# Programski jezici

http://www.programskijezici.matf.bg.ac.rs/

**Univerzitet u Beogradu**
**Matematički fakultet**

# Dizajn programskih jezika

## Materijali za vežbe

**Nastavnik: Milena Vujošević Janičić**
**Asistent: Marjana Šolajić**

**Beograd**
**2018.**

Priprema materijala:
*dr Milena Vujošević Janičić*, docent na Matematičkom fakultetu u Beogradu
*Marjana Šolajić*, asistent na Matematičkom fakultetu u Beogradu
*Branislava Živković*
*Nemanja Mićović*, asistent na Matematičkom fakultetu u Beogradu
*Milica Selaković*, asistent na Matematičkom fakultetu u Beogradu

# Sadržaj


**1 Skript programiranje** **3**
- 1.1 Uvod, kolekcije, matematičke funkcije . . . 3
  - 1.1.1 Uvodni primeri . . . 3
  - 1.1.2 Zadaci za samostalni rad sa rešenjima . . . 4
  - 1.1.3 Zadaci za vežbu . . . 5
- 1.2 Datoteke, niske, JSON format, datum . . . 8
  - 1.2.1 Uvodni primeri . . . 8
  - 1.2.2 Zadaci za samostalni rad sa rešenjima . . . 8
  - 1.2.3 Zadaci za vežbu . . . 9
- 1.3 Argumenti komandne linije, sortiranje, obilazak direktorijuma . . . 11
  - 1.3.1 Uvodni primeri . . . 11
  - 1.3.2 Zadaci za samostalni rad sa rešenjima . . . 12
  - 1.3.3 Zadaci za vežbu . . . 13

**2 Programiranje ograničenja** **15**
- 2.1 Programiranje ograničenja . . . 15
  - 2.1.1 Uvodni primeri . . . 15
  - 2.1.2 Zadaci za samostalni rad sa rešenjima . . . 15
  - 2.1.3 Zadaci za vežbu . . . 16

**3 Komponentno programiranje** **19**
- 3.1 Uvod . . . 19
- 3.2 Zadaci za samostalni rad sa rešenjima . . . 19

**4 Konkurentno programiranje** **21**
- 4.1 Uvod, kreiranje niti, prioritet, sinhronizacija. . . . 21
  - 4.1.1 Uvodni primeri . . . 21
  - 4.1.2 Zadaci za samostalni rad sa rešenjima . . . 21
  - 4.1.3 Zadaci za vežbu . . . 22
- 4.2 Zaključavanje, java katanci i uslovni redovi čekanja. . . . 23
  - 4.2.1 Uvodni primeri . . . 23
  - 4.2.2 Zadaci za samostalni rad sa rešenjima . . . 23
- 4.3 Grafički korisnički interfejs . . . 23
  - 4.3.1 Uvodni primer - železnička stanica . . . 23
  - 4.3.2 Zadaci za samostalni rad sa rešenjima . . . 24
  - 4.3.3 Zadaci za vežbu . . . 24

**5 Generičko programiranje** **27**
- 5.1 Osnove programskog jezika C++, šabloni funkcija . . . 27
  - 5.1.1 Uvodni primeri . . . 27
  - 5.1.2 Zadaci za vežbu . . . 28
- 5.2 Šabloni klasa . . . 28
  - 5.2.1 Uvodni primeri . . . 28
  - 5.2.2 Zadaci za vežbu . . . 29
- 5.3 STL . . . 29
  - 5.3.1 Uvodni primeri . . . 29
  - 5.3.2 Zadaci za vežbu . . . 31

# 1

# Skript programiranje

Potrebno je imati instaliran Python 2.7 na računaru.
Literatura:

(a) https://www.python.org/

(b) http://www.tutorialspoint.com/python

(c) https://wiki.python.org/moin/

## 1.1 Uvod, kolekcije, matematičke funkcije

### 1.1.1 Uvodni primeri

**Zadatak 1.1** Napisati program koji na standardni izlaz ispisuje poruku *Hello world! :)*.

**Zadatak 1.2** Napisati program koji za uneta dva cela broja i nisku ispisuje najpre unete vrednosti, a zatim i zbir brojeva, njihovu razliku, proizvod i količnik.

**Zadatak 1.3** Ako je prvi dan u mesecu ponedeljak napisati funkciju `radni_dan(dan)` koja kao argument dobija dan u mesecu i vraća tačno ako je dan radni dan. Napisati program koji testira ovu funkciju, korisnik sa standardnog ulaza u petlji unosi deset dana i dobija o poruku o tome da li su uneti dani radni ili ne.

**Zadatak 1.4** Napisati program koji na standardni izlaz ispisuje vrednost $6!$, $\log_5 125$ i pseudo slučajan broj iz opsega $[0, 1)$

**Zadatak 1.5** Napisati program koji imitira rad bafera. Maksimalni broj elemenata u baferu je 5. Korisnik sa standardnog ulaza unosi podatke do unosa reči *quit*. Program ih smešta u bafer, posto se bafer napuni unosi se ispisuju na standarni izlaz i bafer se prazni.

**Zadatak 1.6** Korisnik sa standardnog ulaza unosi ceo broj $n$, a potom ciklično pomeren rastuće sortiran niz (pr. 56781234) koji ima $n$ elemenata. Napisati program koji na standarni izlaz ispisuje sortiran niz bez ponavljanja elementa.

**Zadatak 1.7** Napisati funkciju `max_list(lista)` koja vraća najveći element u listi listi. Napisati program koji testira ovu funkciju.

**Zadatak 1.8** Napisati program za rad sa stekom.

- Definisati stek koji sadrži elemente $9, 8, 7$
- Dodati na stek elemente 6 i 5
- Skinuti sa steka element i ispisati ga na standardni izlaz

**Zadatak 1.9** Napisati program koji za uneti prirodan broj $n$ ispisuje vrednosti funkcije $x^2$ u celobrojnim tačkama u intervalu $[0, n]$. Zadatak rešiti korišćenjem mape.

**Zadatak 1.10** Sa standardnog ulaza se unose reči do reči *quit*. Napisati program koji ispisuje unete reči eliminišući duplikate.

**Zadatak 1.11** Napisati funkciju `min_torka(lista)` koja vraća najmanji element u torci torki. Napisati program koji ovu funkciju testira.

### 1.1.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 1.12** **Pogodi broj** Napisati program koji implementira igricu "Pogodi broj". Na početku igre računar zamišlja jedan slučajan broj u intervalu [0,100]. Nakon toga igrač unosi svoje ime i započinje igru. Igrač unosi jedan po jedan broj sve dok ne pogodi koji broj je računar zamislio. Svaki put kada igrač unese broj, u zavisnosti od toga da li je broj koji je unet veći ili manji od zamišljenog broja ispisuje se odgovarajuća poruka. Igra se završava u trenutku kada igrač pogodio zamišljen broj.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
POKRETANJE: pogodi_broj
 -------- IGRA: Pogodi broj ---------
 Unesite Vase ime:
 Milica
 Zdravo Milica. :)
 Zamislio sam neki broj od 1 do 100. Da li mozes da pogodis koji je to broj?
 Unesi broj
 50
 Broj koji sam zamislio je MANJI od 50
 Unesi broj
 25
 Broj koji sam zamislio je VECI od 25
 Unesi broj
 30
 Broj koji sam zamislio je MANJI od 30
 Unesi broj
 27
 "BRAVO!!! Pogodio si! Zamislio sam 27. Bilo je lepo igrati se sa tobom. :)
```

**Zadatak 1.13** **Aproksimacija broja PI metodom Monte Karlo** Napisati program koji aproksimira broj PI koriscenjem metode Monte Karlo. Sa standardnog ulaza unosi se broj N. Nakon toga N puta se bira tačka na slučajan način tako da su obe koordinate tačke iz intervala [0,1]. Broj PI se računa po sledecoj formuli:

$$PI = 4 * A/B$$

- $A$ je broj slučajno izabranih tačaka koje pripadaju krugu poluprečnika 0.5, sa centrom u tački $(0.5, 0.5)$
- $B$ je broj slučajno izabranih tačaka koje pripadaju kvadratu čija temena su tačke $(0, 0), (0, 1), (1, 1), (1, 0)$.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
POKRETANJE: aproksimacija_PI
 Izracunavanje broja PI metodom Monte Karlo
 Unesite broj iteracija:
 5
 Tacka:
 (0.14186247318019474, 0.15644650897353152)
 Tacka:
 (0.8910898038304426, 0.2200563958299553)
 Tacka:
 (0.641604107090444, 0.03712366524007682)
 Tacka:
 (0.4893727376942526, 0.17230005349431587)
 Tacka:
 (0.6199558112390107, 0.32122922953511124)
 Tacka:
 (0.5821041171248978, 0.025052299437462566)
 Broj PI aproksimiran metodom Monte Karlo: 4.0
```

**Zadatak 1.14** **X-O** Napisati program koji implementira igricu X-O sa dva igrača.

*Primer 1*

```
POKRETANJE: XO
INTERAKCIJA SA PROGRAMOM:
 IGRA: X-O pocinje
 Unesite ime prvog igraca:
 Ana
 Zdravo Ana
 Unesite ime drugog igraca:
 Petar
 Zdravo Petar!
 Igrac ('Ana', 'X') igra prvi.
 X : ('Ana', 'X')
 O : ('Petar', 'O')
 Zapocnimo igru
 TABLA
  1 2 3
  --- --- ---
 1 | - | - | - |
  --- --- ---
 2 | - | - | - |
  --- --- ---
 3 | - | - | - |
  --- --- ---
 Ana unesite koordinate polja koje
 zelite da popunite u posebnim linijama:
 Unesite vrstu:
 1
 Unesite kolonu:
 1
 TABLA
  1 2 3
  --- --- ---
 1 | X | - | - |
  --- --- ---
 2 | - | - | - |
  --- --- ---
 3 | - | - | - |
  --- --- ---
 Petar unesite koordinate polja koje
 zelite da popunite u posebnim linijama:
 Unesite vrstu:
 1
 Unesite kolonu:
 2
 TABLA
  1 2 3
  --- --- ---
 1 | X | O | - |
  --- --- ---
 2 | - | - | - |
  --- --- ---
 3 | - | - | - |
  --- --- ---
 Ana unesite koordinate polja koje
 zelite da popunite u posebnim linijama:
 Unesite vrstu:
 2
 Unesite kolonu:
 2
 TABLA
  1 2 3
  --- --- ---
 1 | X | O | - |
  --- --- ---
 2 | - | X | - |
  --- --- ---
 3 | - | - | - |
  --- --- ---
 Petar unesite koordinate polja koje
 zelite da popunite u posebnim linijama:
 Unesite vrstu:
 2
 Unesite kolonu:
 3
 TABLA
  1 2 3
  --- --- ---
 1 | X | O | - |
  --- --- ---
 2 | - | X | O |
  --- --- ---
 3 | - | - | - |
  --- --- ---
 Ana unesite koordinate polja koje
 zelite da popunite u posebnim linijama:
 Unesite vrstu:
 3
 Unesite kolonu:
 3
 TABLA
  1 2 3
  --- --- ---
 1 | X | O | - |
  --- --- ---
 2 | - | X | O |
  --- --- ---
 3 | - | - | X |
  --- --- ---
 BRAVO!!!!!!! Igrac Ana je pobedio!
```

### 1.1.3 Zadaci za vežbu

**Zadatak 1.15** **Ajnc** Napisati program koji implementira igricu Ajnc sa jednim igračem. Igra se sa špilom od 52 karte. Na početku igrač unosi svoje ime nakon čega računar deli dve karte igraču i dve karte sebi. U svakoj sledećoj iteraciji računar deli po jednu kartu igraču i sebi. Cilj igre je sakupiti karte koje u zbiru imaju 21 poen. Karte sa brojevima nose onoliko bodova koliki je broj, dok žandar, dama, kralj nose 10 bodova. Karta As može da nosi 1 ili 10 bodova, u zavisnosti od toga kako igraču odgovara. Igrač koji sakupi 21 je pobedio. Ukoliko igrač premaši 21 bod, pobednik je njegov protivnik. Detaljan opis igre može se naći na narednoj adresi: https://en.wikipedia.org/wiki/Blackjack

*Primer 1*

```
Pokretanje: ajnc
Interakcija sa programom:
 -------- IGRA: Ajnc ---------
 Unesite Vase ime:
 Pavle
 Zdravo Pavle. :)
 Igra pocinje
 Vase karte su:
 1 Herc 5 karo
 Hit ili stand?[H/S]
 H
 Vase karte su:
 1 Herc 5 karo 5 tref
 Cestitam!!! Pobedio si!
 Bilo je lepo igrati se sa tobom. :)
```

*Primer 2*

```
Pokretanje: ajnc
Interakcija sa programom:
 -------- IGRA: Ajnc ---------
 Unesite Vase ime:
 Pavle
 Zdravo Pavle. :)
 Igra pocinje
 Vase karte su:
 Q Tref 7 karo
 Hit ili stand?[H/S]
 H
 Vase karte su:
 Q Tref 7 karo K Herc
 Zao mi je, izgubio si!:(
 Bilo je lepo igrati se sa tobom. :)
```

**Zadatak 1.16** **4 u liniji** Napisati program koji implementira igricu 4 u nizu sa dva igrača. Tabla za igru je dimenzije 8x8. Igrači na početku unose svoja imena, nakon čega računar nasumično dodeljuje crvenu i žutu boju igračima. Igrač sa crvenom bojom igra prvi i bira kolonu u koju ce da spusti svoju lopticu. Cilj igre je da se sakupe 4 loptice iste boje u liniji. Prvi igrač koji sakupi 4 loptice u liniji je pobedio. Detaljan opis igre može se naći na narednoj adresi: https://en.wikipedia.org/wiki/Connect_Four

*Primer 1*

```
Pokretanje: cetri_u_nizu
Interakcija sa programom:
 IGRA: Cetiri u nizu pocinje
 Unesite ime prvog igraca:
 Ana
 Zdravo Ana
 Unesite ime drugog igraca:
 Petar
 Zdravo Petar!
 Igrac ('Ana', 'C') igra prvi.
 C : ('Ana', 'C')
 Z : ('Petar', 'Z')
 Zapocnimo igru
 TABLA
  1 2 3 4 5 6 7 8
  --- --- --- --- --- --- --- ---
 1 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 2 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 3 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 4 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 5 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 6 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 7 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 8 | - | - | - | - | - | - | - | - |
 Ana unesite koordinate polja
 koje zelite da popunite
 u posebnim linijama:
 Unesite vrstu:
 1
 Unesite kolonu:
 1
 TABLA
  1 2 3 4 5 6 7 8
  --- --- --- --- --- --- --- ---
 1 | c | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 2 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 3 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 4 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 5 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 6 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 7 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 8 | - | - | - | - | - | - | - | - |
 Petar unesite koordinate polja
 koje zelite da popunite
 u posebnim linijama:
 Unesite vrstu:
 2
 Unesite kolonu:
 1
 TABLA
  1 2 3 4 5 6 7 8
  --- --- --- --- --- --- --- ---
 1 | c | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 2 | z | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 3 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 4 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 5 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 6 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 7 | - | - | - | - | - | - | - | - |
  --- --- --- --- --- --- --- ---
 8 | - | - | - | - | - | - | - | - |
```

```
Ana unesite koordinate polja
koje zelite da popunite
u posebnim linijama:
Unesite vrstu:
1
Unesite kolonu:
2
TABLA
 1 2 3 4 5 6 7 8
 --- --- --- --- --- --- --- ---
1 | c | c | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
2 | z | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
3 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
4 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
5 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
6 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
7 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
8 | - | - | - | - | - | - | - | - |
Petar unesite koordinate polja
koje zelite da popunite
u posebnim linijama:
Unesite vrstu:
2
Unesite kolonu:
2
TABLA
 1 2 3 4 5 6 7 8
 --- --- --- --- --- --- --- ---
1 | c | c | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
2 | z | z | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
3 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
4 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
5 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
6 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
7 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
8 | - | - | - | - | - | - | - | - |
Ana unesite koordinate polja
koje zelite da popunite
u posebnim linijama:
Unesite vrstu:
1
Unesite kolonu:
3
TABLA
 1 2 3 4 5 6 7 8
 --- --- --- --- --- --- --- ---
1 | c | c | c | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
2 | z | z | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
3 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
4 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
5 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
6 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
7 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
8 | - | - | - | - | - | - | - | - |
```

```
Petar unesite koordinate polja
koje zelite da popunite
u posebnim linijama:
Unesite vrstu:
2
Unesite kolonu:
3
TABLA
 1 2 3 4 5 6 7 8
 --- --- --- --- --- --- --- ---
1 | c | c | c | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
2 | z | z | z | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
3 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
4 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
5 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
6 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
7 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
8 | - | - | - | - | - | - | - | - |
Ana unesite koordinate polja
koje zelite da popunite
u posebnim linijama:
Unesite vrstu:
1
Unesite kolonu:
4
TABLA
 1 2 3 4 5 6 7 8
 --- --- --- --- --- --- --- ---
1 | c | c | c | c | - | - | - | - |
 --- --- --- --- --- --- --- ---
2 | z | z | z | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
3 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
4 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
5 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
6 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
7 | - | - | - | - | - | - | - | - |
 --- --- --- --- --- --- --- ---
8 | - | - | - | - | - | - | - | - |
BRAVO!!!!!!! Igrac Ana je pobedio!
```

## 1.2 Datoteke, niske, JSON format, datum

### 1.2.1 Uvodni primeri

**Zadatak 1.17** Korisnik na standardni ulaz unosi dve niske. Napisati program koji prvo pojavljivanje druge niske u prvoj zamenjuje karakterom $. U slučaju da nema pojavljivanja druge niske u prvoj i da je druga niska kraća ispisuje nadovezane niske sa separatorom -. Ako je druga niska duža od prve program treba da ispiše drugu nisku i njenu dužinu.

**Zadatak 1.18** Napisati program koji ispisuje tekući dan u nedelji, dan, mesec i vreme u formatu *hh:mm:ss*.

**Zadatak 1.19** Napisati program koji ispisuje sadrzaj datoteka *datoteka.txt* na standardni izlaz karakter po karakter.

**Zadatak 1.20** Napisati program koji ispisuje sadrzaj datoteka *datoteka.txt* na standardni izlaz liniju po liniju.

**Zadatak 1.21** Napisati program koji dodaje u datoteku *datoteka.txt* nisku *water* a potom ispisuje njen sadržaj na standardni izlaz.

**Zadatak 1.22** Korisnik na standarni ulaz unosi podatke o imenu, prezimenu i godinama. Program potom kreira JSON objekat *junak*, koji ima podatke *Ime*, *Prezime* i *Godine*, i ispisuje ga na standardni izlaz, a potom i u datoteku *datoteka.txt*.

**Zadatak 1.23** Napisati program koji iz datoteke *datoteka.txt* učitava JSON objekat, a potom na standardni izlaz ispisuje podatke o *imenu*, *prezimenu* i *godininama*.

### 1.2.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 1.24** Napisati program koji sa standardnog ulaza učitava ime datoteke i broj $n$ i računa broj pojavljivanja svakog $n$-grama u datoteci koji su sačinjeni od proizvoljnih karaktera i rezultat upisuje u datoteku `rezultat.json`.

*Primer 1*

```
POKRETANJE: python n-anagram.py
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:
 datoteka.txt
 Unesite n
 2
```

Sadržaj datoteka koje se koriste u primeru 1.24:

Listing 1.1: *datoteka.txt*

```
Ovo je datoteka dat
```

Listing 1.2: *rezultat.json*

```
{
'a ': 1, 'ka': 1, 'ot': 1, 'ek': 1,
' d': 2, ' j': 1, 'da': 2, 'e ': 1,
'o ': 1, 'to': 1, 'at': 2, 'je': 1,
'Ov': 1, 'te': 1, 'vo': 1
}
```

**Zadatak 1.25**
U datoteci `korpa.json` se nalazi spisak kupljenog voća u json formatu:

```
[ { 'ime' : ime_voca, 'kolicina' : broj_kilograma } , ...]
```

U datotekama `maxi_cene.json`, `idea_cene.json`, `shopngo_cene.json` se nalaze cene voća u json formatu:

```
[ { 'ime' : ime_voca, 'cena' : cena_po_kilogramu } , ...]
```

Napisati program koji izračunava ukupan račun korpe u svakoj prodavnici i ispisuje cene na standardni izlaz.

*Primer 1*

```
POKRETANJE: python korpa.py
INTERAKCIJA SA PROGRAMOM:
 Maxi: 631.67 dinara
 Idea: 575.67 dinara
 Shopngo: 674.67 dinara
```

Sadržaj datoteka koje se koriste u primeru 1.25:

Listing 1.3: *korpa.json*

```
[ {"ime" : "jabuke" ,"kolicina": 3.3},
{"ime": "kruske" , "kolicina": 2.1},
{"ime": "grozdje" , "kolicina":  2.6},
```

Listing 1.4: *maksi_cene.json*

```
[ {"ime" :"jabuke", "cena" : 59.9},
{"ime" :"kruske", "cena" : 120},
{"ime" :"grozdje", "cena" : 70},
{"ime" :"narandze", "cena" : 49.9},
{"ime" :"breskve", "cena" : 89.9} ]
```

Listing 1.5: *idea_cene.json*

```
[ {"ime" :"jabuke", "cena" : 39.9},
{"ime" :"kruske", "cena" : 100},
{"ime" :"grozdje", "cena" : 90},
{"ime" :"breskve", "cena" : 59.9} ]
```

Listing 1.6: *shopngo_cene.json*

```
[ {"ime" :"jabuke", "cena" : 69.9},
{"ime" :"kruske", "cena" : 100},
{"ime" :"grozdje", "cena" : 90},
{"ime" :"maline", "cena" : 290},
```

### 1.2.3 Zadaci za vežbu

**Zadatak 1.26** Napisati program koji iz datoteke `ispiti.json` učitava podatke o ispitima i njihovim datumima. Ispisati na standardni izlaz za svaki ispit njegovo ime i status ”Prosao” ukoliko je ispit prosao, odnosno ”Ostalo je jos n dana.”, gde je n broj dana od trenutnog datuma do datuma ispita.

*Primer 1*

```
POKRETANJE: python ispiti.py
INTERAKCIJA SA PROGRAMOM:
 Relacione baze podataka Prosao
 Vestacka inteligencija Prosao
 Linearna algebra i analiticka geometrija Prosao
```

Sadržaj datoteka koje se koriste u primeru 1.26:

Listing 1.7: *ispiti.json*

```
[  {'ime': 'Relacione baze podataka',
'datum': '21.09.2016.'},
{'ime': 'Vestacka inteligencija',
'datum': '17.06.2017.'},
{'ime': 'Linearna algebra i analiticka geometrija',
'datum': '08.02.2017.'} ]
```

**Zadatak 1.27** Napisati program koji izdvaja sve jednolinijske i višelinijske komentare iz .c datoteke čije ime se unosi sa standardnog ulaza, listu jednih i drugih komentara upisuje u datoteku `komentari.json`. Jednolinijski komentari se navode nakon // a višelinijski između /∗ i ∗/.

*Primer 1*

```
Pokretanje: python komentari.py
Interakcija sa programom:
 Unesite ime datoteke:
 program.c
```

Sadržaj datoteka koje se koriste u primeru 1.27:

Listing 1.8: *program.c*

```
#include <stdio.h>

// Primer jednolinijskog komentara

int main(){
/*
Na ovaj nacin ispisujemo tekst
na standardni izlaz koristeci jezik C.
*/
printf("Hello world!");

// Na ovaj nacin se ispisuje novi red
printf("\n");
/*
Ukoliko se funkcija uspesno zavrsila
vracamo 0 kao njen rezultat.
*/
return 0;
}
```

Listing 1.9: *komentari.json*

```
{
  'jednolinijski' : ['Primer jednolinijskog komentara',
                     'Na ovaj nacin se ispisuje novi red'],
  'viselinijski' : ['Na ovaj nacin ispisujemo tekst na standardni
                     izlaz koristeci jezik C.',
                     'Ukoliko se funkcija uspesno zavrsila
                     vracamo 0 kao njen rezultat.']
}
```

**Zadatak 1.28** Napisati program upoređuje dve datoteke čija imena se unose sa standardnog ulaza. Rezultat upoređivanja je datoteka `razlike.json` koja sadrži broj linija iz prve datoteke koje se ne nalaze u drugoj datoteci i obratno. *Napomena* Obratiti pažnju na efikasnost.

*Primer 1*

```
Pokretanje: python razlika.py
Interakcija sa programom:
 Unesite ime datoteke:
 dat1.txt
 Unesite ime datoteke:
 dat2.txt
```

Sadržaj datoteka koje se koriste u primeru 1.28:

Listing 1.10: *dat1.txt*

```
//netacno

same=1;

for(i=0;s1[i]!='\0' && s2[i]!='\0';i++) {
if(s1[i]!=s2[i]) {
same=0;
break;
}
}
return same;
```

Listing 1.11: *dat2.txt*

```
//tacno

for(i=0;s1[i]!='\0' && s2[i]!='\0';i++){

if(s1[i]!=s2[i])
return 0;
}
return s1[i]==s2[i];
```

Listing 1.12: *razlike.json*

```
{
  'dat1.txt' : 7,
  'dat2.txt' : 4
}
```

## 1.3 Argumenti komandne linije, sortiranje, obilazak direktorijuma

### 1.3.1 Uvodni primeri

**Zadatak 1.29** Napisati program koji na standarni izlaz ispisuje argumente komandne linije.

**Zadatak 1.30** Napisati program koji na standrani izlaz ispisuje oznaku za tekući i roditeljski direktorijum, kao i separator koji se koristi za pravljenje putanje.

**Zadatak 1.31** Napisati program koji imitira rad komande *ls*. Program na standardni izlaz ispisuje sadržaj tekućeg direktorijuma.

**Zadatak 1.32** Napisati program koji na standarni izlaz ispisuje sve apsolutne putanje regularnih fajlova koji se nalaze u tekućem direktorijumu.

**Zadatak 1.33** U datoteci *tacke.json* se nalaze podaci o tačkama u sledećem formatu.

Listing 1.13: *tacke.json*

```
[ {"teme":"A" , "koordinate": [10.0, 1.1]},
 {"teme":"B" , "koordinate": [1.0, 15.0]},
 {"teme":"C" , "koordinate": [-1.0, 5.0]} ]
```

Napisati program koji učitava podatke o tačkama iz datoteke *tacke.json* i sortira i po udaljenosti od koordinatnog početka. Na standarni izlaz ispisati podatke pre i posle sortiranja.

### 1.3.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 1.34** Napisati program koji računa odnos kardinalnosti skupova duže i šire za zadati direktorijum. Datoteka pripada skupu duže ukoliko ima više redova od maksimalnog broja karaktera po redu, u suprotnom pripada skupu šire. Sa standardnog ulaza se unosi putanja do direktorijuma. Potrebno je obići sve datoteke u zadatom direktorijumu i njegovim poddirektorijumima (koristiti funkciju os.walk()) i ispisati odnos kardinalnosti skupova duže i šire.

*Primer 1*

```
POKRETANJE: python duze_sire.py
INTERAKCIJA SA PROGRAMOM:
 Unesite putanju do direktorijuma:
 ..
 Kardinalnost skupa duze : Kardinalnost skupa sire
 10 : 15
```

**Zadatak 1.35** Napisati program koji obilazi direktorijume rekurzivno i računa broj datoteka za sve postojeće ekstenzije u tim direktorijumima. Sa standardnog ulaza se unosi putanja do početnog direktorijuma, a rezultat se ispisuje u datoteku `rezultat.json`.

*Primer 1*

```
POKRETANJE: python ekstenzije.py
INTERAKCIJA SA PROGRAMOM:
 Unesite putanju do direktorijuma:
 .
```

Sadržaj datoteka koje se koriste u primeru 1.35:

Listing 1.14: *rezultat.txt*

```
{
 'txt' : 14,
 'py' : 12,
 'c' : 10
}
```

**Zadatak 1.36** U datoteci `radnici.json` nalaze se podaci o radnom vremenu zaposlenih u preduzeću u sledecem formatu:

```
[  { 'ime' : ime\_radnika, 'odmor' : [pocetak, kraj], 'radno_vreme'
: [pocetak, kraj] }, ...]
```

Napisati program koji u zavisnosti od unete opcije poslodavcu ispisuje trenutno dostupne radnike odnosno radnike koji su na odmoru. Moguće opcije su 'd' - trenutno dostupni radnici i 'o' - radnici koji su na odmoru. Radnik je dostupan ukoliko nije na odmoru i trenutno vreme je u okviru njegovog radnog vremena.

2

*Primer 1*

```
POKRETANJE: python odmor.py
INTERAKCIJA SA PROGRAMOM:
 "Unesite opciju koju zelite
 d - dostupni radnici
 o - radnici na odmoru :
 m
 Uneta opcija nije podrzana
```

*Primer 2*

```
POKRETANJE: python odmor.py
INTERAKCIJA SA PROGRAMOM:
 "Unesite opciju koju zelite
 d - dostupni radnici
 o - radnici na odmoru :
 d
 Pera Peric
```

Sadržaj datoteka koje se koriste u primeru 1.36:

Listing 1.15: *radnici.json*

```
[  { 'ime' : 'Pera Peric',
'odmor' : ['21.08.2016.', '31.08.2016.'],
'radno_vreme' : ['08:30', '15:30'] } ]
```

**Zadatak 1.37** Napisati program koji učitava ime datoteke sa standardnog ulaza i na standardni izlaz ispisuje putanje do svih direktorijuma u kojima se nalazi ta datoteka.

*Primer 1*

```
POKRETANJE: python pojavljivanja.py
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:
 1.py
 /home/student/vezbe/cas1/1.py
 /home/student/vezbe/cas7/1.py
 /home/student/vezbe/cas9/1.py
```

### 1.3.3 Zadaci za vežbu

**Zadatak 1.38** Napisati program koji ispisuje na standardni izlaz putanje do lokacija svih Apache virtuelnih hostova na računaru. Smatrati da je neki direktorijum lokacija Apache virtuelnog hosta ukoliko u sebi sadrži `index.html` ili `index.php` datoteku.

*Primer 1*

```
POKRETANJE: python apache.py
INTERAKCIJA SA PROGRAMOM:
 /home/student/PVEB/prviPrimer
 /home/student/licna_strana
 /home/student/PVEB/ispit/jun
```

**Zadatak 1.39** Napisati program koji realizuje autocomplete funkcionalnost. Sa standardnog ulaza korisnik unosi delove reči sve dok ne unese karakter !. Nakon svakog unetog dela reči ispisuju se reči koje počinju tim karakterima. Spisak reči koje program može da predloži se nalazi u datoteci `reci.txt`.

*Primer 1*

```
POKRETANJE: python autocomplete.py
INTERAKCIJA SA PROGRAMOM:
 ma
 mac macka mama maceta madjionicar
 mac
 mac macka maceta
 !
```

Sadržaj datoteka koje se koriste u primeru 1.39:

Listing 1.16: *reci.txt*

```
mac pesma skola macka mama maceta igra madjionicar
```

# 2

# Programiranje ograničenja

Potrebno je imati instaliran Python 2.7 i biblioteku python-constraint. Na Ubuntu 14.04 operativnom sistemu, biblioteka python-constraint se moze instalirati pomocu Pip alata:

```
sudo apt-get -y install python-pip
sudo pip install python-constraint
```

Korisni linkovi i literatura:

http://labix.org/doc/constraint/
https://pypi.python.org/pypi/python-constraint
http://www.hakank.org/constraint_programming_blog/

## 2.1 Programiranje ograničenja

### 2.1.1 Uvodni primeri

**Zadatak 2.1** Modul constraint, osnovne funkcije

### 2.1.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 2.2** Napisati program koji pronalazi trocifren broj `ABC` tako da je količnik `ABC / (A + B + C)` minimalan i `A`, `B` i `C` su različiti brojevi.

**Zadatak 2.3** Dati su novčići od 1, 2, 5, 10, 20 dinara. Napisati program koji pronalazi sve moguće kombinacije tako da zbir svih novčića bude 50.

**Zadatak 2.4** Napisati program koji ređa brojeve u magičan kvadrat. Magičan kvadrat je kvadrat dimenzija 3x3 takav da je suma svih brojeva u svakom redu, svakoj koloni i svakoj dijagonali jednak 15 i svi brojevi različiti. Na primer:

```
4 9 2
3 5 7
8 1 6
```

**Zadatak 2.5** Napisati program koji pronalazi sve vrednosti promenljivih X, Y i Z za koje važi da je $X >= Z$ i $X * 2 + Y * X + Z <= 34$ pri čemu promenljive pripadaju narednim domenima $X \in \{1, 2, ..., 90\}$, $Y \in \{2, 4, 6, ...60\}$ i $Z \in \{1, 4, 9, 16, ..., 100\}$

**Zadatak 2.6** Napisati program koji dodeljuje različite vrednosti različitim karakterima tako da suma bude zadovoljena:

```
  TWO
 +TWO
 ---------
 FOUR
```

**Zadatak 2.7** Napisati program koji pronalazi sve vrednosti promenljivih X, Y, Z i W za koje važi da je $X >= 2 * W$, $3 + Y <= Z$ i $X - 11 * W + Y + 11 * Z <= 100$ pri čemu promenljive pripadaju narednim domenima $X \in \{1, 2, ..., 10\}$, $Y \in \{1, 3, 5, ...51\}$, $Z \in \{10, 20, 30, ..., 100\}$ i $W \in \{1, 8, 27, ..., 1000\}$.

**Zadatak 2.8** Napisati program koji raspoređuje brojeve 1-9 u dve linije koje se seku u jednom broju. Svaka linija sadrži 5 brojeva takvih da je njihova suma u obe linije 25 i brojevi su u rastućem redosledu.

```
1   3
 2 4
  5
 6 8
7   9
```

**Zadatak 2.9** Pekara *Kiflica* proizvodi hleb i kifle. Za mešenje hleba potrebno je 10 minuta, dok je za kiflu potrebno 12 minuta. Vreme potrebno za pečenje ćemo zanemariti. Testo za hleb sadrži 300g brašna, a testo za kiflu sadrži 120g brašna. Zarada koja se ostvari prilikom prodaje jednog hleba je 7 dinara, a prilikom prodaje jedne kifle je 9 dinara. Ukoliko pekara ima 20 radnih sati za mešenje peciva i 20kg brašna, koliko komada hleba i kifli treba da se umesi kako bi se ostvarila maksimalna zarada (pod pretpostavkom da će pekara sve prodati)?

**Zadatak 2.10** Napisati program pronalazi vrednosti A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S (svako slovo predstavlja različit broj) koje su poređane u heksagon na sledeći način:

```
  A,B,C
 D,E,F,G
H,I,J,K,L
 M,N,O,P
  Q,R,S
```

tako da zbir vrednosti duž svake horizontalne i dijagonalne linije bude 38 ( A+B+C = D+E+F+G = ... = Q+R+S = 38, A+D+H = B+E+I+M = ... = L+P+S = 38, C+G+L = B+F+K+P = ... = H+M+Q = 38).

**Zadatak 2.11** Kompanija Start ima 250 zaposlenih radnika. Rukovodstvo kompanije je odlučilo da svojim radnicima obezbedi dodatnu edukaciju. Da bi se radnik obučio programskom jeziku Elixir potrebno je platiti 100 evra po osobi za kurs, ali bi njegovo produktivno znanje ovog programskog jezika donelo 150 projekat/sati mesečno, što bi za kompaniju značilo dobit od 5 evra po projekat/satu. Da bi se radnik obučio programskom jeziku Dart potrebno je platiti 105 evra po osobi za kurs, ali bi njegovo produktivno znanje ovog programskog jezika donelo 170 projekat/sati mesečno, koji bi za kompaniju značili dobit od 6 evra po satu. Ukoliko Start ima na raspolaganju 26000 evra za obuku i maksimalan broj 51200 mogućih projekat/sati mesečno, odrediti na koji nacin kompanija treba da obuči svoje zaposlene kako bi ostvarila maksimalnu dobit.

### 2.1.3 Zadaci za vežbu

**Zadatak 2.12** Za svaku narednu zagonetku, napisati program koji dodeljuje različite vrednosti različitim karakterima tako da suma bude zadovoljena:

```
GREEN + ORANGE = COLORS
MANET + MATISSE + MIRO + MONET + RENOIR = ARTISTS
COMPLEX + LAPLACE = CALCULUS
THIS + IS + VERY = EASY
CROSS + ROADS = DANGER
FATHER + MOTHER = PARENT
WE + WANT + NO + NEW + ATOMIC = WEAPON
EARTH + AIR + FIRE + WATER = NATURE
SATURN + URANUS + NEPTUNE + PLUTO = PLANETS
```

```
SEE + YOU = SOON
NO + GUN + NO = HUNT
WHEN + IN + ROME + BE + A = ROMAN
DONT + STOP + THE = DANCE
HERE + THEY + GO = AGAIN
OSAKA + HAIKU + SUSHI = JAPAN
MACHU + PICCHU = INDIAN
SHE + KNOWS + HOW + IT = WORKS
COPY + PASTE + SAVE = TOOLS
```

**Zadatak 2.13** Za svaku narednu zagonetku, napisati program koji dodeljuje različite vrednosti različitim karakterima tako da suma bude zadovoljena:

```
THREE + THREE + ONE = SEVEN
NINE + LESS + TWO = SEVEN
ONE + THREE + FOUR = EIGHT
THREE + THREE + TWO + TWO + ONE = ELEVEN
SIX + SIX + SIX = NINE + NINE
SEVEN + SEVEN + SIX = TWENTY
ONE + ONE + ONE + THREE + THREE + ELEVEN = TWENTY
EIGHT + EIGHT + TWO + ONE + ONE = TWENTY
ELEVEN + NINE + FIVE + FIVE = THIRTY
NINE + SEVEN + SEVEN + SEVEN = THIRTY
TEN + SEVEN + SEVEN + SEVEN + FOUR + FOUR + ONE = FORTY
TEN + TEN + NINE + EIGHT + THREE = FORTY
FOURTEEN + TEN + TEN + SEVEN = FORTYONE
NINETEEN + THIRTEEN + THREE + TWO + TWO + ONE + ONE + ONE = FORTYTWO
FORTY + TEN + TEN = SIXTY
SIXTEEN + TWENTY + TWENTY + TEN + TWO + TWO = SEVENTY
SIXTEEN + TWELVE + TWELVE + TWELVE + NINE + NINE = SEVENTY
TWENTY + TWENTY + THIRTY = SEVENTY
FIFTY + EIGHT + EIGHT + TEN + TWO + TWO = EIGHTY
FIVE + FIVE + TEN + TEN + TEN + TEN + THIRTY = EIGHTY
SIXTY + EIGHT + THREE + NINE + TEN = NINETY
ONE + NINE + TWENTY + THIRTY + THIRTY = NINETY
```

**Zadatak 2.14** Za svaku narednu zagonetku, napisati program koji dodeljuje različite vrednosti različitim karakterima tako da jednakost bude zadovoljena:

```
MEN * AND = WOMEN
COGITO = ERGO * SUM
((JE + PENSE) - DONC) + JE = SUIS
FERMAT * S = LAST + THEOREM.
WINNIE / THE = POOH
TWO * TWO + EIGHT = TWELVE
```

**Zadatak 2.15** Uraditi sve zadatke koji su pobrojani ovde:
http://www.primepuzzle.com/leeslatest/alphameticpuzzles.html

**Zadatak 2.16** Napisati program koji učitava ceo broj $n$ i ispisuje magičnu sekvencu $S$ brojeva od 0 do $n-1$. $S = (x_0, x_1, ..., x_{n-1})$ je magična sekvenca ukoliko postoji $x_i$ pojavljivanja broja $i$ za $i = 0, 1, ..., n-1$.

**Zadatak 2.17** Čistačica Mica sređuje i čisti kuće i stanove. Da bi sredila i počistila jedan stan potrebno joj je 1 sat, dok joj je za kuću potrebno 1.5 sati. Prilikom čišćenja, Mica potroši neku količinu deterdženta, 120ml po stanu, odnosno 100ml po kući. Mica zaradi 1000 dinara po svakom stanu, odnosno 1500 dinara po kući. Ukoliko Mica radi 40 sati nedeljno i ima 5l deterdženta na raspolaganju, koliko stanova i kuća je potrebno da očisti kako bi imala najveću zaradu?

**Zadatak 2.18** Marija se bavi grnčarstvom i pravi šolje i tanjire. Da bi se napravila šolja, potrebno je 6 minuta, dok je za tanjir potrebno 3 minuta. Pri pravljenju šolje potroši se 75 gr, dok se za tanjir potroši 100 gr gline. Ukoliko ima 20 sati na raspolaganju za izradu svih proizvoda i 250 kg gline, a zarada koju ostvari iznosi 2 evra po svakoj šolji i 1.5 evra po tanjiru, koliko šolja i tanjira treba da napravi kako bi ostvarila maksimalnu zaradu?

**Zadatak 2.19** Jovanin komšija preprodaje računare i računarsku opremu. Očekuje isporuku računara i štampača. Pri tom, računari su spakovani tako da njihova kutija zauzima 360 kubnih decimetara prostora, dok se štampači pakuju u kutijama koje zauzimaju 240 kubnih decimetara prostora. Komšija se trudi da mesečno proda najmanje 30 računara i da taj broj bude bar za 50% veći od broja prodatih štampača. Računari koštaju 200 evra po nabavnoj ceni, a prodaju se po ceni od 400 evra, dok štampači koštaju u nabavci 60 evra i prodaju se za 140 evra. Magacin kojim komšija raspolaže ima svega 30000 kubnih decimetara prostora i mesečno može da nabavi robu u iznosu od najviše 14000 evra. Koliko računara, a koliko štampača komšija treba da proda kako bi se maksimalno obogatio?

# 3

# Komponentno programiranje

Potrebno je imati instalirane sledeće programe:

(a) IntelliJ IDEA (može se preuzeti na strani https://www.jetbrains.com/idea/

(b) JavaFX Scene Builder (može se preuzeti na strani https://www.oracle.com/technetwork/java/javase/downloads/sb2download-2177776.html

Literatura:

(a) http://docs.oracle.com/javafx/2/

(b) http://www.java2s.com/Tutorials/Java/JavaFX/

## 3.1 Uvod

**Zadatak 3.1** Uvodni primer kreiranja aplikacije. StackPane.

**Zadatak 3.2** Forme za unos podataka. GridPane.

**Zadatak 3.3** Slajder. Transformacija kruga. BorderPane.

## 3.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 3.4** Pogledati zadatke iz odeljka 4.1.2. u kojima se pravi GUI.

# 4

# Konkurentno programiranje

Potrebno je imati instalirane sledeće programe:

(a) IntelliJ IDEA (može se preuzeti na strani https://www.jetbrains.com/idea/

(b) Java JDK 1.8 (može se preuzeti na strani http://www.oracle.com/technetwork/java/javase/downloads/index.html

Literatura:

(a) http://www.roseindia.net/java/thread/index.shtml

(b) http://www.javamex.com/tutorials/threads/

## 4.1 Uvod, kreiranje niti, prioritet, sinhronizacija.

### 4.1.1 Uvodni primeri

**Zadatak 4.1** Kreiranje niti korišćenjem java.lang.Thread klase.

**Zadatak 4.2** Kreiranje niti korišćenjem java.lang.Runnable interfejsa.

**Zadatak 4.3** Primer pokretanja procesa sa različitim prioritetom. Join metod.

**Zadatak 4.4** Napisati program kojim se paralelizuje izračunavanje izraza $(2+36)*(15+100)$. Koristiti zasebnu nit za svaku od matematičkih operacija koje se javljaju u izrazu. Main nit treba samo da ispiše vrednost izraza na standardni izlaz.

**Zadatak 4.5** Napisati program kojim se paralelizuje izračunavanje proizvoda $n$-dimenzionalnog vektora skalarom. Korisnik unosi dimenziju vektora, elemente i skalar sa standardnog ulaza. Koristiti zasebnu nit za svaki element vektora. Main nit treba samo da ispiše rezultujući vektor na standardni izlaz.

### 4.1.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 4.6** Napisati program kojim se paralelizuje izračunavanje skalarnog proizvoda dva $n$-dimenzionalna vektora. Korisnik unosi dimenziju vektora i njihove elemente sa standardnog ulaza. Main nit treba samo da ispiše rezultujući skalar na standardni izlaz, a za izračunavanje koristiti zasebne niti.

**Zadatak 4.7** Napisati program kojim se paralelizuje izračunavanje zbira elemenata dva niza dužine $n$ ($[a_1a_2...a_n]+[b_1b_2...bn] = [a1+b_1a_2+b_2...a_n+b_n]$). Korisnik unosi elemente sa standardnog ulaza. Main nit treba samo da ispiše rezultujući niz na standardni izlaz, a za izračunavanje koristiti zasebne niti.

**Zadatak 4.8** Napisati program koji proverava koji su elementi niza prosti brojevi i ispisuje ih na standardni izlaz. Korisnik unosi elemente sa standardnog ulaza. Za proveru da li su odgovarajući elementi niza prosti brojevi koristiti zasebne niti.

**Zadatak 4.9** Napisati program koji računa proizvod dve kvadratne matrice reda $n$ koje korisnik unosi sa standardnog ulaza. Za množenje odgovarajuće vrste i kolone koristiti zasebne niti, dok main nit treba samo da ispiše rezultujuću matricu na standardni izlaz.

**Zadatak 4.10** Napisati program koji paralelno generiše random metodom `n` pozitivnih četvorocifrenih brojeva i određuje među njima broj koji ima najveću apsolutnu vrednost razlike cifre jedinica i hiljada. Za generisanje određenog broja i određivanje apsolutne vrednosti razlike njegove cifre jedinica i hiljada koristiti zasebne niti. Broj niti treba da odgovara broju procesora koji su na raspolaganju. Nakon obrade brojeva, glavna (`main`) nit treba da prikaže traženi broj. Ukoliko ima više takvih, prikazati od njih onaj za koji je `random_seed + i` najmanje. Pretpostaviti da su ulazni podaci ispravni.
UPUTSTVO: *Za generisanje `i`-tog broja, koristiti objekat klase `Random` sa argumentom `random_seed + i`. Četvorocifreni brojevi se mogu dobiti od random generisanog broja iz intervala* $[0, 9000)$ *kome je dodat broj* 1000*.*

**Zadatak 4.11** Dodati zadatku 4.10 grafički korisnički interfejs.
*GUI:* Za osnovni kontejner aplikacije upotrebiti Border Pane dimenzije $200 \times 200$ sa marginama veličine 20. U levu oblast postaviti tekstualno polje za unos broja `n`, u centralnu tekstualno polje za unos broja koji predstavlja `random_seed`, u desnu dugme za učitavanje, a u donju labelu u kojoj treba prikazati tekst sa rezultatom. Korisnik unosi u levo tekstualno polje broj `n`, u centralno veličinu za `random_seed`, nakon čega pritiskom na dugme započinje generisanje brojeva i obrada. Rezultat obrade prikazati u labeli.
UPUTSTVO: *Izvlačenje podataka iz tekstualnih polja i određivanje rezultata implementirati kao metod za akciju nad dugmetom.*

**Zadatak 4.12** Izmeniti zadatak 4.10 tako da se prikažu svi brojevi među generisanim koji imaju najveću apsolutnu vrednost razlike cifre jedinica i hiljada.

**Zadatak 4.13** Napisati program koji paralelno generiše random metodom `n` pozitivnih trocifrenih brojeva i određuje ukupan zbir cifara svih brojeva. Za generisanje određenog broja i određivanje traženog zbira koristiti zasebne niti. Broj niti treba da odgovara broju procesora koji su na raspolaganju. Nakon obrade svih brojeva, glavna (`main`) nit treba samo da prikaže izračunat zbir cifara. Pretpostaviti da su ulazni podaci ispravni. UPUTSTVO: *Za generisanje `i`-tog broja, koristiti objekat klase `Random` sa argumentom `random_seed + i`. Trocifreni brojevi se mogu dobiti od random generisanog broja iz intervala* $[0, 900)$ *kome je dodat broj* 100*.*

### 4.1.3 Zadaci za vežbu

**Zadatak 4.14** Sve zadatke iz prethodnog odeljka rešiti korišćenjem Runnable interfejsa.

**Zadatak 4.15** Napisati program koji proverava koji su elementi niza savršeni brojevi i ispisuje ih na standardni izlaz. Korisnik unosi elemente sa standardnog ulaza. Za proveru da li su odgovarajući elementi niza savršeni brojevi koristiti zasebne niti. Savršenim se naziva prirodni broj kome je zbir pozitivnih delilaca različitih od sebe samog jednak tom broju ili, analogno, onaj broj $n$ kojem je zbir delilaca $2n$. Primer savršenog broja: 28 $(1 + 2 + 4 + 7 + 14 = 28)$.

**Zadatak 4.16** Napisati program koji računa determinantu matrice reda 3 koju korisnik unosi sa standardnog ulaza. Koristiti zasebne niti za izračunavanje pomoćnih determinanti drugog reda, dok main nit treba samo da ispiše rezultujući skalar na standardni izlaz.

**Zadatak 4.17** Dodati zadatku 4.13 grafički korisnički interfejs.
*GUI:* Za osnovni kontejner aplikacije upotrebiti Grid Pane dimenzije $200 \times 200$ sa tri vrste i dve kolone i marginama veličine 15. U prvu vrstu postaviti tekstualno polje za unos broja `n`, u drugu

tekstualno polje za unos broja koji predstavlja `random_seed`, a u poslednju dugme za učitavanje i labelu u kojoj treba prikazati tekst sa rezultatom. Korisnik unosi u odgovarajuća tekstualna polja broj `n` i veličinu za `random_seed`, nakon čega pritiskom na dugme započinje generisanje brojeva i obrada. Rezultat obrade prikazati u labeli.
UPUTSTVO: *Izvlačenje podataka iz tekstualnih polja i određivanje rezultata implementirati kao metod za akciju nad dugmetom.*

## 4.2 Zaključavanje, java katanci i uslovni redovi čekanja.

### 4.2.1 Uvodni primeri

**Zadatak 4.18** Primer narušavanja integriteta podataka. Program demonstrira rad banke čiji transfer metod predstavlja kritičnu sekciju dok su klijenti niti koje izvršavaju prebacivanje novca sa jednog računa na drugi račun.

**Zadatak 4.19** Zaključavanje kritične sekcije. Vrši se sinhronizacija niti klijenata da bi se obezbedila konzistentnost podataka, u ovom slučaju, ukupan saldo u banci.

**Zadatak 4.20** Uslovni redovi čekanja za ulazak u kritičnu sekciju. Ne dozvoljavamo da klijenti koji ne poseduju dovoljnu kolicinu novca na računu izvrše transfer.

### 4.2.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 4.21 čokoladni svet** Postoji veliko skladište čokolade koje ima 100 hiljada čokoladica. Pored skladišta postoje 4 prodavnice u koje može da se smesti najviše 1000 čokoladica. Inicijalno su prazne. Postoji i 100 ljudi i svaki od njih na svakih 0.5 sec oseti potrebu za čokoladicom i odlazi do neke od nasumično odabrane prodavnice. Potom u prodavnici, ako je dostupno, uzme nasumično od 1 do 7 čokoladica. Brzo nakon toga ih pojedu, ali uvek zapamte koliko su dosad uzeli komada. Prodavnice se snabdevaju periodično iz skladišta na svakih 5 sec. One uvek uzmu toliko čokoladica da se dopune do maksimalnog kapaciteta. Modelovati sistem i pobrinuti se da u svakom momentu ukupna količina čokoladica u skladištu, prodavnicama i broj pojedenih komada bude 100 hiljada. Ispisavati promene stanja svake sekunde, koristiti odvojenu nit za ovo.

## 4.3 Grafički korisnički interfejs

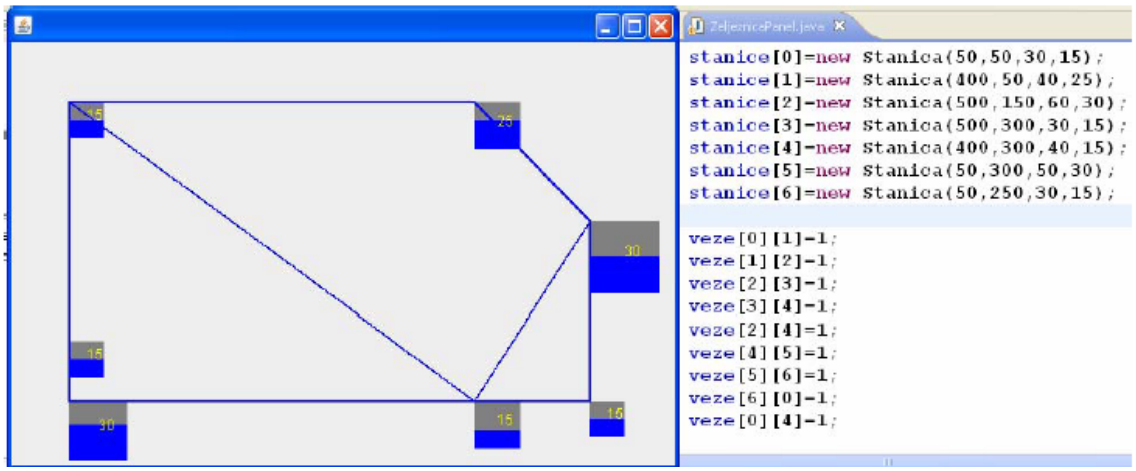
Korisni linkovi:

(a) http://www.roseindia.net/java/example/java/awt/index.shtml

(b) http://www.roseindia.net/tutorial/java/swing/JFrame.html

### 4.3.1 Uvodni primer - železnička stanica

**Zadatak 4.22** Železnica nudi usluge prevoza robe. Vozovi kreću iz date stanice u nasumičnu sledeću samo ako su ili potpuno puni, ili potpuno prazni. Tada u slučaju da su potpuno puni, pokušavaju da istovare svu svoju robu u prvoj sledećoj stanici, inače pokušavaju da napune svoje vagone do punog kapaciteta. Veza između svaka dva grada na mapi ima dva koloseka, tako da prugu ne treba gledati kao deljeni resurs. Sa druge strane, stanice su deljeni resursi, iz razloga što u datom momentu više vozova može čekati na robu u njoj. Stanice su raspoređene kao na slici ispod, a veze između njih su zadate matricom povezanosti. Potrebno je kreirati 5 vozova i inicijalno ih postaviti na različite polazne stanice. Vozovi će imati svoj kapacitet proizvoljno određen u opsegu od 5-10. Takođe, inicijalno su svi prazni, dakle, moraće da uzmu robu iz polazne stanice. Svaki voz treba implementirati kao zasebnu nit i grafički ga predstaviti kao pravougaonik (čija će dužina zavisiti od njegovog kapaciteta). Mehanizme čekanja kod vozova realizovati korišćenjem kondicionih listi. Trenutni kapacitet stanica prikazan je brojem, a potrebno je prikazati i odnos trenutnog stanja i kapaciteta.




Slika 4.1: *Na slici je dat grafički prikaz železnice, a u kodu pored se mogu videti tačne koordinate, kapaciteti i trenutno stanje robe po stanicama, kao i matrica povezanosti stanica*

### 4.3.2 Zadaci za samostalni rad sa rešenjima

**Zadatak 4.23** **Lanac ishrane** U centralnom delu prozora nalazi se njiva sa kupusom, oko nje je livada. Godišnji prinos iznosi 10 glavica, a kako je ovo ubrzana simulacija, pretpostavimo da godina traje samo 10 sekundi. Ukoliko zec naiđe na njivu, on uglavnom pojede jednu, a ako je baš jako gladan najviše dve glavice kupusa, nakon toga napušta njivu. Ako na njivi nema kupusa, zec čeka dok se situacija ne promeni. Zečiji algoritam kretanja je vrlo jednostavan, ukoliko naiđe na njivu ili zidove prozora, on se odbija pri čemu je izlazni ugao jednak ulaznom. Vuk se kreće na identičan način i njemu nije interesantna njiva. Međutim, ukoliko naiđe na zeca on će ga pojesti bez obzira gde se on nalazi, bilo na njivi, bilo na livadi. Napraviti okvir dimenzija 400 x 600 sa prozorom dimenzija 350 x 550 na kom je iscrtana zela njiva dimenzija 100 x 100 na sredini. Svaki zec je zasebna nit pri čemu je potrebno implementirati gore opisano kretanje, detektovanje ulaska u njivu, konzumaciju kupusa i blokiranje ukoliko ga nema dovoljno. Zec grafički izgleda kao plavi krug prečnika 10. Kreirati ukupno 20 zečeva van njive. Svaki vuk je zasebna nit sa gore opisanim kretanjem, pri čemu se on ne blokira se na njivi već samo prelazi preko nje. Vuk je grafički prikazan crnim krugom prečnika 40 i potrebno je da detektuje zeca kao objekat, a to je kada se dva kruga preseku. U tom momentu zec nestaje sa ekrana, tj. uklanja se iz kolekcije zečeva. Kreirati ukupno 2 vuka van njive. Njiva je jedini deljeni resurs i treba ga zaštiiti korišćenjem semafora i liste čekanja u slučaju da nema dovoljno kupusa (primetiti da nema gornje granice, jer zečevi ne donose kupus, već ga samo uzimaju). Njiva je obnovljivi resurs, tako da i ona treba da bude nit, kako bi sa vremena na vreme povećavala količinu kupusa (10 glavica na svakih 10 sekundi).

### 4.3.3 Zadaci za vežbu

**Zadatak 4.24** **Gradilište** Lokalna građevinska firma gradi četiri petospratnice. Petospratnice se grade na presecima stranica kvadrata dimenzija 500 x 500. Za svaku petospratnicu je obezbeđeno po 2 vozila, jedno za transport građevinskog materijala, a drugo za transport ostalih materijala. Prvo vozilo je kamion, a drugo veliki kombi. Razlika je u kapacitetu i brzini kretanja: kamion prenosi duplo više (10000 kg : 5000 kg), ali se krede duplo sporije od kombija (postaviti da je pomeraj za kamion duplo manji nego pomeraj za kombi). Građevinski materijal se nalazi u centralnom delu ekrana (na preseku dijagonala kvadrata), dok se ostali materijali nalaze na obodu prozora desno i levo od centra gradilišta. Da bi se rad na zgradi odvijao regularno, potrebno je imati dovoljne količine građevinskog i ostalog materijala. Kada jednog materijala ponestane (potrebno je bar 100 kg građevinskog i 50 kg ostalog materijala) izgradnja stoji, u suprotnom se svake sekunde utroši po 100 kg jednog i 50 kg drugog materijala. Na skladištima građevinskog i ostalog materijala uvek ima dovoljno da se napuni kamion ili kombi, dakle tu ne postoji nikakvo čekanje. Za izgradnju jednog sprata potrebno je 20 tona materijala (10 građevinskog i 10 ostalog). Kada se zgrade završe vozila obustavljaju rad(implementirati listu čekanja ovde). Na početku svi

kamioni su na skladištu građevinskog materijala i kreću svaki ka petospratnici za koju su zaduženi (oni se kreću po dijagonalama), predstaviti ih plavim kvadratima dimenzije 20. Kombiji (dva postavljena na levom skladištu ostalog materijala, a dva na desnom) se kreću po vertikalama svaki ka petospratnici za koju je zadužen, predstaviti ih zelenim kvadratima stranice 10. Vozila se, čim prebace materijal, vraćaju na skladište sa kog su pošli. Zgrade predstaviti kao kvadrate dimenzija 100 x 100 i ispisivati koliko je trenutno upotrebljeno građevinskog, a koliko ostalog materijala, kad je gradnja gotova, obojiti kvadrat u žuto.

**Zadatak 4.25 Diskoteka** Ekran dimenzija 500 x 500 je podeljen na dva dela, levi i desni iste širine. U levom delu se nalazi bankomat sa neograničenom količinom novca. Predstaviti bankomat kao pravougaonik dimanzija 50 x 50 na sredini leve ivice ekrana. Na celom desnom delu je diskoteka predstavljena pravougaonikom 250 x 500. U diskoteci je 100 ljudi, i svako ima slučajan broj novčanica od 200 dinara, i to ne manje od 5, a ne više od 20. Budući da vreme brzo prolazi, svako troši po jednu novčanicu od 200 dinara u sekundi. Algoritam kretanja čoveka je sledeći: ako nema novca, kreće se pravolinijski ka bankomatu i podiže još 1000 dinara. Ako neko već podiže novac u tom trenutku, čeka ga da završi (implementirati listu čekanja ovde). Ako je podigao novac, najkraćim putem se vraća u diskoteku, i potom se haotično kreće u njoj (na slučajan način birati smer kretanja, ali voditi računa da ne izlazi iz diskoteke). Ljude predstaviti crnim krugovima poluprečnika 5, koji su puni ako čovek ima novca, a prazni ukoliko mu je ponestalo novca.

# 5

# Generičko programiranje

Potrebno je imati instaliran g++.

Literatura:

(a) http://www.generic-programming.org/

(b) http://www.cplusplus.com/

## 5.1 Osnove programskog jezika C++, šabloni funkcija

### 5.1.1 Uvodni primeri

**Zadatak 5.1** Prvi c++ program.

**Zadatak 5.2** Prostor imena.

**Zadatak 5.3** Klasa Trougao.

**Zadatak 5.4** Šabloni funkcija. Opcioni argumenti šablona funkcija.

**Zadatak 5.5** Napisati šablone funkcija za određivanje:

(a) zbira dva objekta

(b) n-tostruke vrednosti nekog objekta

(c) minimalne vrednosti dva objekta

Primeniti prethodne šablone na:

(A) cele brojeve

(B) realne brojeve

(C) objekte klase Trougao

**Zadatak 5.6** Klasa Osoba.

**Zadatak 5.7** Napisati šablone funkcija za:

(a) kreiranje niza tipa T dužine n date kao argument funkcije

(b) učitavanje niza tipa T dužine n date kao argument funkcije

(c) ispisivanje niza tipa T dužine n date kao argument funkcije

(d) određivanje maksimuma niza tipa T dužine n

Primeniti prethodne šablone na:

(A) niz celih brojeva

(B) niz realnih brojeve

(C) niz objekata klase Osoba

### 5.1.2 Zadaci za vežbu

**Zadatak 5.8** Napisati klasu Datum i u njoj obezbediti:

(a) konstruktor sa podrazumevanim vrednostima (1.1.2018.)

(b) pristupne metode

(c) metod za izmenu datuma

(d) metode za čitanje i ispisivanje datuma

(e) aritmetičke operatore +,- i operatore poređenja <, ==, >

**Zadatak 5.9** Napisati klasu KompleksanBroj i u njoj obezbediti:

(a) konstruktor sa dva argumenta (realni i imaginarni deo) i podrazumevanim vrednostima (0,0)

(b) pristupne metode

(c) metod za izmenu kompleksnog broja

(d) metode za čitanje i ispisivanje kompleksnog broja

(e) aritmetičke operatore +,-,*

(f) metod za određivanje modula

(g) metod za određivanje argumenta

**Zadatak 5.10** Napisati sledeće šablone funkcija:

(a) Funkciju koja pronalazi maksimalni element u nizu.

(b) Funkciju koja pronalazi minimalni element u nizu.

(c) Funkciju koja pronalazi prosek niza.

Primeniti takve šablone funkcija nad nizovima:

(A) Celobrojnih vrednosti

(B) Realnih vrednosti

(C) Trodimenzionih tačaka (kod tačaka udaljenost od (0,0,0) definise magnitudu, a po toj udaljenosti se dalje računa i prosek)

## 5.2 Šabloni klasa

### 5.2.1 Uvodni primeri

**Zadatak 5.11** Šablon klasa Koordinate za predstavljanje koordinata tačaka u prostoru.

### 5.2.2 Zadaci za vežbu

**Zadatak 5.12** Napraviti šablon od klase Trougao tako da dužine stranica mogu biti različitog tipa. Napisati program koji testira šablon klasu Trougao za stranice tipa int.

**Zadatak 5.13** Napisati šablon klasu Radnik u okviru prostora imena kompanija. Radnika karakterišu jedinstveni identifikacioni broj id, tipa int i njegova plata (u dinarima), tipa T koji je parametar šablona. U klasi obezbediti:

(a) konstruktor sa dva argumenta za polja podatke

(b) operatore poređenja <, == (poređenje radnika se svodi na poređenje njihovih plata, pretpostaviti da za objekte tipa T postoje definisani operatori poređenja)

(c) metode za računanje visine plate izražene u evrima i dolarima (smatrati da je kurs evra 119.2, a dolara 100.3), pretpostaviti da za objekte tipa T postoji definisano množenje sa konstantama i da će rezultat izračunavanja biti tipa T

Napisati program koji testira šablon klasu Radnik za platu tipa double.

**Zadatak 5.14** Napisati šablon klasu Kvadar u okviru prostora imena geometrija koja kao polja podatke ima širinu, dužinu i visinu kvadra. Sva tri podatka su tipa T koji je parametar šablona. U klasi obezbediti:

(a) konstruktor sa tri argumenta za polja podatke

(b) metod za računanje zapremine kvadra (pretpostaviti da za objekte tipa T postoji definisana operacija množenja i da će rezultat izračunavanja biti tipa T)

(c) operatore poređenja <, == (poređenje kvadara se svodi na poređenje njihovih zapremina, pretpostaviti da za objekte tipa T postoje definisani operatori poređenja)

Napisati program koji testira šablon klasu Kvadar za podatke tipa int.

## 5.3 STL

### 5.3.1 Uvodni primeri

**Zadatak 5.15** Korisnik unosi cene artikala sve dok ne unese 0 (0 je oznaka za kraj ulaza, ne čuva se u kolekciji). Koristeći pogodnu kolekciju iz STL-a za čuvanje cena, napisati program koji:

(a) dodaje cene artikala u kolekciju u redosledu u kom stižu. Nakon učitavanja ispisati cenu svakog trećeg artikla počev od najnovijeg ka najstarijem.

(b) dodaje cene artikala u kolekciju u obrnutom redosledu od onog u kom stižu. Nakon učitavanja cena (čiji se unos završava sa 0), korisnik dodatno unosi još jednu cenu, nakon čega treba obrisati sve artikle skuplje od unete cene i ispisati modifikovanu kolekciju.

(c) dodaje cene artikala tako da neparne stavlja na početak, a parne na kraj kolekcije i tako do kraja ulaza. Ispisati dobijenu kolekciju i zbir unetih cena.

(d) eliminiše duplikate cena iz unetog niza, a zatim ispisuje cene prvo u rastućem, a zatim i u opadajućem poretku.

**Zadatak 5.16** Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim podatke o $n$ studenata u obliku *korisničko ime studenta sa alas-a prosek*. Nakon toga, korisnik unosi korisničko ime studenta čiji prosek želi da mu se prikaže. Ukoliko takav student ne postoji u bazi, omogućiti korisniku da doda podatke o tom studentu, a ukoliko postoji, prikazati trenutni prosek na standardni izlaz i omoguciti korisniku da promeni trenutni prosek za tog studenta. Koristiti pogodnu kolekciju iz STL-a za čuvanje podataka.

**Zadatak 5.17** Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim dva vektora, gde je prvi vektor dužine $n$ i proizvoljnog tipa, a drugi vektor dužine $n-1$ i sačinjen je od stringova čije su vrednosti + ili -. Napisati šablon funkciju za učitavanje u vektor proizvoljnog tipa, kao i za ispis proizvoljnog vektora i šablon funkciju koja prihvata dva vektora pomenutog formata i primenjuje operaciju sabiranja ili oduzimanja nad prvim vektorom i ispisuje dobijeni rezultat. Na primer, ako su učitani vektori: $462 - 34$ i $+ + -+$ dobiće se: $4 + 6 + 2 - (-3) + 4 = 19$. Testirati program nad vektorom celih i realnih brojeva. Pretpostaviti da su ulazni podaci u ispravnom formatu.

**Zadatak 5.18** Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim kolekciju dužine $n$ proizvoljnog tipa, i menja je tako što nad svakim elementom kolekcije primenjuje unarnu transformaciju:

(a) inkrementaciju.

(b) dekrementaciju.

(c) kvadriranje.

Napisati šablon funkciju za učitavanje i ispis proizvoljne kolekcije, kao i šablon funkciju koja menja kolekciju na pomenut način. Testirati program nad vektorom celih brojeva i listom realnih brojeva. Pretpostaviti da su ulazni podaci u ispravnom formatu.

**Zadatak 5.19** Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim kolekciju dužine $n$ proizvoljnog tipa, i menja je tako što zadržava samo elemente koji ispunjavaju sledeći unarni uslov:

(a) da su deljivi sa 3.

(b) da nisu deljivi sa 5.

(c) da su veci od 7.

(d) da su manji od 11.

Napisati šablon funkciju za učitavanje i ispis proizvoljne kolekcije, kao i šablon funkciju koja menja kolekciju na pomenut način. Testirati uslove deljivosti/nedeljivosti nad vektorima celih brojeva, a poredbene uslove nad listama realnih brojeva. Pretpostaviti da su ulazni podaci u ispravnom formatu.

**Zadatak 5.20** Uopštavanje prethodnog zadatka: Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim kolekciju dužine $n$ proizvoljnog tipa i menja kolekciju tako što zadržava samo elemente koji ispunjavaju unarni uslov:

(a) da su deljivi sa dodatno učitanim celim brojem.

(b) da su veci od dodatno učitanog realnog broja.

Napisati šablon funkciju za učitavanje i ispis proizvoljne kolekcije, kao i šablon funkciju koja menja kolekciju na pomenut način. Testirati uslov deljivosti nad vektorom celih brojeva, a poredbeni uslov nad listom realnih brojeva. Pretpostaviti da su ulazni podaci u ispravnom formatu.

**Zadatak 5.21** Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj $n$, a zatim kolekciju dužine $n$ proizvoljnog tipa i ispisuje elemente nule linearne jednačine sa realnim koeficijentima koji se dodatno učitavaju. Ispisati i broj pronađenih nula u kolekciji. Napisati šablon funkciju za učitavanje i ispis proizvoljne kolekcije, kao i šablonski funkcional pod nazivom Linear koji pri kreiranju prihvata dva realna argumenta, odnosno koeficijente linearne jednačine. Testirati program nad vektorom celih i realnih brojeva. Pretpostaviti da su ulazni podaci u ispravnom formatu.

**Zadatak 5.22** Program učitava podatke o prodavnici u sledećem formatu, broj vrsta artikala, a zatim za svaki artikal ime (pretpostaviti da je jedinstveno), količinu i cenu. Prikazati redovne cene i stanje u prodavnici za sve artikle, a zatim prikazati i cene artikala za vreme akcije - popusta od 10% na sve. Implementirati šablon funkciju `for_each` iz biblioteke algorithm za modifikovanje kolekcije sa podacima o prodavnici na traženi način.

**Zadatak 5.23** Modifikovati prethodni program ako se akcijski popust (u procentima) unosi kao vrednost sa standardnog ulaza. Ukoliko je uneta vrednost veća od 50, podrazumevano primeniti akciju u pola cene. Napisati odgovarajući funkcional pod nazivom Akcija koji za argument prihvata vrednost popusta koji treba da se obračuna za sve artikle.

**Zadatak 5.24** Program učitava podatke o prodavnici u sledećem formatu, broj vrsta artikala, a zatim imena svih artikala, pa nakon toga raspoloživu količinu za sve artikle u odgovarajućem redosledu. Prikazati stanje u prodavnici za sve artikle. Zatim za one artikle kojih nema dovoljno na stanju (minimalno 100), postaviti količinu na 100 i prikazati novo stanje u prodavnici. Implementirati šablon funkciju `replace_if` iz biblioteke algorithm za modifikovanje kolekcije sa podacima o prodavnici na traženi način.

**Zadatak 5.25** Modifikovati prethodni program ako se minimalna količina za artikle unosi kao vrednost sa standardnog ulaza. Ukoliko je uneta vrednost veća od 100, podrazumevano postaviti količinu na 100. Napisati odgovarajući funkcional pod nazivom Dopuna koji za argument prihvata vrednost koju treba postaviti kao novu količinu za artikle kojih nema dovoljno na stanju.

### 5.3.2 Zadaci za vežbu

**Zadatak 5.26** Za vežbu implementirati preostale šablone definisane u STL biblioteci algorithm. http://www.cplusplus.com/reference/algorithm/