

# Predavanje 05 – Programski tok, if, for, seznam

## Programski tok

Skripte, ki jih napišemo, izvajajo ukaze zaporedno iz vrha navzdol. Včasih želimo nekatere ukaze preskočiti, druge pa ponoviti večkrat. To lahko storimo ročno, če uporabljamo bližnjico **Ctrl + Enter**, če pa uporabimo ukaz `source()` se skripta izvede v celoti. Če želimo, da se nekateri deli skripte avtomatsko preskočijo oziroma ponovijo, potrebujemo nekaj ukazov, ki nam bodo to omogočili. Primere, kjer bi to potrebovali, bomo spoznali v nadaljevanju.

## Ukaz if

Ukaz **if** nam omogoča, da se izbrani ukazi izvedejo samo, če velja določen pogoj. V R-ju to napišemo takole.

```
if (<pogoj>) {  
  # ukazi, ki se poženejo, če velja pogoj  
}
```

V zgornjem primeru je `<pogoj>` lahko spremenljivka ali izraz, katerega rezultat je **TRUE** ali **FALSE**, torej tipa **boolean**. Ukazi znotraj zaviti oklepajev `{ }` se izvedejo le v primeru, če je pogoj **TRUE**.

V mapi `data_raw` se nahajajo podatki o spletnih nakupih darilnih in zabavnih izdelkov v Excelovih datotekah. Podatki so bili del znanstvene raziskave Chen, Sain, and Guo (2012), dostopni so na <https://archive.ics.uci.edu/ml/datasets/Online+Retail>. V mapi sta dve datoteki s temi podatki in sicer `online-retail.xlsx`, ki ste jo spoznali že na drugih predavanjih in vsebuje 3 liste, ter `online-retail-large.xlsx`, ki vsebuje veliko listov. Izmed vseh držav je iz originala odstranjena samo Velika Britanija, ker ima nekaj sto tisto izdelkov in bi zato imeli težavi pri odpiranju datoteke v Excelu.

Odprimo najprej prvi list krajše datoteke.

```
library(openxlsx)  
podatki <- read.xlsx("./data_raw/online-retail.xlsx", 1)  
head(podatki)
```

##	InvoiceNo	StockCode	Description	Quantity	UnitPrice
## 1	C538971	22153	ANGEL DECORATION STARS ON DRESS	-48	0.42
## 2	539330	37449	CERAMIC CAKE STAND + HANGING CAKES	8	8.50
## 3	539330	37446	MINI CAKE STAND WITH HANGING CAKES	8	1.45
## 4	539330	22962	JAM JAR WITH PINK LID	12	0.85
## 5	539330	21428	SET3 BOOK BOX GREEN GINGHAM FLOWER	4	4.25
## 6	539330	22113	GREY HEART HOT WATER BOTTLE	4	3.75
##	Country				
## 1	Austria				
## 2	Austria				
## 3	Austria				

```
## 4 Austria
## 5 Austria
## 6 Austria
```

V zgornji tabeli lahko opazimo, da je v podatkih kvantiteta v prvi vrstici negativna, kar je mogoče tudi napaka v podatkih. R takšnih napak seveda avtomatsko ne more odkriti, si pa lahko napišemo ukaze, ki bodo zaznali te nepravilnosti.

Najprej si pogledjmo kako bi ugotovili katere vrednosti so negativne.

```
logicni_vektor <- podatki$Quantity < 0
head(logicni_vektor)
```

```
## [1] TRUE FALSE FALSE FALSE FALSE FALSE
```

Zgornja ukaza preverita vse vrednosti *Quantity* in za vsako vrednost, ki je negativna, vrnete **TRUE**. Nas pa zanima, če je vsaj ena vrednost **TRUE**. To lahko preprosto naredimo z vgrajeno funkcijo **any()**, ki vrne **TRUE**, če je katerakoli vrednost v podanem vektorju **TRUE**. Sorodno deluje **all()**, ki vrne **TRUE**, če so vse podane vrednosti **TRUE**.

Poglejmo si to na primerih.

```
any(FALSE, FALSE, FALSE)
```

```
## [1] FALSE
```

```
all(FALSE, FALSE, FALSE)
```

```
## [1] FALSE
```

```
any(TRUE, TRUE, FALSE)
```

```
## [1] TRUE
```

```
all(TRUE, TRUE, FALSE)
```

```
## [1] FALSE
```

```
any(TRUE, TRUE, TRUE)
```

```
## [1] TRUE
```

```
all(TRUE, TRUE, TRUE)
```

```
## [1] TRUE
```

```
any(podatki$Quantity < 0)
```

```
## [1] TRUE
```

To sta uporabni funkciji za varno delo z **if** stavkom, ker drugače nam bo R zajamral, da zna uporabiti le prvo vrednost vektorja.

```
if (podatki$Quantity < 0) {
  print("Opozorilo: Podatki vsebujejo negativne količine.")
}
```

## Warning in if (podatki\$Quantity < 0) {: the condition has length > 1 and only  
## the first element will be used

## [1] "Opozorilo: Podatki vsebujejo negativne količine."

S funkcijo `any()` pa deluje brez opozorila.

```
if (any(podatki$Quantity < 0)) {
  print("Opozorilo: Podatki vsebujejo negativne količine.")
}
```

## [1] "Opozorilo: Podatki vsebujejo negativne količine."

Preverimo še, če so mogoče negativne tudi cene izdelkov.

```
if (any(podatki$UnitPrice < 0)) {
  print("Opozorilo: Podatki vsebujejo negativne cene.")
}
```

Na izpisu ni ničesar, ker se je ukaz `print()` preskočil. Če želimo obvestilo, da je bilo vse OK lahko uporabimo `if-else` stavek.

```
if (<pogoj>) {
  # ukazi, ki se poženejo, če velja pogoj
} else {
  # ukazi, ki se poženejo, če ne velja pogoj
}
```

V našem primeru bi to uporabili na naslednji način.

```
if (any(podatki$UnitPrice < 0)) {
  print("Opozorilo: Podatki vsebujejo negativne cene.")
} else {
  print("Cene so OK.")
}
```

## [1] "Cene so OK."

## Seznami

V tej skripti smo sedaj vedno prebrali le en list iz Excelove datoteke. Preden nadaljujemo z branjem vseh listov si oglejmo podatkovno strukturo, ki je primerna za hranjenje več tabel hkrati. Struktura je podobna **vektorju** in se imenuje **seznam** oziroma **list**. **List** se od **vektorja** razlikuje predvsem po tem, da lahko hrani različne tipe spremenljivk in tudi druge strukture različnih dolžin.

```
seznam <- list(1, "dva", c(TRUE, FALSE), c(5,6,7), "šest")
seznam
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] "dva"
##
## [[3]]
## [1] TRUE FALSE
##
## [[4]]
## [1] 5 6 7
##
## [[5]]
## [1] "šest"
```

Indeksiranje seznama je podobno kot pri vektorjih le da uporabljamo dvojne oglate oklepaje `[[ ]]`.

```
seznam[[2]]
```

```
## [1] "dva"
```

Vrednosti seznama lahko tudi poimenujemo podobno, kot stolpce `data.frame`.

```
seznam <- list(stevilol = 1, niz1 = "dva", bool1 = c(TRUE, bool = FALSE),
              vektor = c(5,6,7), niz2 = "šest")
seznam[['niz2']]
```

```
## [1] "šest"
```

```
seznam[['bool1']]
```

```
##          bool
## TRUE FALSE
```

Seznamu lahko tudi preprosto dodajamo nove vredosti.

```
seznam[['stevilo2']] <- 5
seznam[[7]] <- 'konec'
seznam
```

```
## $stevilol
## [1] 1
##
## $niz1
## [1] "dva"
##
## $bool1
```

```
##      bool
## TRUE FALSE
##
## $vektor
## [1] 5 6 7
##
## $niz2
## [1] "šest"
##
## $stevilo2
## [1] 5
##
## [[7]]
## [1] "konec"
```

Vsak element seznama lahko shrani tudi celotno tabelo. Zato bomo seznam uporabili, da bo vsak element shranil svoj list Excelove datoteke! Kako bi sedaj naložili vse liste datoteke 'online-retail.xlsx'?

```
podatki <- list() #Ustvarimo prazen seznam
podatki[[1]] <- read.xlsx("./data_raw/online-retail.xlsx", 1)
podatki[[2]] <- read.xlsx("./data_raw/online-retail.xlsx", 2)
podatki[[3]] <- read.xlsx("./data_raw/online-retail.xlsx", 3)
head(podatki[[1]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 C538971 22153 ANGEL DECORATION STARS ON DRESS -48 0.42
## 2 539330 37449 CERAMIC CAKE STAND + HANGING CAKES 8 8.50
## 3 539330 37446 MINI CAKE STAND WITH HANGING CAKES 8 1.45
## 4 539330 22962 JAM JAR WITH PINK LID 12 0.85
## 5 539330 21428 SET3 BOOK BOX GREEN GINGHAM FLOWER 4 4.25
## 6 539330 22113 GREY HEART HOT WATER BOTTLE 4 3.75
## Country
## 1 Austria
## 2 Austria
## 3 Austria
## 4 Austria
## 5 Austria
## 6 Austria
```

```
head(podatki[[2]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 537022 22791 T-LIGHT GLASS FLUTED ANTIQUE 12 1.25
## 2 537022 21287 SCENTED VELVET LOUNGE CANDLE 12 1.25
## 3 537022 79337 BLUE FLOCK GLASS CANDLEHOLDER 6 1.65
## 4 537022 85111 SILVER GLITTER FLOWER VOTIVE HOLDER 12 1.25
## 5 537022 85038 6 CHOCOLATE LOVE HEART T-LIGHTS 6 2.10
## 6 537022 22809 SET OF 6 T-LIGHTS SANTA 6 2.95
## Country
## 1 Italy
## 2 Italy
## 3 Italy
```

```
## 4 Italy
## 5 Italy
## 6 Italy
```

```
head(podatki[[3]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 541932 22699 ROSES REGENCY TEACUP AND SAUCER 24 2.55
## 2 541932 22697 GREEN REGENCY TEACUP AND SAUCER 24 2.55
## 3 541932 22957 SET 3 PAPER VINTAGE CHICK PAPER EGG 24 2.95
## 4 541932 22720 SET OF 3 CAKE TINS PANTRY DESIGN 24 4.25
## 5 541932 72760B VINTAGE CREAM 3 BASKET CAKE STAND 16 8.49
## 6 541932 22763 KEY CABINET MA CAMPAGNE 12 8.50
## Country
## 1 Greece
## 2 Greece
## 3 Greece
## 4 Greece
## 5 Greece
## 6 Greece
```

Super, deluje! Sedaj pa ponovimo enako za datoteko 'online-retail-large.xlsx'. Imamo res voljo pisati enak stavek nekaj 10x? Obstajati mora boljša rešitev!

## Ukaz for

Vidimo, da smo v prejšnjem primeru za branje podatkov praktično isti stavek ponovili trikrat. R nam omogoča, da se ponovnemu pisanju lahko izognemo z uporabo **for** zanke. For zanka v R-ju zgleda takole.

```
for (i in <vektor>) {
  # ukazi, ki se ponavljajo
}
```

Zgornji primer ponovi vse ukaze znotraj zavitih oklepajev {} za vsako vrednost v <vektor>. Poglejmo si preprost primer.

```
for (i in 1:3) {
  print("Ponovitev!")
}
```

```
## [1] "Ponovitev!"
## [1] "Ponovitev!"
## [1] "Ponovitev!"
```

Vsaka ponovitev zanke ni popolnoma enaka. V vsaki ponovitvi je drugačna vrednost spremenljivke *i*. To lahko ponazorimo z naslednjim primerom.

```
for (i in 1:3) {
  print("Ponovitev!")
  print(i)
}
```

```
## [1] "Ponovitev!"
## [1] 1
## [1] "Ponovitev!"
## [1] 2
## [1] "Ponovitev!"
## [1] 3
```

Sedaj pa imamo dovolj znanja, da preprosteje preberemo vse liste Excelovih datotek.

```
podatki <- list()
for (i in 1:3) {
  podatki[[i]] <- read.xlsx("./data_raw/online-retail.xlsx", i)
}
head(podatki[[1]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 C538971 22153 ANGEL DECORATION STARS ON DRESS -48 0.42
## 2 539330 37449 CERAMIC CAKE STAND + HANGING CAKES 8 8.50
## 3 539330 37446 MINI CAKE STAND WITH HANGING CAKES 8 1.45
## 4 539330 22962 JAM JAR WITH PINK LID 12 0.85
## 5 539330 21428 SET3 BOOK BOX GREEN GINGHAM FLOWER 4 4.25
## 6 539330 22113 GREY HEART HOT WATER BOTTLE 4 3.75
## Country
## 1 Austria
## 2 Austria
## 3 Austria
## 4 Austria
## 5 Austria
## 6 Austria
```

```
head(podatki[[3]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 541932 22699 ROSES REGENCY TEACUP AND SAUCER 24 2.55
## 2 541932 22697 GREEN REGENCY TEACUP AND SAUCER 24 2.55
## 3 541932 22957 SET 3 PAPER VINTAGE CHICK PAPER EGG 24 2.95
## 4 541932 22720 SET OF 3 CAKE TINS PANTRY DESIGN 24 4.25
## 5 541932 72760B VINTAGE CREAM 3 BASKET CAKE STAND 16 8.49
## 6 541932 22763 KEY CABINET MA CAMPAGNE 12 8.50
## Country
## 1 Greece
## 2 Greece
## 3 Greece
## 4 Greece
## 5 Greece
## 6 Greece
```

In še večja datoteka. Vidimo, da je vse, kar moramo spremeniti število listov in ime vhodne datoteke.

```
podatki <- list()
for (i in 1:37) {
  podatki[[i]] <- read.xlsx("./data_raw/online-retail-large.xlsx", i)
}
head(podatki[[1]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 536370 22728 ALARM CLOCK BAKELIKE PINK 24 3.75
## 2 536370 22727 ALARM CLOCK BAKELIKE RED 24 3.75
## 3 536370 22726 ALARM CLOCK BAKELIKE GREEN 12 3.75
## 4 536370 21724 PANDA AND BUNNIES STICKER SHEET 12 0.85
## 5 536370 21883 STARS GIFT TAPE 24 0.65
## 6 536370 10002 INFLATABLE POLITICAL GLOBE 48 0.85
## Country
## 1 France
## 2 France
## 3 France
## 4 France
## 5 France
## 6 France
```

```
head(podatki[[37]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 571035 21238 RED RETROSPOT CUP 8 0.85
## 2 571035 21243 PINK POLKADOT PLATE 8 1.69
## 3 571035 23240 SET OF 4 KNICK KNACK TINS DOILY 6 4.15
## 4 571035 23209 LUNCH BAG VINTAGE DOILY 10 1.65
## 5 571035 23201 JUMBO BAG ALPHABET 10 2.08
## 6 571035 23205 CHARLOTTE BAG VINTAGE ALPHABET 10 0.85
## Country
## 1 RSA
## 2 RSA
## 3 RSA
## 4 RSA
## 5 RSA
## 6 RSA
```

Deluje! Vedno potrebujemo le 3-4 vrstice kode, da odpremo poljubno število listov. V tem primeru je sicer nekoliko moteče, da ne moremo preprosto dostopati do na primer podatkov Kanade. Zakaj ne? Ker ne poznamo njenega indeksa. Z uporabo ukaza `getSheetNames()` lahko branje podatkov še nekoliko izboljšamo. Funkcija `getSheetNames()` nam vrne vsa imena listov v Excelovem zvezku. Potem lahko prebermo vsakega, tako da v `<vektor>` vstavimo ta imena, in bo for zanka šla skozi vse njegove vrednosti. Potem lahko shranjujemo v seznam kar z imeni držav, in ne s številskimi indeksi, kar nam bo olajšalo dostop do držav po imenih!

```
podatki <- list()
sheetNames <- getSheetNames("./data_raw/online-retail-large.xlsx")
for (imeLista in sheetNames) {
  podatki[[imeLista]] <- read.xlsx("./data_raw/online-retail-large.xlsx", imeLista)
}
head(podatki[[1]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 536370 22728 ALARM CLOCK BAKELIKE PINK 24 3.75
## 2 536370 22727 ALARM CLOCK BAKELIKE RED 24 3.75
## 3 536370 22726 ALARM CLOCK BAKELIKE GREEN 12 3.75
## 4 536370 21724 PANDA AND BUNNIES STICKER SHEET 12 0.85
```



```
## 5      536370      21883          STARS GIFT TAPE          24      0.65
## 6      536370      10002      INFLATABLE POLITICAL GLOBE      48      0.85
##      Country
## 1      France
## 2      France
## 3      France
## 4      France
## 5      France
## 6      France
```

```
head(podatki[["Canada"]])
```

```
##      InvoiceNo StockCode          Description Quantity UnitPrice
## 1      546533      20886          BOX OF 9 PEBBLE CANDLES      12      1.95
## 2      546533      79030D          TUMBLER, BAROQUE          6      1.65
## 3      546533      21132          SILVER STANDING GNOME          4      4.25
## 4      546533      84879          ASSORTED COLOUR BIRD ORNAMENT      8      1.69
## 5      546533      84755 COLOUR GLASS T-LIGHT HOLDER HANGING      16      0.65
## 6      546533      85116          BLACK CANDELABRA T-LIGHT HOLDER      6      2.10
##      Country
## 1      Canada
## 2      Canada
## 3      Canada
## 4      Canada
## 5      Canada
## 6      Canada
```

Sedaj lahko tabele znotraj lista indeksiramo preko številke ali kar preko imen njenih držav.

Kaj pa če želimo odpreti vse liste do lista z imenom 'Canada'? Uporabimo lahko prejšnjo zanko in jo predčasno prekinemo z besedo **break**!

```
podatki <- list()
sheetNames <- getSheetNames("./data_raw/online-retail-large.xlsx")
for (imeLista in sheetNames) {
  podatki[[imeLista]] <- read.xlsx("./data_raw/online-retail-large.xlsx", imeLista)
  if (imeLista == "Canada") {
    break
  }
}
head(podatki[[1]])
```

```
##      InvoiceNo StockCode          Description Quantity UnitPrice
## 1      536370      22728          ALARM CLOCK BAKELIKE PINK      24      3.75
## 2      536370      22727          ALARM CLOCK BAKELIKE RED      24      3.75
## 3      536370      22726          ALARM CLOCK BAKELIKE GREEN      12      3.75
## 4      536370      21724 PANDA AND BUNNIES STICKER SHEET      12      0.85
## 5      536370      21883          STARS GIFT TAPE          24      0.65
## 6      536370      10002      INFLATABLE POLITICAL GLOBE      48      0.85
##      Country
## 1      France
## 2      France
## 3      France
```

```
## 4 France
## 5 France
## 6 France
```

```
head(podatki[["Canada"]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 546533 20886 BOX OF 9 PEBBLE CANDLES 12 1.95
## 2 546533 79030D TUMBLER, BAROQUE 6 1.65
## 3 546533 21132 SILVER STANDING GNOME 4 4.25
## 4 546533 84879 ASSORTED COLOUR BIRD ORNAMENT 8 1.69
## 5 546533 84755 COLOUR GLASS T-LIGHT HOLDER HANGING 16 0.65
## 6 546533 85116 BLACK CANDELABRA T-LIGHT HOLDER 6 2.10
## Country
## 1 Canada
## 2 Canada
## 3 Canada
## 4 Canada
## 5 Canada
## 6 Canada
```

```
length(podatki)
```

```
## [1] 31
```

Kaj pa če želimo list z imenom 'Canada' le preskočiti? V tem primeru uporabimo privzeto besedo **next**. **Next** prekine izvajanja jedra zanke. Lahko si predstavljate, da se vrne nazaj na prvo vrstico zanke.

```
podatki <- list()
sheetNames <- getSheetNames("./data_raw/online-retail-large.xlsx")
for (imeLista in sheetNames) {
  if (imeLista == "Canada") {
    next
  }
  podatki[[imeLista]] <- read.xlsx("./data_raw/online-retail-large.xlsx", imeLista)
}
head(podatki[[1]])
```

```
## InvoiceNo StockCode Description Quantity UnitPrice
## 1 536370 22728 ALARM CLOCK BAKELIKE PINK 24 3.75
## 2 536370 22727 ALARM CLOCK BAKELIKE RED 24 3.75
## 3 536370 22726 ALARM CLOCK BAKELIKE GREEN 12 3.75
## 4 536370 21724 PANDA AND BUNNIES STICKER SHEET 12 0.85
## 5 536370 21883 STARS GIFT TAPE 24 0.65
## 6 536370 10002 INFLATABLE POLITICAL GLOBE 48 0.85
## Country
## 1 France
## 2 France
## 3 France
## 4 France
## 5 France
## 6 France
```

```
head(podatki[["Canada"]])
```

```
## NULL
```

```
length(podatki)
```

```
## [1] 36
```

Poglejmo si še en primer. V dateki *stroskiDelavnica.xlsx* imamo zapisane stroške gospodninjstva. Vsak list predstavlja svoj mesec. Prvi list ima opozorilo, da se poleg treh stalnih stroškov včasih pojavi tudi strošek za plin. Naložimo vse podatke brez prvega lista, ki se imenuje 'Opozorilo'.

```
podatki <- list()
sheetNames <- getSheetNames("./data_raw/stroskiDelavnica.xlsx")
for (imeLista in sheetNames) {
  if (imeLista == "Opozorilo") {
    next # preskočimo prvi list
  }
  podatki[[imeLista]] <- read.xlsx("./data_raw/stroskiDelavnica.xlsx", imeLista)
}
head(podatki[[1]])
```

```
##           Strosek Vrednost
## 1      Elektriika   121.40
## 2 Komunala+voda    16.00
## 3      Internet    29.03
```

```
length(podatki)
```

```
## [1] 71
```

Izrišimo stroške elektrike po mesecih. Najprej naredimo tabelo, ki izloči samo potrebne podatke.

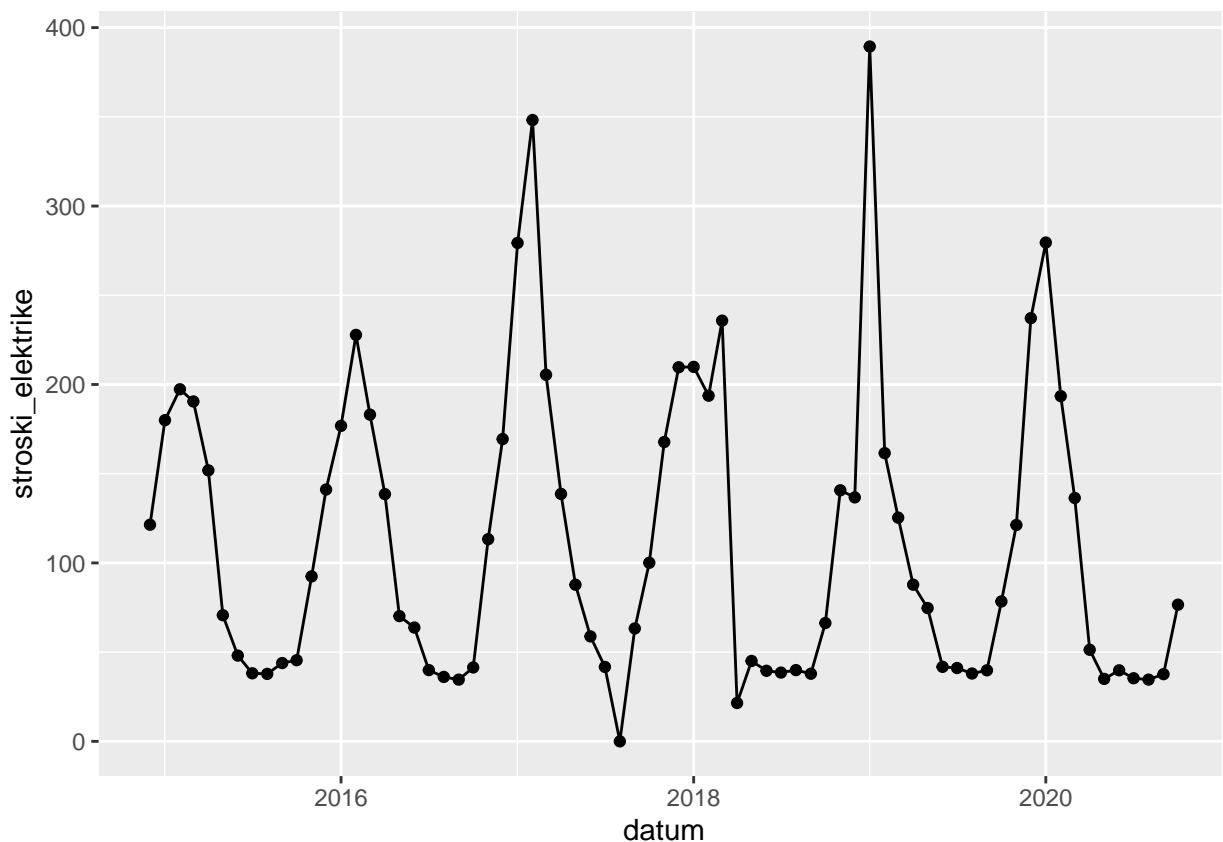
```
mesecni_podatki <- data.frame(mesec = character(0),
                             stroski_elektrike = numeric(0),
                             stringsAsFactors = FALSE)

vrstica <- 1
for (imeLista in sheetNames) {
  if (imeLista == "Opozorilo") {
    next #Preskočimo prvi list
  }
  podatki_temp <- podatki[[imeLista]]
  mesecni_podatki[vrstica, 1] <- imeLista
  mesecni_podatki[vrstica, 2] <-
    podatki_temp[podatki_temp$Strosek == "Elektriika", "Vrednost"]
  vrstica = vrstica + 1
}
# dodamo datume
datumi <- seq.Date(as.Date("2014-12-01"), as.Date("2020-10-01"), by = "months")
mesecni_podatki$datum <- datumi
head(mesecni_podatki)
```

```
##          mesec stroski_elektrike      datum
## 1 DECEMBER 2014          121.40 2014-12-01
## 2  JANUAR 2015          179.99 2015-01-01
## 3 FEBRUAR 2015          197.32 2015-02-01
## 4  MAREC 2015          190.48 2015-03-01
## 5  APRIL 2015          151.87 2015-04-01
## 6   MAJ 2015           70.69 2015-05-01
```

Sedaj pa preprosto z `ggplot()` izrišemo krivuljo.

```
library(ggplot2)
ggplot(mesecni_podatki, aes(x = datum, y = stroski_elektrike, group = 1)) +
  geom_point() + geom_line()
```



## Domača naloga

- a) Preberite podatke v mapi *data\_raw* o ameriških volitvah. Podatke smo pobrali 6. novembra iz: [https://www.kaggle.com/unanimad/us-election-2020?select=president\\_county\\_candidate.csv](https://www.kaggle.com/unanimad/us-election-2020?select=president_county_candidate.csv). Za vsako zvezno državo posebej izračunajte število glasov, ki sta jih dobila glavna kandidata Joe Biden in Donald Trump in jih izpišite na konzolo.

Primer rešitve:

```
## [1] "Zvezna država: Delaware"
## [1] "Joe Biden: 295413 - Donald Trump: 199857"
## [1] ""
## [1] "Zvezna država:"
## [1] "Delaware"
## [1] "Joe Biden:"
## [1] 295413
## [1] "Donald Trump:"
## [1] 199857
## [1] "-----"
## [1] "Zvezna država: District of Columbia"
## [1] "Joe Biden: 258561 - Donald Trump: 14449"
## [1] ""
## [1] "Zvezna država:"
## [1] "District of Columbia"
## [1] "Joe Biden:"
## [1] 258561
## [1] "Donald Trump:"
## [1] 14449
## [1] "-----"
```

- b) Za vsako zvezno državo posebej izračunajte število glasov, ki so jih dobili kandidati in jih shranite v datoteko rezultati.csv. Rezultat zapišite le, če ima kandidat vsaj en glas v zvezni državi. Namig 1: da dobite unikatne vrednosti zveznih držav in kandidatov uporabite funkcijo `unique()`. Namig 2: najlažje bo to rešiti z dvema for zankama, kjer naj bo ena znotraj druge. **Težja!**

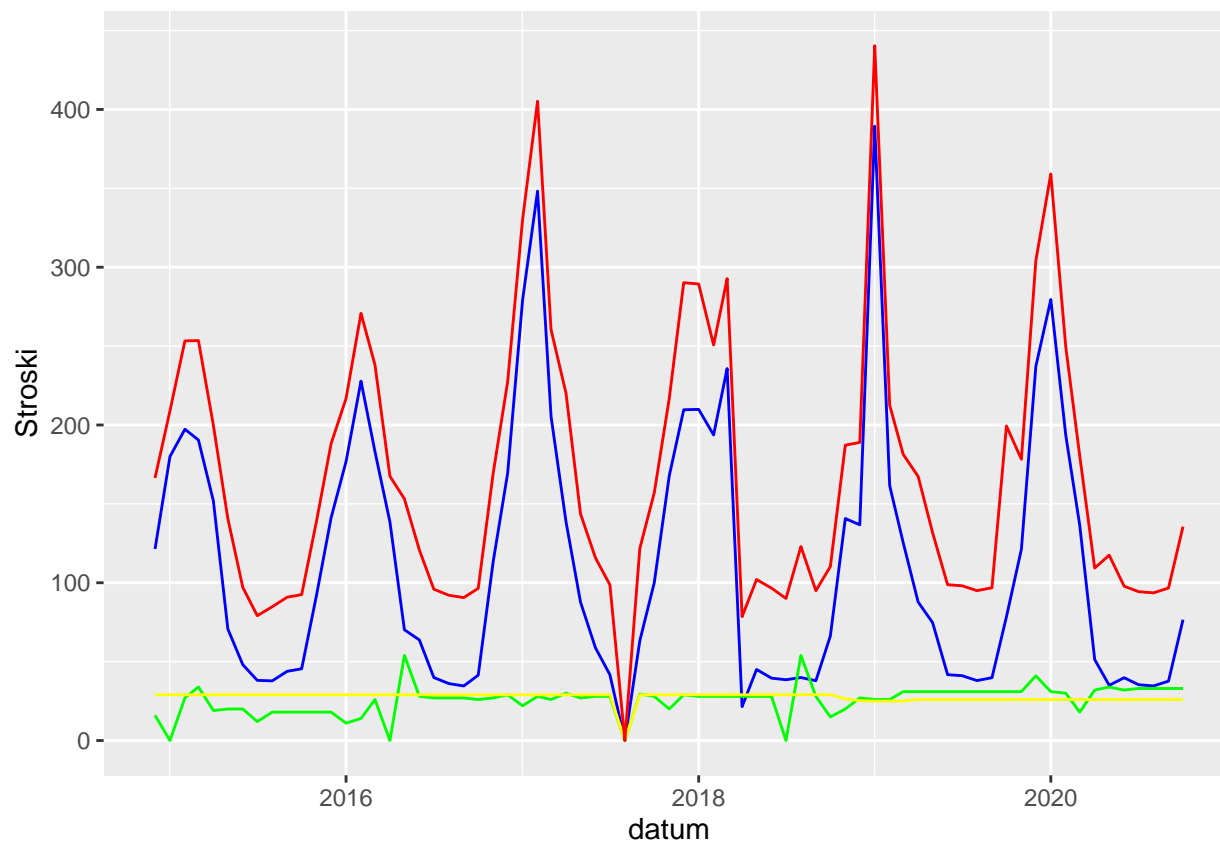
Primer shranjene tabele:

```
##           drzava      kandidat glasovi
## 1      Delaware    Joe Biden  295413
## 2      Delaware    Donald Trump 199857
## 3      Delaware    Jo Jorgensen   4979
## 4      Delaware    Howie Hawkins  2135
## 5 District of Columbia Joe Biden  258561
## 6 District of Columbia Donald Trump 14449
```

- c) Preberite podatke datoteke *stroskiDelavnica.xlsx* in ustvarite tabelo, ki vsebuje poleg stroškov elektrike še stroške za Komunala+voda in Internet ter skupne mesečne stroške.

```
##           mesec stroski_elektrike stroski_komunale stroski_internet
## 1 DECEMBER 2014           121.40             16           29.03
## 2  JANUAR 2015           179.99              0           28.99
## 3 FEBRUAR 2015           197.32             27           28.99
## 4   MAREC 2015           190.48             34           29.02
## 5   APRIL 2015           151.87             19           28.96
## 6    MAJ 2015            70.69             20           29.04
##   stroski_skupaj      datum
## 1      166.43 2014-12-01
## 2      208.98 2015-01-01
## 3      253.31 2015-02-01
## 4      253.50 2015-03-01
## 5      199.83 2015-04-01
## 6      140.63 2015-05-01
```

d) Poskušajte vse štiri stroške tudi izrisati na en graf. Če ne gre, pa naredite štiri posamezne grafe.



#Reference

Chen, Daqing, Sai Laing Sain, and Kun Guo. 2012. "Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining." *Journal of Database Marketing & Customer Strategy Management* 19 (3): 197–208.