Programski jezici

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

Univerzitet u Beogradu
Matematički fakultet

Programske paradigme

Materijali za vežbe

Nastavnik: Milena Vujošević Janičić
Asistent: Marjana Gligorijević

Beograd
2020.

Priprema materijala:
dr Milena Vujošević Janičić, docent na Matematičkom fakultetu u Beogradu
Marjana Gligorijević, asistent na Matematičkom fakultetu u Beogradu
Branislava Živković
Nemanja Mićović, asistent na Matematičkom fakultetu u Beogradu
Milica Selaković
Milan Čugurović, asistent na Matematičkom fakultetu u Beogradu
Ivan Ristović, asistent na Matematičkom fakultetu u Beogradu

Sadržaj

1

Funkcionalni koncepti u programskom jeziku Python

Potrebno je imati instaliran Python 3.7 na računaru.
Literatura:

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

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

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

1.1 Žašto Python?

Programski jezik Python predstavlja trenutno najpopularniji programski jezik. Koristi se svuda, od prosvete preko privrede i industrije do medicine i mnogih drugih oblasti. Najbolji pokazatelj toga su trenutne rang liste popularnosti programskih jezika, na kojima Python dominira. Organizacija IEEE rangirala je Python kao #1 jezik za 2018. godinu, pre čega je bio rangiran kao #1 u 2017. godini, a #3 u 2016. godini. GitHut, vizualizacija GitHub zastupljenosti jezika, stavlja Python na izuzetno visoku #3 poziciju.

Python kultura propagira open-source ideje, zajednicu koja je povezana kako na lokalnom tako i na globalnom nivou, koja održava svoj jezik, i deli svoje znanje sa drugima. Filozofija jezika je toliko jaka da je čak ugradjena i u sam jezik. Ovo se može videti tako što se interpreteru zada komanda "import this". Tom prilikom na ekranu se prikaže kompletan manifest jezika kao i osnovne ideje i vrednosti istog.

Osnovne prednosti jezika jesu jasnost, jednostavnost, intuitivnost, konciznost i ekspresivnost, kao i izuzetno jaka i aktivna zajednica. Dodatno sjajne biblioteke koje implementiraju mnogobrojne funkcionalnosti. Cena svega ovoga jeste efikasnost, često su Python programi dosta sporiji od osnovnih konkurenata. Medjutim, na ovome se aktivno radi, pišu se biblioteke u Pythonu koje su jako efikasne (primer *numpy*) i koje umnogome popravljaju efikasnost rada u Pythonu.

Detaljnije poredjenje može se videti na linku: https://benchmarksgame-team.pages.debian.net/benchmarksgame/

1.2 Uvod, funkcije, kolekcije, moduli

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 Napisati program koji definiše praznu (bez tela) funkciju sa tri argumenta i anonimne funkcije za sabiranje dva argumenta i kvadriranje argumenta. Ilustrovati korišćenje dokstringova (dokumentacionih stringova).

Zadatak 1.7 Ilustrovati prenos argumenata funkciji i rad sa imutabilnim i mutabilnim objektima. Proveriti promene argumenata korišćenjem funkcije identifikatora.

Zadatak 1.8 Definisati funkciju printf, koja na standardni izlaz štampa svoje argumente (ne praviti nikakve pretpostavke o broju argumenata). Modifikovati definisanu funkciju tako da kao argumente prima proizvoljan broj imenovanih argumenata.
Definisati funkciju koja računa proizvod svojih argumenata ukoliko je broj prosleđenih argumenata veći od jedan, inače vraća None objekat.

Zadatak 1.9 Napisati program koji sa standardnog ulaza učitava listu celih brojeva, a zatim na standardni izlaz ispisuje njenu dužinu, sumu njenih elemenata, njen maksimalni element kao i broj pojavljivanja broja jedan u unetoj listi.

Zadatak 1.10 Korisnik sa standardnog ulaza unosi ceo broj n, a potom ciklično pomeren rastuće sortiran niz (pr. 5 6 7 8 1 2 3 4) koji ima n elemenata. Napisati program koji na standarni izlaz ispisuje sortiran niz bez ponavljanja elementa (pr. 1 2 3 4 5 6 7).

Zadatak 1.11 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 1.12 Napisati funkciju `parovi(a,b,c, d)` koja generiše listu parova celih brojeva (x, y), za koje `x` pripada segmentu $[a, b]$, a `y` pripada segmentu $[c, d]$. Testirati rad funkcije pozivom u programu.

1.3 Funkcije višeg reda

Zadatak 1.13 Marko, Petar i Pavle su polagali ispit iz predmeta programske paradigme. Napisati program koji sa standardnog ulaza učitava ocene koji su dobili, a potom ispisuje listu parova (student, ocena) na standardni izlaz.

Zadatak 1.14 Napisati program koji sa standarnog ulaza učitava nisku, a na standardni izlaz ispisuje nisku u kojoj su sva mala slova pretvorena u velika.

Zadatak 1.15 Napisati program koji na standardni izlaz ispisuje sumu, koristeći funkciju `reduce` prvih n prirodnih brojeva gde se n unosi sa standardnog ulaza.

1.3.1 Dodatni zadaci sa korišćenjem datoteka i JSON formata

Zadatak 1.16 Napisati program koji iz datoteke niska.txt učitava nisku, a u datoteku slova.txt ispisuje sve karaktere koji nisu slova.

Zadatak 1.17 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 *junak.txt*.

Zadatak 1.18 Napisati program koji iz datoteke *datoteka.txt* učitava JSON objekat, a potom na standardni izlaz ispisuje podatke o *imenu*, *prezimenu* i *godininama*.

Zadatak 1.19 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 1.19:

Listing 1.1: *korpa.json*

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

Listing 1.2: *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 1.3: *idea_cene.json*

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

Listing 1.4: *shopngo_cene.json*

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

1.3.2 Zadaci za vežbu

Zadatak 1.20 U datoteci *knjige.json* se nalaze podaci o knjigama u knjižari u sledećem formatu.

Listing 1.5: *knjige.json*

```
[ {"oznaka":broj , "naziv_autor": naziv knjige i autor, "kolicina":
   kolicina, "cena": cena po komadu}, ...]
```

Napisati program koji ispisuje listu parova (oznaka, ukupna vrednost). Ukupna vrednost se sračunava na osnovu cene i broja primeraka, a povećava se za 10 dinara ukoliko ukupna cena knjiga sa istom oznakom ne prelazi minimalnu cenu od 100 dinara.

Primer 1

```
POKRETANJE: python knjige.py
INTERAKCIJA SA PROGRAMOM:
 [('34587', 163.8), ('98762', 284.0), ('77226', 108.85000000000001), ('88112', 84.97)]
```

Sadržaj datoteke koja se koristi u primeru 1.20:

Listing 1.6: *knjige.json*

```
1 [ {"oznaka": 34587, "naziv_autor": "Learning Python, Mark Lutz", "
    kolicina": 4, "cena": 40.95},
2 {"oznaka": 98762, "naziv_autor": "Programming Python, Mark Lutz", "
    kolicina": 5, "cena": 56.80},
3 {"oznaka": 77226, "naziv_autor": "Head First Python, Paul Barry", "
    kolicina": 3, "cena": 32.95},
4 {"oznaka": 88112, "naziv_autor": "Einfuhrung in Python3, Bernd Klein",
    "kolicina": 3, "cena": 24.99} ]
```

Zadatak 1.21 Datoteka `fudbaleri.json` sadrži podatke o fudbalerima (ime, nacionalnost i broj golova) u sledećem formatu:

```
1 [ { "Ime" : "Alexis Sanchez",  "Nacionalnost" :"Cile", "Golovi" : 17} ,
    ...]
```

(a) Definisati funkciju `uporedi` koristeći lambda izraz, koja poredi dva fudbalera po broju postignutih golova. Funkcija vraća -1, 0 ili 1 ukoliko je prvi fubaler postigao manji, jednak i veći broj golova u odnosu na drugog fudbalera

(b) Napisati program koji iz izabrane datoteke izdvaja fudbalere određene nacionalnosti i sortira ih rastuće po broju golova. Željena nacionalnost (npr. 'Engleska') i ime datoteke u kojoj se nalaze podaci o fudbalerima se zadaju kao argumenti komandne linije, a rezultat rada programa se upisuje u datoteku `izabrana_nacionalnost.json` (npr. Engleska_nacionalnost.json). U slučaju greške prilikom pokretanja programa, ispisati tekst `Greska` na standardni izlaz.

Primer 1

```
POKRETANJE: python 1.py Engleska arsenal.json
ENGLESKA_NACIONALNOST.JSON
 [{"Nacionalnost": "Engleska",
  "Ime": "Alex Oxlade-Chamberlain",
  "Golovi": 2},
  {"Nacionalnost": "Engleska",
  "Ime": "Theo Walcott", "Golovi": 8}]
```

Sadržaj datoteke koja se koristi u primeru 1.21:

Listing 1.7: *fudbaleri.json*

```
1
2 [{"Nacionalnost": "Cile", "Ime": "Alexis Sanchez", "Golovi" : 17 },
3 {"Nacionalnost": "Francuska", "Ime": "Olivier Giroud", "Golovi" : 8},
4 {"Nacionalnost": "Engleska", "Ime": "Theo Walcott", "Golovi" : 8},}
5 {"Nacionalnost": "Engleska", "Ime": "Alex Oxlade-Chamberlain", "Golovi"
    : 2},
6 {"Nacionalnost": "Francuska", "Ime": "Laurent Koscielny", "Golovi" : 2
   } ]
```

Zadatak 1.22 Datoteka `utakmice.json` sadrži podatke o utakmicama (timovi koji se takmiče i vreme početka utakmice) za svaki sport posebno u sledećem formatu:

```
[ { "Timovi": "Liverpool-Arsenal", "Vreme": "18:00"} , ...]
```

(a) Napisati funkciju `u_vremenskom_intervalu(utakmice, pocetak, kraj)` koja vraća listu utakmica čiji se početak nalazi u vremenskom opsegu početak kraj. Funkciju implementirati bez korišćenja petlji.

(b) Napisati program koji sa standardnog ulaza učitava podatke o početku i kraju vremenskog intervala u formatu *%H:%M* i iz datoteka učitava podatke o utakmicama za sve sportove, i na standardni izlaz ispisuje listu utakmica čije se vreme početka nalazi u opsegu unešenog vremenskog intervala.

Primer 1

```
Pokretanje: python 1.py
Ulaz:
  Unesite početak intervala: 17:00
  Unesite kraj intervala: 18:00
Izlaz:
   {"Timovi": "Liverpool-Arsenal", "Vreme": "18:00"}
   {"Timovi": "Milan-Cheivo", "Vreme": "17:00"}
   {"Timovi": "Koln-Bayern", "Vreme": "17:30"}
```

Sadržaj datoteke koja se koristi u primeru 1.22:

Listing 1.8: *utakmice.json*

```
[{"Timovi": "Liverpool-Arsenal", "Vreme": "18:00"},
 {"Timovi": "Milan-Cheivo", "Vreme": "17:00"},
 {"Timovi": "Eibar-Real Madrid", "Vreme": "18:30"},
 {"Timovi": "Roma-Napoli", "Vreme": "16:15"},
 {"Timovi": "Koln-Bayern", "Vreme": "17:30"} ]
```

Zadatak 1.23 Napisati program koji na standardni izlaz ispisuje apsolutne putanje do svih datoteka koje sadrže reč koja se unosi sa standardnog ulaza ili u slučaju da ni jedna datoteka ne sadrži zadatu reč ispisati poruku `Ni jedna datoteka ne sadrzi zadatu rec`. Progam napisati korišćenjem lambda izraza i bez korišćenja petlji.

Primer 1

```
Pokretanje: python 1.py
Ulaz:
  Unesite zadatu rec: include
Izlaz:
  /home/student/programiranje2/ispit/1.c
  /home/student/programiranje2/ispit/2.c
  /home/student/programiranje2/ispit/3.c
```

Primer 1

```
Pokretanje: python 1.py
Ulaz:
  Unesite zadatu rec: Lenovo
Izlaz:
  Ni jedna datoteka ne sadrzi zadatu rec
```

2

Funkcionalni koncepti u programskom jeziku Haskell

Potrebno je imati instaliran GHC kompajler na računaru.
Literatura:

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

(b) https://wiki.haskell.org/Haskell

2.1 Uvod

2.1.1 Uvodni primeri

Zadatak 2.1 Napisati funkciju `main` koja ispisuje poruku `Zdravo! :)`.

Zadatak 2.2 Napisati funkciju `duplo n` koja računa dvostruku vrednost celog broja `n`.

Zadatak 2.3 Napisati funkciju `ostatak3 n` koja računa ostatak pri deljenju broja `n` brojem tri.

Zadatak 2.4 Napisati funkciju `korenCeli n` koja računa realni koren celog broja `n` korišćenjem ugrađene funkcije `sqrt`.

Zadatak 2.5 Napisati funkciju `sumaPrvih n` koja računa sumu prvih `n` prirodnih brojeva (rekurzivno, bez korišćenja formule).

Zadatak 2.6 Napisati funkciju `lista a b` koja pravi listu celih brojeva iz segmenta $[a, b]$. U slučaju da granice segmenta nisu ispravne, rezultat je prazna lista.

Zadatak 2.7 Napisati funkciju `parMax p` koja određuje veći element iz para realnih brojeva `p`.

2.1.2 Zadaci za samostalni rad sa rešenjima

Zadatak 2.8 Napisati funkciju `proizvodPrvih n` koja računa proizvod prvih `n` prirodnih brojeva (rekurzivno, bez korišćenja formule). Pretpostaviti da je argument ispravan.

Primer 1

```
POKRETANJE: proizvodPrvih 5
IZLAZ:
 120
```

Primer 2

```
POKRETANJE: proizvodPrvih 30
IZLAZ:
 265252859812191058636308480000000
```

Zadatak 2.9 Napisati funkciju `prost n` koja vraća `True` ako je `n` prost, `False` inače. Pretpostaviti da je argument ispravan.

Primer 1

```
POKRETANJE: prost 5
IZLAZ:
 True
```

Primer 2

```
POKRETANJE: prost 12
IZLAZ:
 False
```

Zadatak 2.10 Napisati funkciju `nzd a b` koja računa najveći zajednički delilac brojeva `a` i `b` (koristiti Euklidov algoritam). Pretpostaviti da su argumenti ispravni.

Primer 1

```
POKRETANJE: nzd 12 15
IZLAZ:
 3
```

Primer 2

```
POKRETANJE: nzd 12 13
IZLAZ:
 1
```

Zadatak 2.11 Napisati funkciju `tipJednacine a b c` koja vraća tip kvadratne jednačine $a * x^2 + b * x + c = 0$ (`Degenerisana`, `Jedno resenje`, `Dva resenja`, `Bez resenja`).

Primer 1

```
POKRETANJE: tipJednacine 1 2 1
IZLAZ:
 "Jedno resenje"
```

Primer 2

```
POKRETANJE: tipJednacine (-1) 8 5
IZLAZ:
 "Bez resenja"
```

Zadatak 2.12 Napisati funkciju `izDekadne x osn` koja prebacuje broj `x` iz dekadne u osnovu `osn` i funkciju `uDekadnu x osn` koja prebacuje broj `x` iz osnove `osn` u dekadnu osnovu. Pretpostaviti da je $osn > 1$ i $osn < 10$.

Primer 1

```
POKRETANJE: izDekadne 101 2
IZLAZ:
 1100101
```

Primer 2

```
POKRETANJE: uDekadnu 765 8
IZLAZ:
 501
```

Zadatak 2.13 Napisati funkciju `ceoDeo x` koja računa ceo deo korena pozitivnog broja `x` (bez korišćenja ugrađenih funkcija za koren i/ili stepen).

Primer 1

```
POKRETANJE: ceoDeo 15
IZLAZ:
 3
```

Primer 2

```
POKRETANJE: ceoDeo 100
IZLAZ:
 10
```

Zadatak 2.14 Napisati funkciju `harm n` koja pravi listu prvih `n` elemenata harmonijskog reda. Pretpostaviti da je argument ispravan.

Primer 1

```
POKRETANJE: harm 2
IZLAZ:
 [1.0,0.5]
```

Primer 2

```
POKRETANJE: harm 5
IZLAZ:
 [1.0,0.5,0.3333333333333333,0.25,0.2]
```

Zadatak 2.15 Napisati funkciju `delioci n` koja pravi listu svih pravih delioca pozitivnog broja `n`. Pretpostaviti da je argument ispravan.

Primer 1

```
POKRETANJE: delioci 12
IZLAZ:
 [2,3,4,6]
```

Primer 2

```
POKRETANJE: delioci 13
IZLAZ:
 []
```

Zadatak 2.16 Napisati funkciju `nadovezi l1 l2 n` koja nadovezuje na listu `l1` `n` puta listu `l2`. Pretpostaviti da je argument `n` pozitivan broj.

Primer 1

```
POKRETANJE: nadovezi [1] [2] 3
IZLAZ:
 [1,2,2,2]
```

Primer 2

```
POKRETANJE: nadovezi [] [1,2,3] 2
IZLAZ:
 [1,2,3,1,2,3]
```

2.1.3 Zadaci za vežbu

Zadatak 2.17 Napisati funkciju `sumaKvadrata n` koja računa sumu kvadrata prvih `n` prirodnih brojeva (rekurzivno, bez korišćenja formule).

Primer 1

```
Pokretanje: sumaKvadrata 5
Izlaz:
 55
```

Primer 2

```
Pokretanje: sumaKvadrata 10
Izlaz:
 385
```

Zadatak 2.18 Napisati funkciju `brojDelilaca n` koja vraća broj pravih delilaca prirodnog broja `n`.

Primer 1

```
Pokretanje: brojDelilaca 5
Izlaz:
 0
```

Primer 2

```
Pokretanje: brojDelilaca 12
Izlaz:
 4
```

Zadatak 2.19 Napisati funkciju `fib n` koja računa `n`-ti element Fibonačijevog niza. Pretpostaviti da je argument ispravan.

Primer 1

```
Pokretanje: fib 5
Izlaz:
 5
```

Primer 2

```
Pokretanje: fib 12
Izlaz:
 144
```

Zadatak 2.20 Napisati funkciju `osnova x osn1 osn2` koja prebacuje broj `x` iz osnove `osn1` u osnovu `osn2`. Pretpostaviti da su `osn1` i `osn1` brojevi veći od 1 i manji od 10.

Primer 1

```
Pokretanje: osnova 100 10 3
Izlaz:
 10201
```

Primer 2

```
Pokretanje: osnova 525 8 2
Izlaz:
 101010101
```

Zadatak 2.21 Napisati funkciju `parni n` koja pravi listu prvih `n` parnih prirodnih brojeva. Pretpostaviti da je argument ispravan.

Primer 1

```
Pokretanje: parni 3
Izlaz:
 [2,4,6]
```

Primer 2

```
Pokretanje: parni 10
Izlaz:
 [2,4,6,8,10,12,14,16,18,20]
```

Zadatak 2.22 Napisati funkciju `fibLista n` koja pravi listu prvih `n` elemenata Fibonačijevog niza. Pretpostaviti da je argument ispravan.

Primer 1

```
Pokretanje: fibLista 5
Izlaz:
 [1,1,2,3,5]
```

Primer 2

```
Pokretanje: fibLista 10
Izlaz:
 [1,1,2,3,5,8,13,21,34,55]
```

Zadatak 2.23 Napisati funkciju `jednocifreniDelioci n` koja pravi listu svih jednocifrenih delilaca prirodnog broja `n`. Pretpostaviti da je argument ispravan.

Primer 1

```
Pokretanje: jednocifreniDelioci 15
Izlaz:
 [1,3,5]
```

Primer 2

```
Pokretanje: jednocifreniDelioci 40
Izlaz:
 [1,2,4,5,8]
```

2.2 Liste

2.2.1 Uvodni primeri

Zadatak 2.24 Napisati funkciju koja računa dužinu proizvoljne liste bez i sa korišćenjem šablona liste.

Zadatak 2.25 Napisati funkcije koje određuju glavu i rep proizvoljne liste bez korišćenja ugrađenih funkcija za rad sa listama.

Zadatak 2.26 Napisati funkciju `parni a b` koja generiše listu parnih celih brojeva iz segmenta $[a, b]$ i funkciju `neparni a b` koja generiše listu neparnih celih brojeva iz segmenta $[a, b]$.

Zadatak 2.27 Napisati funkciju `parovi a b c d` koja generiše listu parova celih brojeva (x, y), za koje `x` pripada segmentu $[a, b]$, a `y` pripada segmentu $[c, d]$.

Zadatak 2.28 Napisati funkciju `zavisnoY a b` koja generiše listu parova celih brojeva (x, y), za koje `x` pripada segmentu $[a, b]$, a `y` pripada segmentu $[x, b]$.

2.2.2 Zadaci za samostalni rad sa rešenjima

Zadatak 2.29 Napisati funkciju `bezbedanRep l` koja ukoliko je lista `l` prazna vraća praznu listu, inače vraća rep liste `l`, koristeći: a) uslovne izraze b) ograđene jednačine c) uparivanje šablona

Primer 1

```
POKRETANJE: bezbedanRep [1,2,3]
IZLAZ:
 [2,3]
```

Primer 2

```
POKRETANJE: bezbedanRep []
IZLAZ:
 []
```

Zadatak 2.30 Napisati funkciju `savrseni n` koja pravi listu savršenih brojeva manjih od `n`. Broj je savršen ukoliko je jednak sumi svojih faktora (tj. delilaca), ne uključujući taj broj.

Primer 1

```
POKRETANJE: savrseni 50
IZLAZ:
 [6,28]
```

Primer 2

```
POKRETANJE: savrseni 1000
IZLAZ:
 [6,28,496]
```

Zadatak 2.31 Napisati funkciju `zbirPar n` koja pravi listu parova (a, b) takvih da su `a` i `b` prirodni brojevi čiji je zbir jednak `n`.

Primer 1

```
POKRETANJE: zbirPar 1
IZLAZ:
 []
```

Primer 2

```
POKRETANJE: zbirPar 10
IZLAZ:
 [(1,9),(2,8),(3,7),(4,6),(5,5),(6,4),(7,3),(8,2),(9,1)]
```

Zadatak 2.32 Napisati funkciju `poslednji l` koja određuje poslednji element proizvoljne liste `l`.

Primer 1

```
POKRETANJE: poslednji ["ponedeljak","nedelja"]
IZLAZ:
 "nedelja"
```

Primer 2

```
POKRETANJE: poslednji [5,4,3,2,1]
IZLAZ:
 1
```

Zadatak 2.33 Napisati funkciju `spoji l` koja spaja listu listi istog tipa `l` u jednu listu.

Primer 1

```
POKRETANJE: spoji [["jedan"],["tri"],["pet"]]
IZLAZ:
 ["jedan","tri","pet"]
```

Primer 2

```
POKRETANJE: spoji [[],[1],[1,2],[1,2,3]]
IZLAZ:
 [1,1,2,1,2,3]
```

Zadatak 2.34 Napisati funkciju `sufiksi l` koja pravi listu svih sufiksa proizvoljne liste `l`.

Primer 1

```
POKRETANJE: sufiksi [1,2,3]
IZLAZ:
 [[1,2,3],[2,3],[3],[]]
```

Primer 2

```
POKRETANJE: sufiksi []
IZLAZ:
 [[]]
```

Zadatak 2.35 Napisati funkciju `izbaci k l` koja izbacuje `k`-ti element iz liste `l`. U slučaju da je zadata neispravna pozicija u listi, funkcija vraća nepromenjenu listu.

Primer 1

```
POKRETANJE: izbaci 2 [1,2,3,4,5]
IZLAZ:
 [1,2,4,5]
```

Primer 2

```
POKRETANJE: izbaci (-2) [1,2,3]
IZLAZ:
 [1,2,3]
```

Zadatak 2.36 Napisati funkciju `ubaci k n l` koja ubacuje u listu `l` na poziciju `k` element `n`. U slučaju da je zadata neispravna pozicija u listi, dodati element `n` na kraj liste.

Primer 1

```
POKRETANJE: ubaci 2 5 [1,2,3]
IZLAZ:
 [1,2,5,3]
```

Primer 2

```
POKRETANJE: ubaci 10 5 [1,2,3,4,5]
IZLAZ:
 [1,2,3,4,5,5]
```

2.2.3 Zadaci za vežbu

Zadatak 2.37 Ana uči prirodne brojeve i zanima je na koje sve načine može da razloži dati broj u obliku proizvoda dva prirodna broja, kao i koliko takvih razlaganja ima. Definisati sledeće funkcije koje pomažu Ani u rešavanju pomenutog problema:

a) `razlozi n` - za dati prirodan broj `n` vraća listu parova `(a,b)` takvih da su `a` i `b` prirodni brojevi iz intervala `[1,n]` i da je proizvod `a` i `b` jednak `n`

Primer 1

```
POKRETANJE: razlozi 3
IZLAZ:
 [(1,3),(3,1)]
```

Primer 2

```
POKRETANJE: razlozi 4
IZLAZ:
 [(1,4),(2,2),(4,1)]
```

b) `brojP n` - računa na koliko se različitih načina dati prirodan broj `n` može predstaviti u obliku proizvoda dva prirodna broja

Primer 1

```
POKRETANJE: brojP 3
IZLAZ:
 2
```

Primer 2

```
POKRETANJE: brojP 4
IZLAZ:
 3
```

Zadatak 2.38 Petar je u školi dobio zadatak da proveri da li u datoj listi parova postoji par oblika `(prethodnik, sledbenik)` za dati prirodan broj. Definisati sledeće funkcije koje pomažu Petru u rešavanju pomenutog problema:

a) `par n` - za dati prirodan broj `n` vraća par brojeva `(prethodnik,sledbenik)`

Primer 1

```
POKRETANJE: par 5
IZLAZ:
 (4,6)
```

Primer 2

```
POKRETANJE: par 12
IZLAZ:
 (11,13)
```

b) `postojiPar lista_parova` - za datu listu parova `(a,b)` i prirodan broj `n` proverava da li u listi postoji bar jedan par `(a,b)` takav da je `a` prethodnik broja `n`, a `b` sledbenik broja `n`

Primer 1

```
POKRETANJE: postojiPar [(1,4),(1,3)] 2
IZLAZ:
 True
```

Primer 2

```
POKRETANJE: postojiPar [(2,4),(2,3)] 2
IZLAZ:
 False
```

Zadatak 2.39 U toku je prijavljivanje za seminar na temu `IT problemi`. Zbog ograničenog broja mesta, sa liste prijavljenih se mogu primiti samo oni čiji je redni broj prijave manji od kapaciteta sale za održavanje seminara. Napisati funkciju `primljeni n lista` koja deli listu prijavljenih na dve liste na osnovu kapaciteta sale `n`, prva lista sadrži prvih `n` elemenata i predstavlja učesnike seminara, a druga lista sadrži ostatak i predstavlja prijavljenje koji se stavljaju na listu čekanja za sledeći termin održavanja.

Primer 1

```
POKRETANJE: primljeni 2 ["Marko Mitic", "Aleksa Jovic", "Andjela Antic"]
IZLAZ:
 [["Marko Mitic", "Aleksa Jovic"],["Andjela Antic"]]
```

Zadatak 2.40 Prodavnica `MiniMaxi` slavi 25 godina postojanja i tim povodom je organizovala posebnu akciju za svoje kupce. Svaki artikal vrednosti do 1000 dinara, čija je cena deljiva sa 25 se dobija u pola svoje cene. Napisati funkciju `usteda lista` koja na osnovu liste cena kupljenih artikala, računa koliku je uštedu ostvario kupac.

Primer 1

```
POKRETANJE: usteda [56.2,125,1025,658,300]
IZLAZ:
 212.5
```

Primer 2

```
POKRETANJE: usteda [56.2,125.99,102,658,326]
IZLAZ:
 0
```

Zadatak 2.41 Učesnici nagradne igre `Sedmica` zadaju kombinacije od sedam različitih pozitivnih brojeva manjih od 30, nakon čega se generiše dobitna kombinacija. Glavnu nagradu dobija učesnik koji je pogodio svih sedam brojeva, a utešne nagrade oni koji su imali veći broj pogodaka od promašaja u svojoj kombinaciji. Napisati funkcije:

a) `pogodak ucesnikL dobitnaL` - vraća par (a, b) takav da a predstavlja broj pogodaka, a b broj promašaja u kombinaciji učesnika koja je zadata listom `ucesnikL` na osnovu dobitne kombinacije, liste `dobitnaL`.

Primer 1

```
POKRETANJE: pogodak [2,12,15,17,19,25,29] [12,13,20,21,25,28,29]
IZLAZ:
 (3,4)
```

b) `nagrada ucesnikL dobitnaL` - vraća poruku o nagradi: `Sedmica`, `Utesna nagrada` ili `Vise srece drugi put`. Kombinacija učesnika je zadata listom `ucesnikL`, a dobitna kombinacija listom `dobitnaL`. Učesnik osvaja nagradu `Sedmica` ukoliko je pogodio dobitnu kombinaciju, utešnu nagradu ukoliko njegova kombinacija ima više pogodaka od promašaja, inače ne osvaja nista.

Primer 1

```
POKRETANJE: nagrada [9,13,15,17,19,21,23] [13,15,19,21,22,23,27]
IZLAZ:
 "Utesna nagrada"
```

2.3 Funkcije

2.3.1 Uvodni primeri

Zadatak 2.42 Definisati funkciju `spoji lista1 lista2` koja pravi listu uređenih parova tako što spaja redom elemente prve liste sa elementima druge liste u parove rezultujuće liste.

Zadatak 2.43 Definisati funkciju `uvecaj lista` koja svaki element celobrojne liste uvećava za jedan.

Zadatak 2.44 Definisati funkciju `pozitivni lista` koja izdvaja sve pozitivne elemente iz liste.

Zadatak 2.45 Definisati funkciju `prviNePozitivni lista` koja izdvaja najduži prefiks pozitivnih elemenata liste.

Zadatak 2.46 Definisati funkciju `sum lista` koja računa sumu elemenata celobrojne liste korišćenjem ugrađene funkcije `foldr` .

Zadatak 2.47 Definisati funkciju `absSume lista_listi` koja na osnovu liste listi celih brojeva pravi listu apsolutnih suma elemenata liste listi korišćenjem kompozicije funkcija za rad sa listama.

Zadatak 2.48 Definisati funkciju `sledbenici l` koja vraća listu sledbenika elemenata liste `l` koji su prirodni brojevi.

Zadatak 2.49 Definisati rekurzivnu funkciju `list_elem x l` koja proverava da li lista `l` sadrži dati element `x`. Dodatno, definisati prethodnu funkciju korišćenjem funkcija `or` i `map`.

Zadatak 2.50 Definisati funkcije za sortiranje liste parova brojeva: a) rastuće/opadajuće po prvom elementu para, b) rastuće/opadajuće po drugom elementu para.

2.3.2 Razni primeri

Zadatak 2.51 Korišćenjem funkcija `and` i `map`, definisati funkciju `list_all p l` koja proverava da li svi elementi liste `l` zadovoljavaju dato svojstvo `p`. Testirati funkciju zadavanjem svojstava `(>0)` i `(<0)`.

Zadatak 2.52 Definisati funkciju `reverse' l` za obrtanje liste `l` pomoću funkcije `foldl`.

Zadatak 2.53 Definisati funkciju `delioci n` koja pravi listu delilaca datog prirodnog broja `n` korišćenjem funkcije `filter`. Korišćenjem prethodne funkcije definisati funkciju `prost n` koja proverava da li je dati prirodan broj `n` prost. Dodatno, korišćenjem funkcije `prost` i komprehensije lista definisati funkciju `prosti n` koja određuje sve proste brojeve od 1 do `n`.

Zadatak 2.54 Definisati rekurzivnu funkciju `cifre n` koja određuje cifre datog prirodnog broja `n` (redosled cifara u listi nije bitan).

Zadatak 2.55 Definisati listu svih Armstrongovih brojeva (k-tocifreni broj je Armstrongov ako je zbir k-tih stepena cifara tog broja jednak samom broju). Odrediti 15 Armstrongovih brojeva.

2.3.3 Zadaci za samostalni rad sa rešenjima

Zadatak 2.56 Definisati rekurzivnu funkciju `varijacije l k` koja generiše listu koja sadrži sve varijacije sa ponavljanjem elemenata date liste `l` dužine `k`.

Primer 1

```
Pokretanje: varijacije [1,2,3] 2
Izlaz:
 [[1,1],[1,2],[1,3],[2,1],[2,2],[2,3],[3,1],[3,2],[3,3]]
```

Zadatak 2.57 Data je lista `l` koja sadrži liste ocena raznih učenika osnovne škole. Definisati funkciju `prosekOdlicni l` koja računa prosek svih odličnih učenika.

Primer 1

```
Pokretanje: prosekOdlicni [[1,2,3],[5,5,5],[4,5,5]]
Izlaz:
 4.8333335
```

Zadatak 2.58 Ana uči brojanje i razlikovanje boja koristeći kutiju punu jednobojnih kuglica. Ona prvo žmureći iz kutije izvuče određeni broj kuglica i poređa ih u niz u redosledu izvlačenja. Zatim izabere proizvoljnu boju i odredi na kojoj se sve poziciji u nizu izvučenih kuglica nalazi kuglica željene boje. Napisati funkciju `pozicije x l` koja vraća listu pozicija elementa `x` u listi `l`.

Primer 1

```
Pokretanje: pozicije "bela" ["plava","bela","bela","ljubicasta","bela"]
Izlaz:
 [1,2,4]
```

Zadatak 2.59 Učesnici nagradne igre `Sedmica` mogu proveriti dobitak putem sajta, gde se dobitna kombinacija objavljuje odmah nakon izvlačenja. Brojevi iz dobitne kombinacije se, radi jednostavnije provere pogodaka, uvek prikazuju u rastućem redosledu. Napisati funkciju `qsort l` koja rastuće sortira listu `l` algoritmom `qsort`.

Primer 1

```
Pokretanje: qsort [4,5,2,11,29,38,9]
Izlaz:
 [2,4,5,9,11,29,38]
```

Primer 2

```
Pokretanje: qsort [7,1,10,15,30,32,20]
Izlaz:
 [1,7,10,15,20,30,32]
```

Zadatak 2.60 Milan obožava da sakuplja sličice fudbalera. Da bi jednostavnije pratio koje mu sličice iz kolekcije još uvek nedostaju, čuva ih rastuće sortirane po rednom broju. Planirao je da dopuni svoju kolekciju na narednoj razmeni sličica, pa za tu priliku želi da iz svoje kolekcije izbaci sve nepotrebne duplikate koje će poneti na razmenu. Napisati funkciju `brisiPonavljanja l` koja briše sva uzastopna ponavljanja elemenata u listi `l`.

Primer 1

```
Pokretanje: brisiPonavljanja [4,5,5,2,11,11,11]
Izlaz:
 [4,5,2,11]
```

Primer 2

```
Pokretanje: brisiPonavljanja [10,10,10,11]
Izlaz:
 [10,11]
```

Zadatak 2.61 Kasirka Mica mora da ručno kuca artikle na kasi jer se pokvario skener barkodova. Pomozite Mici da taj posao obavi što brže grupisanjem artikala iste vrste na pokretnoj traci. Napisati funkciju `podlistePonavljanja l` koja grupiše sva uzastopna ponavljanja nekog elementa liste `l` u podlistu tako da rezultat bude lista listi.

Primer 1

```
Pokretanje: podlistePonavljanja ["jabuke","jogurt","jogurt","hleb"]
Izlaz:
 [["jabuke"],["jogurt","jogurt"],["hleb"]
```

Zadatak 2.62 Napisati funkciju `broj lista` koja vraća broj određen ciframa koje se nalaze u listi čitajući ih sa početka ka kraju liste i funkciju `brojObrnut lista` koja vraća broj određen ciframa koje se nalaze u listi čitajući ih sa kraja ka početku liste.

Primer 1

```
Pokretanje: broj [1,0,1]
Izlaz:
 101
```

Primer 2

```
Pokretanje: brojObrnut [0,5,2]
Izlaz:
 250
```

Zadatak 2.63 U toku su prijave za plesno takmičenje parova. Pored nagrade za najbolji par, dodeljuju se i pojedinačne nagrade za najboljeg ženskog i muškog takmičara. Da bi žiri jednostavnije beležio poene i odredio nagrade, potrebno je da im se dostave, pored liste parova koji se takmiče, i liste samo muških, tj. samo ženskih takmičara, odvojeno. Napisati funkciju `listaUPar lista` koja pretvara listu parova u par dve liste, tako da prva lista sadrži prve elemente

svih parova, a druga druge elemente svih parova pod pretpostavkom da je prvi u paru uvek ženska osoba, a drugi muška (implementacija funkcije unzip).

Primer 1

```
Pokretanje: listaUPar [("Ivana","Milan"), ("Ana","Jovan"), ("Anica","Petar")]
Izlaz:
  (["Ivana","Ana","Anica"], ["Milan","Jovan","Petar"])
```

Zadatak 2.64 Nakon još jedne košarkaške sezone, potrebno je sumirati rezultate svih igrača. Svaki igrač ima jedinstveni redni broj, pod kojim se u bazi, u listi prezimena, čuva njegovo prezime, a u listi pogodaka, ostvaren broj poena u sezoni. Uparivanjem odgovarajućih podataka, napraviti za svakog igrača jedinstveni par oblika `(prezime, poeni)`. Napisati funkciju `parOdListi lista1 lista2` koja pravi listu parova od dve liste, liste prezimena i liste pogodaka, tako da prvi element svakog para bude iz prve liste, a drugi element svakog para bude iz druge liste (implementacija funkcije zip).

Primer 1

```
Pokretanje: parOdListi ["Mikic","Peric","Jovic"] [100,76,96]
Izlaz:
  [("Mikic",100),("Peric",76),("Jovic",96)]
```

Zadatak 2.65 Formacija igre `kolo` je polukrug sa istaknutom ulogom prvog i poslednjeg igrača. Napisati funkciju `ucesljaj mIgraci zIgraci` koja pravi jednu formaciju za kolo naizmeničnim učešljavanjem igrača iz date grupe muških i ženskih igrača, `mIgraci` i `zIgraci`, redom.

Primer 1

```
Pokretanje: ucesljaj ["Petar","Aleksa","Filip"] ["Milica","Jovana","Anica"]
Izlaz:
  ["Petar","Milica","Aleksa","Jovana","Filip","Anica"]
```

2.3.4 Zadaci za vežbu

Zadatak 2.66 Aleksa i Luka žele da komuniciraju preko šifrovanih poruka koje predstavljaju listom niski. Aleksa šifruje željene podatke na sledeći način: ukoliko se niska koju šalje sastoji samo od cifara, na njen početak i kraj dodaje karakter `C`, ako se sastoji samo od malih slova, na njen početak i kraj dodaje karakter `S`, a inače na njen početak i kraj dodaje karakter `O`. Definisati sledeće funkcije koje pomažu Aleksi da pošalje Luki šifrovanu poruku:

a) `broj s` - za datu nisku `s` proverava da li su svi njeni karakteri cifre

Primer 1

```
Pokretanje: broj "123"
Izlaz:
  True
```

Primer 2

```
Pokretanje: broj ['c','a','o']
Izlaz:
  False
```

a) `mala s` - za datu nisku `s` proverava da li su svi njeni karakteri mala slova

Primer 1

```
Pokretanje: mala "pozdrav"
Izlaz:
  True
```

Primer 2

```
Pokretanje: mala ['C','a','o']
Izlaz:
  False
```

c) `sifruj ls` - datu listu niski `ls` transformiše na sledeći način: ukoliko se niska koju šalje sastoji samo od cifara, na njen početak i kraj dodaje karakter `C`, ako se sastoji samo od malih slova, na njen početak i kraj dodaje karakter `M`, a inače na njen početak i kraj dodaje karakter `O`.

Primer 1

```
Pokretanje: sifruj ["11","maj","petak"]
Izlaz:
  ["C11C","MmajM","MpetakM"]
```

Primer 2

```
Pokretanje: sifruj ["poz","Poz","poZ"]
Izlaz:
  ["MpozM","OPozO","OpoZO"]
```

Zadatak 2.67 Za razliku od Alekse i Luke, Ana i Milica imaju problem dešifrovanja podataka. Da bi Ana dešifrovala podatke koje joj Milica pošalje potrebno je da svaku nisku iz dobijene liste transformiše na sledeći način: ukoliko dobijena niska počinje cifrom, sa njenog početka izbaciti onoliko karaktera koliko ta niska ima cifara, a ukoliko dobijena niska počinje malim slovom, sa njenog početka izbaciti onoliko karaktera koliko ta niska ima malih slova. Pretpostaviti da će ovakvim dešifrovanjem Ana uvek dobiti ispravne izvorne podatke. Definisati sledeće funkcije koje pomažu Ani da dešifruje Miličine poruke:

a) `cifre s` - za datu nisku `s` vraća broj karaktera niske koji su cifre

Primer 1

```
POKRETANJE: cifre "11Maj"
IZLAZ:
 2
```

Primer 2

```
POKRETANJE: cifre ['m','a','j']
IZLAZ:
 0
```

b) `mala s` - za datu nisku `s` vraća broj karaktera niske koji su mala slova

Primer 1

```
POKRETANJE: mala "petak"
IZLAZ:
 5
```

Primer 2

```
POKRETANJE: mala ['C','a','o']
IZLAZ:
 2
```

c) `desifruj ls` - datu listu niski `ls` transformiše na sledeći način: ukoliko niska počinje cifrom, sa njenog početka izbacuje onoliko karaktera koliko ta niska ima cifara, a ukoliko dobijena niska počinje malim slovom, sa njenog početka izbacuje onoliko karaktera koliko ta niska ima malih slova.

Primer 1

```
POKRETANJE: desifruj ["aaa11","1minamaj2017","101petak"]
IZLAZ:
 ["11","maj2017","petak"]
```

Zadatak 2.68 Napisati funkciju `magicniParovi l` koja pravi listu parova čiji su prvi elementi, elementi liste prirodnih brojeva `l`, a drugi elementi odgovarajući magični brojevi elemenata liste `l`. Magičan broj prirodnog broja `n` se dobija kao proizvod njegovih cifara.

Primer 1

```
POKRETANJE: magicniParovi [12,101,154]
IZLAZ:
 [(12,2),(101,0),(154,20)]
```

Primer 2

```
POKRETANJE: magicniParovi [1000,99,111,222]
IZLAZ:
 [(1000,0),(99,81),(111,1),(222,8)]
```

Zadatak 2.69 Podaci o cenama artikala u prodavnici su zadati u obliku liste realnih brojeva. Definisati funkciju `prosek lista_cena` koja računa prosečnu cenu artikla u prodavnici.

Primer 1

```
POKRETANJE: prosek [199.99, 125, 10.99, 45.99, 123.50]
IZLAZ:
 101.09400000000001
```

2.4 Tipovi

2.4.1 Uvodni primeri

Zadatak 2.70 Uvesti novo ime `Par` za par dva elementa proizvoljnog tipa i napisati funkciju za množenje elemenata instance tipa `Par Int`.

Zadatak 2.71 Definisati bulovski tip podataka.

Zadatak 2.72 Definisati tip podataka `Trougao` sa konstruktorima `Jednakostranicni`, `Jednakokraki` i `Raznostranicni` i napisati funkciju koja računa obim instance tipa `Trougao`.

Zadatak 2.73 Definisati tip podataka `Zivotinja` koji može biti `Pas`, `Macka` ili `Papagaj`. Zatim definisati tip `Ljubimac` (karakterisan imenom, godinama i tipom zivotinje).

Zadatak 2.74 Definisati tip podataka `Pravougaonik` (karakterisan dvema stranicama) i instancirati klase `Show` i `Eq` nad kreiranim tipom.

Zadatak 2.75 Definisati novi rekurzivni tip `Lista` sa konstruktorima `Null` i `Konstanta` i napisati funkciju koja računa dužinu instance tipa `Lista`.

Zadatak 2.76 Napisati funckije glava i rep koje bezbedno vraćaju glavu i rep liste koristeći tip `Maybe`.

2.4.2 Zadaci za samostalni rad sa rešenjima

Zadatak 2.77 Pikseli na celobrojnoj ravni zadati su uređenim parom `type Pos = (Int, Int)`. Svaki piksel može biti uključen `(True)` ili isključen `(False)`. Na taj način kreiramo sliku `type Pic = Pos -> Bool`. Možemo pretpostaviti da je svaka slika dobro definisana. Uzimaju se u obzir samo pikseli sa koordinatama od (0,0) do (9,9).

a) Definisati slike `empty :: Pic`, `full :: Pic` i `only :: Pos → Pic` tako da `empty` ne uključuje nijedan piksel, `full` uključuje sve piksele, a `only` uključuje samo dati piksel.

Primer 1

```
POKRETANJE: empty (2,2)
IZLAZ:
 False
```

Primer 2

```
POKRETANJE: full (1,3)
IZLAZ:
 True
```

Primer 3

```
POKRETANJE: (only (1,1)) (2,3)
IZLAZ:
 False
```

b) Definisati funkcije `inverse :: Pic -> Pic`, `union :: Pic -> Pic -> Pic` i `intersect :: Pic -> Pic -> Pic` koje redom vraćaju inverz date slike, uniju dve date slike i presek dve date slike.

Primer 1

```
POKRETANJE: intersect empty full (2,4)
IZLAZ:
 False
```

Primer 2

```
POKRETANJE: union empty (only (1,1)) (1,1)
IZLAZ:
 True
```

Primer 3

```
POKRETANJE: inverse (only (1,1)) (1,1)
IZLAZ:
 False
```

c) Definisati funkciju `render :: Pic -> [[Bool]]` koja pretvara datu sliku u listu linija, pri čemu svaka linija predstavlja listu elemenata tipa `Bool`. Funkcija treba da uzme u obzir samo pozicije između (0,0) i (4,4). Prva linija sadri piksele od (0,0) do (0,4), druga od (1,0) do (1,4) i tako dalje.

Primer 1

```
POKRETANJE: render (only (3,3))
IZLAZ:
  [[False,False,False,False,False],
     [False,False,False,False,False],
     [False,False,False,False,False],
     [False,False,False,True,False],
     [False,False,False,False,False]]
```

d) Definisati funkciju `extract :: [[Bool]] → Pic` inverznu funkciji `render`. Funkcija treba da postavi samo piksele na pozicijama između $(0, 0)$ i $(4, 4)$.

Primer 1

```
POKRETANJE: extract (render full) (1,2)
IZLAZ:
  True
```

Primer 2

```
POKRETANJE: extract (render only (1,2)) (1,2)
IZLAZ:
  True
```

Zadatak 2.78 Definisati tip podataka `Map` za predstavljanje konačnog preslikavanja elemenata tipa `k` u elemente tipa `v`. Preslikavanje realizovati u obliku sortiranog binarnog stabla. Dodatno:

a) Definisi funkciju `null :: Map k v -> Bool` koja proverava da li je data mapa prazna.

b) Definisati funkciju `insertWithKey :: Ord k => (k -> v -> v -> v) -> k -> v -> Map k v -> Map k v` koja datom ključu pridružuje datu vrednost ako taj ključ nije ranije definisan u mapi, a ako jeste, onda se nova vrednost određuje primenom date funkcije na taj ključ, staru vrednost i novu vrednost.

c) Definisati funkciju `insertWith :: Ord k => (v -> v -> v) -> k -> v -> Map k v -> Map k v` koja datom ključu pridružuje datu vrednost, ako taj ključ nije ranije definisan u mapi, a ako jeste, onda se nova vrednost određuje primenom date funkcije na staru vrednost i novu vrednost.

d) Definisati funkciju `insert :: Ord k => k -> v -> Map k v -> Map k v` koja datom ključu pridružuje datu vrednost (ako je ključ ranije definisan, stara vrednost se zanemaruje).

e) Definisati funkciju `fromList :: Ord k => [(k, v)] -> Map k v` koja listu parova (ključ, vrednost) pretvara u mapu.

f) Definisati funkciju `search :: Ord k => Map k v -> k -> Maybe v` koja određuje vrednost datog ključa u mapi.

g) Definisati funkciju `findWithDefault :: Ord k => v -> k -> Map k v -> v` koja za dati ključ i mapu vraća vrednost ključa u mapi (ako vrednost tog ključa nije definisana, vraća se podrazumevana vrednost data kao prvi parametar funkcije).

h) Definisati funkciju `member :: Ord k => Map k v -> (k -> Bool)` koja za datu mapu vraća funkciju koja za dati ključ proverava da li se nalazi u mapi.

i) Definisati funkciju `adjustWithKey :: Ord k => (k -> v -> v) -> k -> Map k v -> Map k v` koja menja vrednost datom ključu tako što novu vrednost određuje primenom funkcije na ključ i staru vrednost.

j) Definisati funkciju `adjustAll :: (k -> v -> v) -> Map k v -> Map k v` koja ažurira sve vrednosti u mapi na osnovu date funkcije koja novu vrednost određuje na osnovu ključa i stare vrednosti.

k) Definisati funkciju `deleteMin :: Map k v -> ((k, v), Map k v)` koja iz date mape uklanja čvor sa najmanjim ključem.

l) Definisati funkciju `delete :: Ord k => k -> Map k v -> Map k v` koja iz date mape uklanja dati ključ.

m) Definisati funkcional `foldMap :: (a -> k -> v -> a) -> a -> Map k v -> a` koji vrsi agregaciju svih elemenata mape u infiksnom redosledu. Funkcional prima funkciju koja na osnovu prethodno agregirane vrednosti, ključa i vrednosti pridružene ključu vraća novu agregiranu vrednost i početnu vrednost agregacije.

n) Korišćenjem funkcionala `foldMap` definisati funkciju `size :: Map k v -> Int` koja izračunava veličinu mape - broj ključeva.

o) Definisati funkciju `toList :: Ord k => Map k v -> [(k, v)]` koja kreira listu parova (ključ,vrednost) za sve elemente date mape, sortiranu po vrednosti ključa.

p) Definisati funkciju `union :: Ord k => Map k v -> Map k v -> Map k v` koja pravi uniju dve mape. Ako je u obe mape vrednost pridružena istom ključu, u uniji ona treba da bude ista kao u drugoj mapi.

2.4.3 Zadaci za vežbu

Zadatak 2.79 Trapez je zadat dužinom svojih osnovica i visine redom kao `type Trapez = (Double,Double,Double)`.

a) Definisati funkciju `povrsina :: Trapez -> Double` koja vraća površinu datog trapeza. Pretpostaviti da su dužine ispravno zadate.

Primer 1

```
POKRETANJE: povrsina (16,8,4)
IZLAZ:
 48.0
```

Primer 2

```
POKRETANJE: povrsina (14,8,11)
IZLAZ:
 121.0
```

b) Definisati funkciju `sumaP :: [Trapez] -> Int -> Double` koja za datu listu trapeza i ceo broj `n` računa sumu površina prvih `n` trapeza iz liste. Pretpostaviti da su podaci ispravno zadati.

Primer 1

```
POKRETANJE: sumaP [(14,8,11),(23,15,9),(8,14,11)] 2
IZLAZ:
 292.0
```

c) Definisati funkciju `manjaP :: [Trapez] -> Double -> Bool` koja za datu listu trapeza i realan broj `n` vraća da li u listi postoji trapez čija je površina manja od `n`. Pretpostaviti da su podaci ispravno zadati.

Primer 1

```
POKRETANJE: manjaP [(16,8,4),(23,15,9),(14,8,11)] 5.2
IZLAZ:
 False
```

Zadatak 2.80 Podaci o cenama artikala u prodavnici su zadati kao `type Cene = [Double]`.

a) Definisati funkciju `razlika :: Cene -> Cene -> [Double]` koja vraća razliku cena artikala iz dve različite prodavnice. Pretpostaviti da u datim listama, podaci na istim pozicijama odgovaraju istim artiklima i da su liste jednakih dužina.

Primer 1

```
POKRETANJE: razlika [120,115,12.23] [119,174.56,89.9]
IZLAZ:
 [1.0,-59.56,-77.67]
```

b) Definisati funkciju `dupliraj :: Cene -> Int -> Cene` koja za date podatke o cenama i ceo broj `n` duplira cene svih artikala počev od `n`-tog. Pretpostaviti da je indeks `n` ispravno zadat i da se broji od 0 u listi.

Primer 1

```
Pokretanje: dupliraj [12,15.5,120] 2
Izlaz:
  [12,15.5,240.0]
```

Primer 2

```
Pokretanje: dupliraj [12,15.5,120,17.8] 0
Izlaz:
  [24.0,31.0,240.0,35.6]
```

c) Definisati funkciju `jeftinija :: Cene -> Cene -> Cene` koja duplira cene svih artikala iz prodavnice u kojoj ima više jeftinijih artikala i vraća novodobijene cene za tu prodavnicu. Pretpostaviti da u datim listama, podaci na istim pozicijama odgovaraju istim artiklima i da su liste jednakih dužina. U slučaju da je broj jeftinijih artikala isti, modifikovati cene za prvu prodavnicu.

Primer 1

```
Pokretanje: jeftinija [120,115,12.23] [119,174.56,89.9]
Izlaz:
  [240.0,230.0,24.46]
```

3

Logičko programiranje

3.1 Jezik Prolog

3.1.1 O jeziku Prolog

Prolog (eng. *PROgramming in LOGic*) je deklarativan programski jezik namenjen rešavanju zadataka simboličke prirode. Prolog se temelji na teorijskom modelu logike prvog reda. Početkom 1970-ih godina Alain Kolmerauer (eng. *Alain Colmeraуer*) i Filipe Rousel (eng. Philippe Roussel) na Univerzitetu u Marselju (eng. University of Aix-Marseille), zajedno sa Robertom Kovalskim (eng. *Robert Kowalski*) sa Odeljka Veštačke Inteligencije (eng. *Department of Artifical Intelligence*) na Univerzitetu u Edinburgu (eng. *University of Edinburgh*), razvili su osnovni dizajn jezika Prolog.

3.1.2 Instalacija BProlog-a

U okviru kursa će biti korišćena distribucija Prologa pod nazivom `BProlog`. `BProlog` se može preuzeti sa zvanične Veb strane http://www.picat-lang.org/bprolog/.

Potrebno je preuzeti adekvatnu verziju za Vaš sistem i otpakovati je. U dobijenom direktorijumu će postojati izvršiva datoteka `bp` kojom se može pokrenuti `BProlog` interpreter. Preporučeno je dodati `bp` u `PATH` kako bi `BProlog` bio dostupan iz komandne linije.

Na primer, pretpostavimo da imamo 64bitni Linux. Potrebno je preuzeti datoteku `bp81_linux64.tar.gz` i smestiti je u direktorijum po izboru, na primer `/home/korisnik/Downloads`. Potom treba izvršiti sledeće naredbe:

```
cd ~/Downloads
tar -xvf bp81_linux64.tar.gz
sudo mv BProlog /opt
sudo ln -s /opt/BProlog/bp /usr/bin/bprolog
```

Nakon toga, `BProlog` interpreter se iz konzole može pokrenuti komandom `bprolog`.

3.2 Uvod

3.2.1 Uvodni primeri

Zadatak 3.1 U bazu znanja uneti informacije o životinjama i njihovim odnosima po pitanju veličine. Napisati pravilo koje omogućava da se proveri koja je od dve životinje veća, kao i da generiše sve životinje za koje je neka životinja veća.

Zadatak 3.2 Unifikacija. Jednakost.

Zadatak 3.3 Aritmetički operatori. Operatori *is* i *cut*.

Zadatak 3.4 Rekurzivni predikat, primer porodičnog stabla.

Zadatak 3.5 Napisati Prolog predikate:

- `prestupna` koji određuje da li je godina prestupna
- `brdana` koji određuje koliko dana ima prosleđeni mesec

3.2.2 Zadaci za samostalni rad sa rešenjima

Zadatak 3.6 Napisati sledeće predikate:

a) `maksimum(A, B, M)` - određuje maksimum za dva broja `A` i `B`

b) `suma(N, S)` - za dati prirodan broj `N` računa sumu prvih `N` brojeva

c) `sumaParnih(N, S)` - za dati paran prirodan broj `N` računa sumu parnih brojeva od 2 do `N`

d) `proizvod(N, P)` - za dati prirodan broj `N` računa proizvod prvih `N` prirodnih brojeva

e) `proizvodNeparnih(N, P)` - za dati neparan prirodan broj `N` računa proizvod neparnih brojeva od 1 do `N`

f) `cifre(N)` - ispisuje cifre prirodnog broja `N` rečima

3.2.3 Zadaci za vežbu

Zadatak 3.7 Napisati sledeće predikate:

a) `sumaCifara(N, SC)` - određuje sumu cifara prirodnog broja `N`

b) `brojCifara(N, BC)` - određuje broj cifara prirodnog broja `N`

c) `maxCifra(N, MC)` - određuje maksimalnu cifru prirodnog broja `N`

d) `sumaKvadrata(N, SK)` - računa sumu kvadrata prvih `N` prirodnih brojeva

e) `fakt(N, F)` - računa faktorijel prirodnog broja `N`

g) `sumaDel(X, D)` - računa sumu pravih delilaca broja `X`

Zadatak 3.8 Ako su date činjenice oblika:

- `ucenik( SifraUcenika, ImeUcenika, Odeljenje)`
- `ocene( SifraUcenika, SifraPredmeta, Ocena)`
- `predmet( SifraPredmeta, NazivPredmeta, BrojCasova)`

Napisati sledeće predikate:

a) `bar2PeticeSifra(S)` - određuje šifru `S` učenika koji ima bar dve petice iz različitih predmeta

b) `bar2PeticeIme(X)` - određuje ime `X` učenika koji ima bar dve petice iz različitih predmeta

c) `odeljenjePetice(X,Y)` - određuje odeljenje `X` u kome postoje bar dve petice iz predmeta sa šifrom `Y`

Zadatak 3.9 Ako su date činjenice oblika:

- `film(NazivFilma, ZanrFilma, ImeReditelja, SifraGlumca)`
- `glumac(SifraGlumca, ImeGlumca, GodRodj, MestoRodj)`

Napisati sledeće predikate:

a) `filmskiUmetnik(X)` - `X` je filmski umetnik ako je reditelj nekog filma i igra u nekom filmu

b) `glumacBarDva(X)` - određuje ime glumca `X` koji igra u bar dva različita filma

c) `opstiGlumac(X)` - određuje ime glumca `X` koji igra u bar dva filma različitog žanra

d) `zanrovskiGlumac(X,Y)` - određuje ime glumca `X` koji igra u filmu žanra `Y`

3.3 Liste

3.3.1 Uvodni primeri

Zadatak 3.10 Osnovni pojmovi i predikati za rad sa listama.

3.3.2 Zadaci za samostalni rad sa rešenjima

Zadatak 3.11 Napisati sledeće predikate:

a) `dodajPocetak(X, L, NL)` - dodaje `X` na početak liste `L`

b) `dodajKraj(X, L, NL)` - dodaje `X` na kraj liste `L`

c) `obrisiPrvi(L, NL)` - briše prvi element, tj. glavu liste

d) `obrisiPoslednji(L, NL)` - briše poslednji element liste

e) `obrisi(X, L, NL)` - briše sva pojavljivanja elementa `X` u listi `L`

f) `obrisiPrvo(X, L, NL)` - briše samo prvo pojavljivanje elementa `X` u listi `L`

g) `obrisiK(L, K, NL)` - briše `K`-ti element liste `L`

Zadatak 3.12 Napisati predikat `podeli(L, L1, L2)` koji deli listu `L` na dve liste, listu pozitivnih elemenata `L1` i listu negativnih elemenata `L2`.

Zadatak 3.13 Napisati predikat `dupliraj(L, NL)` koji od date liste `L` formira novu listu `NL` tako što svaki negativan element duplira, tj. dva puta upisuje u novu listu.

Zadatak 3.14 Napisati predikat `zameni(X, Y, L, NL)` koji od date liste `L` formira novu listu `NL` zamenom elemenata `X` i `Y`.

Zadatak 3.15 Napisati predikat `pretvori(L, X)` koji za datu listu cifara `L` formira broj određen tim ciframa.

Zadatak 3.16 Napisati predikat `maxEl(L, X)` koji određuje maksimalni element liste `L`.

Zadatak 3.17 Napisati predikate za sortiranje liste rastuće:

a) `insertionSort(L, SL)` - insertion sort algoritam se zasniva na ubacivanju redom svakog elementa liste na svoje pravo mesto (mesto u sortiranoj listi)

b) `mergeSort(L, SL)` - merge sort algoritam se zasniva na dekompoziciji liste, tj. listu delimo na dva jednaka dela, te delove sortiramo i posle toga ih objedinjujemo

3.3.3 Zadaci za vežbu

Zadatak 3.18 Napisati predikat `parNepar(L, L1, L2)` koji deli listu `L` na dve liste, listu parnih elemenata `L1` i listu neparnih elemenata `L2`.

Zadatak 3.19 Napisati predikat `podeli(L, N, L1, L2)` koji deli listu `L` na dve liste `L1` i `L2`, pri čemu je zadata dužina prve liste `L1`.

Zadatak 3.20 Napisati predikat `ogledalo(L1, L2)` koji proverava da li je lista `L1` jednaka obrnutoj listi liste `L2`.

Zadatak 3.21 Napisati predikat `interval(X, Y, L)` koji kreira listu `L` koja sadži sve cele brojeve iz intervala zadatog sa prva dva argumenta.

Zadatak 3.22 Napisati predikat `skalar(L1, L2, S)` koji određuje skalarni proizvod dva vektora, tj. listi brojeva `L1` i `L2`.

Zadatak 3.23 Napisati predikat `sortirana(L)` koji proverava da li je lista `L` sortirana, bilo opadajuće ili rastuće.

Zadatak 3.24 Napisati predikat `spoji(L1, L2, L)` koji spaja dve rastuće sortirane liste `L1` i `L2` u treću tako da i ona bude sortirana rastuće.

3.4 Razni zadaci

3.4.1 Zadaci sa rešenjima

Zadatak 3.25 Napisati predikate `nzd(N, M, NZD)` i `nzs(N, M, NZS)` koji određuju najveći zajednički delilac i najmanji zajednički sadržalac prirodnih brojeva `N` i `M` redom.

Zadatak 3.26 Ako su date činjenice oblika:

- `stan(Porodica, KvadraturaStana)`
- `clan(Porodica, BrojClanova)`

Napisati predikat `poClanu(Porodica, Prosek)` koji određuje prosečan broj kvadrata stana po članu porodice koja živi u njemu.

Zadatak 3.27 Ako je data baza znanja:

- `automobil(SifraAutomobila, NazivAutomobila)`
- `vlasnik(ImeVlasnika, SifraAutomobila)`
- `brziSifra(SX, SY)` - automobil šifre `SX` je brži od automobila šifre `SY`

Napisati predikate:

a) `brziNaziv(X, Y)` - automobil naziva `X` je brži od automobila naziva `Y`

b) `imaAutomobil(X)` - `X` je vlasnik nekog automobila

c) `imaBrzi(X, Y)` - `X` je vlasnik bržeg automobila od onog čiji je vlasnik `Y`.

Zadatak 3.28 Napisati predikat `savrsen(N)` koji proverava da li je prirodan broj `N` savršen, tj. da li je jednak sumi svojih pravih delilaca. U slučaju da se prosledi neispravan argument, predikat treba da ispiše poruku o grešci i prekine program.

Zadatak 3.29 Napisati predikat `izbaci3(N, X)` koji iz prirodnog broja `N` izbacuje sve cifre manje 3. U slučaju da se prosledi neispravan argument, predikat treba da prekine program.

Zadatak 3.30 Napisati predikate za liste brojeva:

a) `duplikati(L, L1)` - izbacuje duplikate iz liste `L`

b) `unija(L1, L2, L)` - određuje uniju listi `L1` i `L2`

c) `presek(L1, L2, L)` - određuje presek listi `L1` i `L2`

d) `razlika(L1, L2, L)` - određuje razliku listi `L1` i `L2`

Zadatak 3.31 Napisati program koji rešava sledeću zagonetku. Postoji pet kuća, svaka različite boje u kojoj žive ljudi različitih nacionalnosti koji piju različita pića, jedu različita jela i imaju različite kućne ljubimce. Važi sledeće:

- Englez živi u crvenoj kući
- Španac ima psa
- kafa se pije u zelenoj kući

- Ukrajinac pije čaj
- zelena kuća je odmah desno uz belu
- onaj koji jede špagete ima puža
- pica se jede u žutoj kući
- mleko se pije u srednjoj kući
- Norvežanin živi u prvoj kuci s leva
- onaj koji jede piletinu živi pored onoga koji ima lisicu
- pica se jede u kući koja je pored kuće u kojoj je konj
- onaj koji jede brokoli pije sok od narandze
- Japanac jede suši
- Norvežanin živi pored plave kuće

Čija je zebra, a ko pije vodu?

Zadatak 3.32 Napisati program koji rešava sledeću zagonetku. Svakog vikenda, Milan čuva petoro komšijske dece. Deca se zovu Kata, Lazar, Marko, Nevenka i Ognjen, a prezivaju Filipović, Grbović, Hadžić, Ivanović i Janković. Svi imaju različit broj godina od dve do šest. Važi sledeće:

- jedno dete se zove Lazar Janković
- Kata je godinu dana starija od deteta koje se preziva Ivanović koje je godinu dana starije od Nevenke
- dete koje se preziva Filipović je tri godine starije od Marka
- Ognjen je duplo stariji od deteta koje se preziva Hadžić

Kako se ko zove i koliko ima godina?

3.4.2 Zadaci za vežbu

Zadatak 3.33 Napisati predikat `uzastopni(X, Y, Z, L)` koji proverava da li su prva tri argumenta uzastopni elementi u listi `L`.

Zadatak 3.34 Napisati predikat `kompresuj(L, KL)` koji u datoj listi `L` eliminiše uzastopne duplikate.

Zadatak 3.35 Napisati predikat `prefiksi(L, P)` koji određuje sve liste koje su prefiksi date liste `L`.

Zadatak 3.36 Napisati predikat `sufiksi(L, S)` koji određuje sve liste koje su sufiksi date liste `L`.

Zadatak 3.37 Napisati predikat `opadajuce(N, L)` koji za dat prirodan broj `N` formira listu brojeva od `N` do `1`.

Zadatak 3.38 Napisati predikat `form(N, L)` kojim se formira lista od prirodnih brojeva deljivih sa 5 i manjih od datog prirodnog broja `N`.

Zadatak 3.39 Napisati program koji rešava sledeću zagonetku. Četiri žene se zovu Petra, Milica, Lenka i Jovana, a prezivaju Perić, Mikić, Lazić i Jović. One imaju četiri kćerke koje se takodje zovu Petra, Milica, Lenka i Jovana. Važi sledeće:

- nijedna majka nema prezime koje počinje istim slovom kao ime
- nijedna kćerka nema prezime koje počinje istim slovom kao ime

- nijedna kćerka se ne zove kao majka
- majka koja se preziva Perić se zove isto kao Miličina kćerka
- Lenkina kćerka se zove Petra

Odrediti imena majki i kćerki.

Zadatak 3.40 Napisati program koji rešava sledeću zagonetku. Četiri para je došlo na maskenbal:

- Markova zena se maskirala kao macka
- dva para su stigla pre Marka i njegove žene, a jedan muskarac je bio maskiran u medveda
- prvi koji je stigao nije bio Vasa, ali je stigao pre onoga koji je bio maskiran u princa
- žena maskirana u vešticu (nije Bojana) je udata za Peru, koji se maskirao kao Paja patak
- Marija je došla posle Laze, a oboje su stigli pre Bojane
- žena maskirana u Ciganku je stigla pre Ane, pri čemu nijedna od njih nije udata za muškarca maskiranog u Betmena
- žena maskirana u Snežanu je stigla posle Ivane

Odrediti kako je bio obučen koji par.

Zadatak 3.41 Izračunavanje vrednosti aritmetičkog izraza korišćenjem listi možete pogledati ovde:
https://rosettacode.org/wiki/Arithmetic_evaluation#Prolog

Zadatak 3.42 Implementacije raznih problema u Prologu možete pogledati ovde:
https://rosettacode.org/wiki/Category:Prolog

4

Programiranje ograničenja - Prolog i Python

Potrebno je imati distribuciju Prologa pod nazivom BProlog i instaliran Python 3 sa bibliotekom python-constraint. Na Ubuntu operativnom sistemu, biblioteka python-constraint se može instalirati narednom komandom (python3 zameniti aliasom koji koristite na svom računaru za Python): `sudo -H python3 -m 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/

4.1 Programiranje ograničenja

4.1.1 Uvodni primeri

Zadatak 4.1 Osnovni pojmovi i postavka problema.

4.1.2 Zadaci za samostalni rad sa rešenjima

Zadatak 4.2 Napisati program koji pronalazi petocifren broj `ABCDE` za koji je izraz `A+2*B-3*C+4*D-5*E` minimalan i `A`, `B`, `C`, `D` i `E` su različite cifre.

Zadatak 4.3 Dati su novčići od 1, 2, 5, 10, 20 dinara. Napisati program koji pronalazi sve moguće kombinacije tako da zbir svih novčića bude 50 i da se svaki novčič pojavljuje bar jednom u kombinaciji.

Zadatak 4.4 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 4.5 Napisati program koji pronalazi sve vrednosti promenljivih X, Y, 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 $Z \in \{1, 10, 20, ..., 100\}$

Zadatak 4.6 Napisati program koji dodeljuje različite vrednosti različitim karakterima tako da suma bude zadovoljena:

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

Zadatak 4.7 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 $W \in \{1, 8, 15, 22, ..., 1000\}$.

Zadatak 4.8 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 4.9 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 4.10 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 4.11 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.

4.1.3 Zadaci za vežbu

Zadatak 4.12 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 4.13 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 4.14 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 4.15 Uraditi sve zadatke koji su pobrojani ovde:
http://www.primepuzzle.com/leeslatest/alphameticpuzzles.html

Zadatak 4.16 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 4.17 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 4.18 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?