primer. Jednom naredbom se iseca matrica koja sadrži prvu i treću kolonu originalne matrice:

```
> A                     # print the matrix
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7
> A[ ,c(1,3)]  # the 1st and 3rd columns
     [,1] [,2]
[1,]    2    3
[2,]    1    7
```

Imenovanje vrsta i kolona matrica

Ako se vrstama i kolonama matrice ***dodele imena***, onda se prilikom isecanja za indeksiranja elemenata matrice mogu umesto brojeva koristiti imena.

Primer. Dodela imena vrstama i kolonama matrice A se realizuje na sledeći način:

```
> dimnames(A) = list(
+   c("row1", "row2"),         # row names
+   c("col1", "col2", "col3")) # column names
```

Prikaz matrice A je posle izvršenja naredbe imenovanja drugačiji:

```
> A                 # print A
     col1 col2 col3
row1    2    4    3
row2    1    5    7
```

Sada se isecanje, tj. pristup elemantima matrice može realizovati korišćenjem imena vrsta i/ili kolona:

```
> A["row2", "col3"] # element at 2nd row, 3rd column
[1] 7
```

Operacije sa matricama

Postoje različiti načini za ***kreiranje matrice*** u R-u. Ako se matrica kreira direktno od elelemata sa podacima, tada se podrazumeva da ti podaci ispunjavaju matricu kolona po kolona, od početka prema kraju.

Primer. Po izvršenju sledeće nardbe za kreiranje matrice i dodelu vrednosti promenljivoj B:

```
> B = matrix(
+   c(2, 4, 3, 1, 5, 7),
+   nrow=3,
+   ncol=2)
```

Promenljiva B će referisati na matricu, dimenzije 3x2, koja ima sledeći oblik:

```
> B             # B has 3 rows and 2 columns
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7
```

Operacije sa matricama (2)

Korišćenjem metoda **t**, od date matrice se može dobiti ***transponovana matrica***, kod koje su vrste i kolone originalne matrice zamenile mesta.

Primer. Ilustruje kako se vrši transponovanje matrice B, dimenzije 3x2, koja ima sledeći oblik:

```
> B             # B has 3 rows and 2 columns
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7

> t(B)          # transpose of B
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7
```

Operacije sa matricama (3)

Kolone dve matrice koje imaju isti broj vrsta se mogu kombinovati u veću matricu. To se postiže korišćenjem **cbind** funkcije.

Primer. Ilustruje kako se vrši kombinovanje matrica B, dimenzije 3x2 i matrice C, dimenzije 3x1:

```
> B = matrix(
+   c(2, 4, 3, 1, 5, 7),
+   nrow=3,
+   ncol=2)
> B             # B has 3 rows and 2 columns
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7
```

```
> C = matrix(
+   c(7, 4, 2),
+   nrow=3,
+   ncol=1)
> C             # C has 3 rows
     [,1]
[1,]    7
[2,]    4
[3,]    2
```

```
> cbind(B, C)
     [,1] [,2] [,3]
[1,]    2    1    7
[2,]    4    5    4
[3,]    3    7    2
```

Operacije sa matricama (4)

Na sličan način, vrste dve matrice koje imaju isti broj kolonase mogu kombinovati u veću matricu. To se postiže korišćenjem **rbind** funkcije.

Primer. Ilustruje kako se vrši kombinovanje matrica B, dimenzije 3x2 i matrice D, dimenzije 1x2:

```
> B = matrix(
+   c(2, 4, 3, 1, 5, 7),
+   nrow=3,
+   ncol=2)
> B             # B has 3 rows and 2 columns
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7
```

```
> D = matrix(
+   c(6, 2),
+   nrow=1,
+   ncol=2)
> D             # D has 2 columns
     [,1] [,2]
[1,]    6    2
```

```
> rbind(B, D)
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7
[4,]    6    2
```

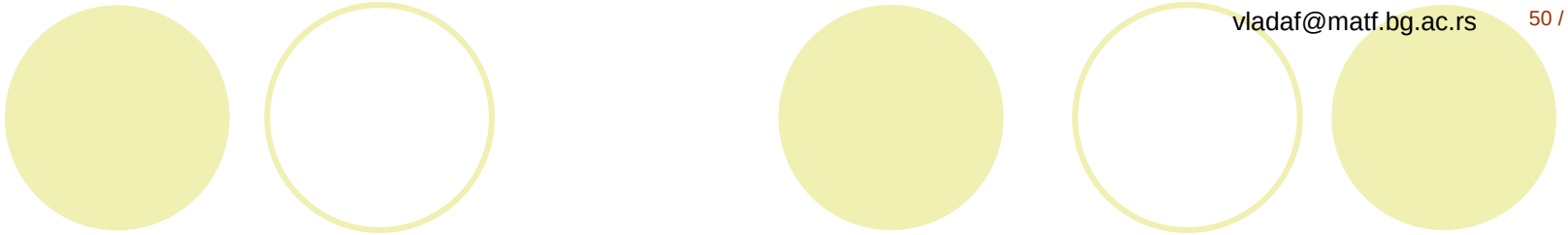
Operacije sa matricama (5)

Korišćenjem funkcije **c** se od date matrice kreira vektor koji sadrži sve elemente matrice. Rezultujući vektor nastaje nadovezivanjem kolona matrice-argumenta.

Primer. Ilustruje kako se od matrica B, dimenzije 3x2 dobija vektor koji sadrži njene članove (komponente vektora su „izvučene“ iz matrice kolona po kolona):

```
> B = matrix(
+   c(2, 4, 3, 1, 5, 7),
+   nrow=3,
+   ncol=2)
> B             # B has 3 rows and 2 columns
     [,1] [,2]
[1,]    2    1
[2,]    4    5
[3,]    3    7
```

```
> c(B)
[1] 2 4 3 1 5 7
```



Liste

Liste

Lista je generički vektor koji sadrži druge objekte.

Lista se može kreirati u R-u korišćenjem **list** funkcije.

Primer. Ilustruje kako se kreira lista od vektora brojeva, vektora sa niskama, vektora logičkih vrednosti i jedne brojčane vrednosti:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> b = c(TRUE, FALSE, TRUE, FALSE, FALSE)
> x = list(n, s, b, 3)   # x contains copies of n, s, b
```

Prilikom kreiranja liste, argumenti funkcije list se kopiraju i novonapravljene kopije argumenata postaju članovi liste.

Liste (2)

Slično kao i vektor i matrica, i lista se može ***isecati***. Isecanje liste se u R-u realizuje korišćenjem operatora srednje zagrade **[ ]**.

Primer. Ilustruje kako se iseca lista i kao rezultat dobija lista koja sadrži drugi element originalne liste:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> b = c(TRUE, FALSE, TRUE, FALSE, FALSE)
> x[2]
[[1]]
[1] "aa" "bb" "cc" "dd" "ee"
```

Liste (3)

Slično kao kod vektora i matrice, i pri isecanju liste se može istovremeno izvršiti isecanje više elemenata liste.

Primer. Ilustruje kako se jednom naredbom iseca lista i kao rezultat dobija lista koja sadrži drugi i četvrti element originalne liste:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> b = c(TRUE, FALSE, TRUE, FALSE, FALSE)
> x = list(n, s, b, 3)   # x contains copies of n, s, b
> x[c(2, 4)]
[[1]]
[1] "aa" "bb" "cc" "dd" "ee"

[[2]]
[1] 3
```

Liste (4)

Elementima liste se može i ***direktno pristupati***. U tu svrhu se korsti operator dvostrukih srednjoh zagrada **[[ ]]**.

Primer. Ilustruje direktan pristup drugom elementu liste – u ovom slučaju rezultat nije lista koja sadrži drugi elemenat liste, već sam drugi elemenat:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> b = c(TRUE, FALSE, TRUE, FALSE, FALSE)
> x = list(n, s, b, 3)   # x contains copies of n, s, b
```

```
> x[[2]]
[1] "aa" "bb" "cc" "dd" "ee"
```

Liste (5)

Korišćenjem operatora direktnog pristupa elementima liste, moguće je realizovati izmenu sadržaja liste.

Primer. Ilustruje izmenu sadržaja liste:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> b = c(TRUE, FALSE, TRUE, FALSE, FALSE)
> x = list(n, s, b, 3)   # x contains copies of n, s, b
> x[[2]][1] = "ta"
> x[[2]]
[1] "ta" "bb" "cc" "dd" "ee"
```

Imenovanje komponenti liste

Kao i kod vektora i matrica, moguće je dodeliti imena komponentama liste i potom ta imena koristiti (umesto brojčanih vrednosti) za referisanje na članove liste.

Primer. Ilustruje kreiranje dvočlane liste sa imenovanim članovima:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v
$bob
[1] 2 3 5

$john
[1] "aa" "bb"
```

Imenovanje komponenti liste (2)

Sada se za isecanje liste mogu koristiti imena komponenti.

Primer. Ilustruje isecanje liste korišćenjem imena komponente:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v["bob"]
$bob
[1] 2 3 5
```

Primer. Ilustruje isecanje liste sa više komponenti, korišćenjem imena komponenti:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v[c("john", "bob")]
$john
[1] "aa" "bb"

$bob
[1] 2 3 5
```

Imenovanje komponenti liste (3)

I za direktan pristup članovima liste se takođe mogu koristiti imena komponenti.

Primer. Ilustruje direktan pristup članu liste preko operatora [[ ]], korišćenjem imena komponente:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v[["bob"]]
[1] 2 3 5
```

Korišćenjem operatora **$**, takođe se može realizovati direktan pristup elemnetu liste preko imena komponente.

Primer. Ilustruje direktan pristup članu liste preko operatora $, korišćenjem imena komponente:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v$bob
[1] 2 3 5
```

Imenovanje komponenti liste (4)

Moguće je da se putanjama za pretragu sistema R pridruže liste, čime se omogućava da se za pristup komponeti liste piše samo ime komponente (tj. da se ne mora pisati ime liste na koju se primenjuje operator $ praćen imenom komponente).

Pridruživanje liste putanjama za pretragu se postiže funkcijom **attach**, a prekid pridruživanja funkcijom **detach**.

Primer. Neka je data lista na sledeći način:

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
```

Pridruživanje liste putanjama za pretragu postiže se sa:

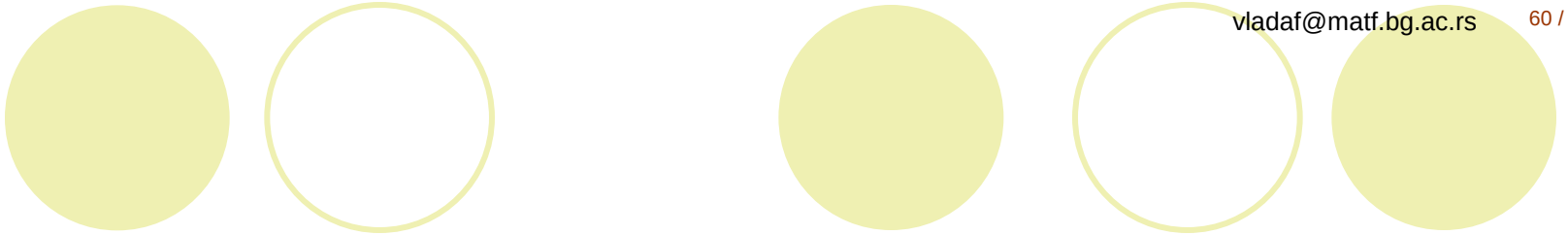
```
> attach(v)
```

Sada se može pristupati imenovanim komponentama liste bez eksplicitnog referisanja na listu:

```
> bob
[1] 2 3 5
```

Po završetku sa korišćenjem komponenti liste, poželjno je prekinuti pridruživanje:

```
> detach(v)
```



Okvir sa podacima

Okvir sa podacima

Okvir sa podacima služi za čuvanje podataka nad kojima se obično vrše statističke analize. Okvir sa podacima je lista vektora iste dužine.

Kreiranje okvira sa podacima na osnovu datih vektora se postiže funkcijom **data.frame**.

Primer. Ilustruje kreiranje okvira sa podacima:

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc")
> b = c(TRUE, FALSE, TRUE)
> df = data.frame(n, s, b)      # df is a data frame
```

Sistem R već sadrži veliki broj okvira sa podacima koji predstavljaju podatke dobijene prilikom raznovrsnih istraživanja.

Korisnik ne mora da kreira okvire sa podacima ugrađene u R, već može odmah da ih koristi.

Još veći broj takvih okvira sa podacima se nalazi u dodatnim R bibliotekama, koje se mogu uključiti po potrebi (a napredniji korisnik može kreirati i svoje okvire sa podacima i svoje biblioteke).

Okvir sa podacima (2)

U primerima koji slede koriistićemo ugrađeni okvir sa podacima mtcars. Ovaj okvir sadrži podatke iz časopisa Motor Trend US magazine iz 1974 i odnosi se na karakteristike raznih modela putničkih automobila.

Kao i kod drugih ugrađenih okviri sa podacima, prva vrsta okvira, nazvana ***zaglavlje***, sadrži imena kolona.
Svaka od sledećih vrsta je ***vrsta sa podacima***: prvi elemenat u vrsti sa podacima je oznaka vrste, a iza oznake slede podaci.
Jedan podatak u vrsti sa podacima je ***ćelija***.

Primer. Prikaz okvira sa podacima mtcars realizuje se jednostavnim navođenjem promenljive mtcars u odzivnom znaku R sistema:

```
> mtcars
                mpg cyl disp  hp drat   wt ...
Mazda RX4      21.0   6  160 110 3.90 2.62 ...
Mazda RX4 Wag  21.0   6  160 110 3.90 2.88 ...
Datsun 710     22.8   4  108  93 3.85 2.32 ...
                ............
```

Okvir sa podacima (3)

Za izvlačenje podatka iz ćelije, koristi se operator srednjih zagrada **[ ]**, unutar kojih se navode koordinate ćelije (pozicija vrste i pozicija kolone razdvojene zarezom).

Primer. Ilustruje izvlačenje podatka iz ćelije okvira sa podacima (u ovom slučaju ćelija u preseku prve vrste i druge kolone):

```
> mtcars[1, 2]
[1] 6
```

Umesto numeričkih koordinata, mogu se koristiti i imena vrsta i kolona.

Primer. Ilustruje izvlačenje podatka iz ćelije okvira sa podacima preko imena vrste i kolone:

```
> mtcars["Mazda RX4", "cyl"]
[1] 6
```

Okvir sa podacima (4)

Broj vrsta sa podacima u okviru sa podacima određuje se korišćenjem **nrow** funkcije.

Primer. Broj vrsta sa podacima u okviru mtcars:

```
> nrow(mtcars)    # number of data rows
[1] 32
```

Broj kolona u okviru sa podacima se određuje funkcijom **ncol**.

Primer. Broj kolona u okviru sa podacima mtcars :

```
> ncol(mtcars)    # number of columns
[1] 11
```

Umesto prikaza celog okvira sa podacima, ponekad je od koristi prikazati samo mali deo radi kratkog pregleda. To se postiže funkcijom **head**.

Primer. Prikaz samo dela okvira sa podacima mtcars :

```
> head(mtcars)
                   mpg cyl disp  hp drat   wt ...
Mazda RX4         21.0   6  160 110 3.90 2.62 ...
                   .............
```

Okvir sa podacima (5)

Primer. Više informacija o okviru sa podacima mtcars može se naći u dokumentaciji sistema R:

```
> help(mtcars)
```

Kolone okvira sa podacima

Vektoru-koloni okvira sa podacima se pristupa pomoću operatora dvostrukih srednjih zagrada **[[ ]]**.

Primer. Određivanje devete kolone-vektora u okviru sa podacima mtcars:

```
> mtcars[[9]]
 [1]  1 1 1 0 0 0 0 0 0 0 ...
```

Primer. Određivanje iste kolone-vektora u okviru sa podacima mtcars, korišćenjem imena kolone i operatora [[ ]]:

```
> mtcars[["am"]]
 [1]  1 1 1 0 0 0 0 0 0 0 ...
```

Kolona-vektor sa podacima može da se dobije i korišćenjem operatora **$**.

Primer. Određivanje iste kolone-vektora u okviru sa podacima mtcars, korišćenjem imena kolone i operatora $:

```
> mtcars$am
 [1]  1 1 1 0 0 0 0 0 0 0 ...
```

Kolone okvira sa podacima (2)

Još jedan način za određivanje vektora-kolone okvira sa podacima je primena operatora jednostrukih srednjih zagrada **[ ]** sa dva argumenta razdvojena zarezom, pri čemu je unutar zagrada prvi argument prazan, a drugi argument definiše o kojoj se koloni okvira radi.

Primer. Određivanje iste kolone-vektora u okviru sa podacima mtcars, korišćenjem imena kolone i operatora [ ]:

```
> mtcars[,"am"]
 [1]  1 1 1 0 0 0 0 0 0 0 0 ...
```

Isecanje po kolonama

Isecanje po kolonama okvira sa podacima se realizuje primenom operatora jednostrukih srednjih zagrada **[ ]** sa jednim argumentom unutar zagrada.
Argument unutar zagrada koji opisuje koje će kolone biti isečene.
Dobijeni rezultat sadrži kolonu sa imenima vrsta sa podacima (prvu kolonu).

Primer. Ilustruje kako se dobija isečak okvira sa podacima koji sadrži prvu kolonu okvira:

```
> mtcars[1]
                  mpg
Mazda RX4        21.0
Mazda RX4 Wag    21.0
Datsun 710       22.8
                  .....
```

Primer. Dobijanje istog isečak okvira sa podacima pomoću naziva kolone:

```
> mtcars["mpg"]
                  mpg
Mazda RX4        21.0
Mazda RX4 Wag    21.0
Datsun 710       22.8
                  .....
```

Isecanje po kolonama (2)

Ako je potrebno izvršiti istovremeno isecanje okvira sa podacima po većem broju kolona, onda se identifikacije kolona „spakuju“ u vektor i na okvir sa podacima se primeni operator jednostrukih srednjih zagrada sa vektorom u kom su „spakovane“ identifikacije kolona kao argumentom.

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži dve kolone mpg i hp:

```
> mtcars[c("mpg", "hp")]
                   mpg  hp
Mazda RX4         21.0 110
Mazda RX4 Wag     21.0 110
Datsun 710        22.8  93
                   .....
```

Isecanje po vrstama

Isecanje po vrstama se realizuje primenom operatora jednostrukih srednjih zagrada **[ ]** sa dva argumenta razdvojena zarezom, pri čemu unutar zagrada prvi argument definiše o kojoj se vrsti/vrstama okvira radi, a drugi argument je prazan.

Definisanje po kojim vrstama će se isecati se može postići navođenjem ***rednih brojeva*** vrsta.

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži 24-tu vrstu:

```
> mtcars[24,]
            mpg cyl disp  hp drat   wt  ...
Camaro Z28 13.3   8  350 245 3.73 3.84  ...
```

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži 3-ću i 24-tu vrstu:

```
> mtcars[c(3, 24),]
            mpg cyl disp  hp drat   wt  ...
Datsun 710 22.8   4  108  93 3.85 2.32  ...
Camaro Z28 13.3   8  350 245 3.73 3.84  ...
```

Isecanje po vrstama (2)

Definisanje po kojim vrstama će se vršiti isecanje se može realizovati navođenjem ***imena*** tih vrsta.

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži vrstu sa podacima o modelu automobila Camaro Z28:

```
> mtcars["Camaro Z28",]
            mpg cyl disp  hp drat   wt  ...
Camaro Z28 13.3   8  350 245 3.73 3.84  ...
```

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži vrstu sa podacima o modelima Datsun 710 i Camaro Z28:

```
> mtcars[c("Datsun 710", "Camaro Z28"),]
            mpg cyl disp  hp drat   wt  ...
Datsun 710 22.8   4  108  93 3.85 2.32  ...
Camaro Z28 13.3   8  350 245 3.73 3.84  ...
```

Isecanje po vrstama (3)

Prilokom isecanja okvira sa podacima po vrstama, često se koristi ***vektor logičkih vrednosti***. Taj vektor se „preklapa“ sa vrstama okvira iz kog se vrši isecanje i prepisuju se one vrste okvira za koje je na odgovarajućoj poziciji u vektoru logičkih vrednosti TRUE, a ostale vrste ne bivaju prepisane.

Primer. Ilustruje kako se dobija isečak okvira sa podacima mtcars koji sadrži informacije o modelima sa automatskim menjačem. Prvo se oformi vektor logičkih vrednosti L, sa informacijom koji model ima automatski menjač, a koji ne:

```
> L = mtcars$am == 0
> L
 [1]   FALSE FALSE FALSE  TRUE ...
```

Potom se tako dobijen vektor logičkih vrednosti iskoristi za isecanje po vrstama okvira sa podacima mtcars:

```
> mtcars[L,]
                   mpg cyl  disp  hp drat    wt  ...
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215  ...
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440  ...
```

Isecanje po vrstama (4)

Primer. Ilustruje kako se iz okvira sa podacima mtcars dobijaju informacije o potrošnji goriva (miles per gallon) za modele automobila sa automatskim menjačem.

Kao u prethodnom primeru, prvo se oformi vektor logičkih vrednosti L, sa informacijom koji model ima automatski menjač, a koji ne:

```
> L = mtcars$am == 0
> L
 [1]   FALSE FALSE FALSE  TRUE ...
```

Potom se tako dobijen vektor logičkih vrednosti iskoristi za isecanje po vrstama okvira sa podacima, a istovremeno se vrši i isecanje po koloni mpg:

```
> mtcars[L,]$mpg
 [1] 21.4 18.7 18.1 14.3 24.4 ...
```

Korišćeni izvori

Deo materijala ove prezentacije je preuzet sa sajta

http://www.r-tutor.com/

Deo materijala je preuzet sa sajta

http://www.e-statistika.rs