R

Vežbe 4

Funkcija seq

- Ova funkcija koristi se za pravljenje sekvence brojeva
- U zavisnosti od argumenata, može dati različite sekvence brojeva

```
#elementi od 1 do 100 sa korakom 10 - celi brojevi
lista1 = seq(1, 100, 10)

#elementi od 1 do 10 sa korakom 0.1 - realni brojevi
lista2 = seq(1, 10, 0.1)

#32 broja u zadatom opsegu
lista3 = seq(1, 10, length.out = 32)

for (i in seq(1, 10))
{
  print(i)
}
```

Data frame

- **Data frame** predstavlja posebnu strukturu podataka u programskom jeziku R koja koristi dosta karakteristika lista i matrica
- Sadrži imenovane kolone koje u se sastoje od podataka istog tipa i imenovane redove, od kojih svaki ima jedinstveno ime

```
#data frame se moze praviti od lista
n = c(2, 3, 5)
s = c("aa", "bb", "cc")
b = c(T, F, T)

df = data.frame(n, s, b)

#pristup elementima data frame-a
df[1,]
df["s"]
```

Ugrađeni data frame

- R sadrži neke ugrađene data frame-ove koji se mogu koristiti
- Jedan od njih je mtcars

```
#stampa ceo sadrzaj ugradjenog data frame-a
mtcars
?mtcars

#prvih 6/ poslednjih 6 elemenata data frame-a
head(mtcars)
tail(mtcars)
```

Zadatak 1 - postavka

- Iz ugrađenog data frame-a **mtcars** izdvojiti u posebne promenljive automobile sa manuelnim i automatskim menjačem (parametar „am“)
- Pronaći i odštampati podatke o automobilu koji troši najmanje goriva (za automobile sa manuelnim menjačem), i podatke o automobilu koji troši najviše goriva (za automobile sa automatskim menjačem). Podatke odštampati na osnovu pređenog puta po galonu (mpg parametar)
- Odštampati podatke o prosečnom pređenom putu po galonu goriva svih automobila, automobila sa manuelnim menjačem, i automobila sa automatskim menjačem

Zadatak 1 - rešenje

```
#najmanja potrosnja manuelnih
manuelni = mtcars[mtcars$am == 1,]
najmanji_manuelni = which(manuelni$mpg == max(manuelni$mpg))
print(manuelni[najmanji_manuelni,])

#najveca potrosnja automatika
automatici = mtcars[mtcars$am == 0,]
najveci_automatik = which(automatici$mpg == min(automatici$mpg))
print(automatici[najveci_automatik,])

#prosecne potrosnje
paste("Prosek predjenog puta svih je:", mean(mtcars$mpg))
paste("Prosek predjenog puta manuelnih je:", mean(manuelni$mpg))
paste("Prosek predjenog puta automatika je:", mean(automatici$mpg))
```

Zadatak 2 - postavka

- Iz fajla „countries.csv“ učitati podatke o BDP-u zemalja i procentualnom učešću industrije i usluga u BDP-u svake zemlje

```
#ucitavanje podataka iz csv fajla
data = read.csv(file="countries.csv", header=T)
```

- Na osnovu procentualnog učešća u BDP-u izračunati vrednost industrije, usluga i ostalih delatnosti za svaku od zemalja
- Odštampati podatke o:
  - zemlji koja ima maksimalni (minimalni) udeo industrije (u procentima)
  - zemlji koja ima maksimalni (minimalni) udeo usluga (u procentima)
  - zemlji koja ima najvredniju industriju (usluge, ostale delatnosti)

Zadatak 2 – rešenje I

```
data = read.csv(file="countries.csv", header=T)

vrednost_industrije = data$gdp * (data$percent_industry/100)
vrednost_usluga = data$gdp * (data$percent_services/100)
ostalo = data$gdp * ((100 - data$percent_industry - data$percent_services)/100)

#zemlja sa maksimalnim udelom industrije u procentima
industrijalizovane_zemlje = which(data$percent_industry == max(data$percent_industry))
print(data[industrijalizovane_zemlje,])

#najmanje industrijalizovane zemlje u procentima
neindustrijalizovane_zemlje = which(data$percent_industry == min(data$percent_industry))
print(data[neindustrijalizovane_zemlje,])

#zemlje sa maksimalnim nivoom usluga u procentima
zemlje_sa_uslugama = which(data$percent_services == max(data$percent_services))
print(data[zemlje_sa_uslugama,])
```

Zadatak 2 – rešenje II

```
#zemlje sa minimalnim nivoom usluga u procentima
zemlje_bez_usluga = which(data$percent_services == min(data$percent_services))
print(data[zemlje_bez_usluga,])

#zemlje sa najvrednijom industrijom
najvrednija_industrija = which(vrednost_industrije == max(vrednost_industrije))
print(data[najvrednija_industrija,])
paste("Vrednost:", max(vrednost_industrije))

#zemlje sa najvrednijim uslugama
najvrednije_usluge = which(vrednost_usluga == max(vrednost_usluga))
print(data[najvrednije_usluge,])
paste("Vrednost:", max(vrednost_usluga))

#zemlje sa najvecom vrednoscu ostalih aktivnosti
najvise_ostalih = which(ostalo == max(ostalo))
print(data[najvise_ostalih,])
paste("Vrednost:", max(ostalo))
```

Zadatak 3 - postavka

- U fajlu „stanovnistvo.csv“ dati su podaci o kretanju stanovništva u Kragujevcu u prethodnih 50-ak godina. Za svaku godinu dati su podaci o broju rođenih, umrlih i ukupnom broju stanovnika.
- Na osnovu ovih podataka nacrtati grafik na kom je prikazan odnos rođenih i umrlih u Kragujevcu, zajedno sa odgovarajućim naslovom i legendom
- Na osnovu dostupnih podataka na zasebnom grafiku prikazati kretanje stanovništva u Kragujevcu

Zadatak 3 - rešenje

```
stanovnistvo = read.csv("stanovnistvo.csv", header=T)

plot(stanovnistvo$rodjeni, type="l", ann = F, axes = F, col = "blue", ylim =
c(min(stanovnistvo$rodjeni, stanovnistvo$umrli),

max(stanovnistvo$rodjeni, stanovnistvo$umrli)))
axis(1, at=1:length(stanovnistvo$rodjeni), labels = stanovnistvo$godina)
axis(2)
box()

lines(stanovnistvo$umrli, type="l", col = "red", lty = 2)
title("Kretanje broja rodjenih i umrlih u Kragujevcu")

legend("topleft", c("rodjeni", "umrli"), cex = 0.8, col = c("blue", "red"), lty =
c(1, 2), title = "Legenda")

plot(stanovnistvo$broj_stanovnika, type = "l", col="green", lty=2, ann = F, axes =
F)
axis(1, at=1:length(stanovnistvo$godina), labels=stanovnistvo$godina)
axis(2)
box()
title("Kretanje broja stanovnika u Kragujevcu")
```

Zadatak 4 - postavka

- Napisati funkciju koja za prosleđen realan broj **x** i ceo broj **n** bez korišćenja petlje računa:

$$\sum_{i=1}^{n} \frac{x^i}{i}$$

- Napisati funkciju koja za prolseđene parametre **a**, **d** i **n** ispisuje prvih **n** članova aritmetičkog niza sa početkom **a** i korakom **d**, i vraća sumu takvog aritmetičkog niza

Zadatak 4 - rešenje

```
fja1 = function(x, n)
{
  lista = c(1:n)
  br = x^lista
  suma = sum(br/lista)
  return (suma)
}

fja2 = function(a, d, n)
{
  niz = seq(from = a, by = d, length.out = n)
  print(niz)
  return (sum(niz))
}
```