Programski jezici i paradigme

Zbirka zadataka

Ivan Ristović
ivan.ristovic@math.rs

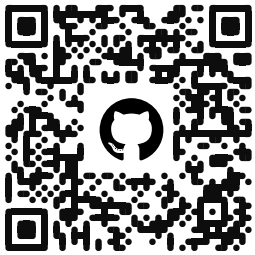
Milan Čugurović
milan.cugurovic@math.rs

Milena Vujošević Janičić
milena.vujosevic.janicic@math.rs

Sadržaj

Glava 1

Komponentno programiranje



Uvod

Komponentno programiranje predstavlja proširenje objektno orijentisane paradigme sa ciljem poboljšanja razdvojenosti logike različitih delova softverskog rešenja. Delovi, tj. komponente, su odvojene samostalne jedinice koda čijim se vezivanjem gradi složeniji sistem. Komponente se dizajniraju tako da se mogu razvijati nezavisno jedne od drugih, sa unapred dogovorenim interfejsom koji se održava tokom razvoja. Komponentno programiranje je veoma popularno u kontekstu veb i mikroservisnih aplikacija, gde se, na primer, izgled aplikacije dizajnira nezavisno od ostatka aplikacije, i gradi se komponentama grafičkog interfejsa.

Konvencije

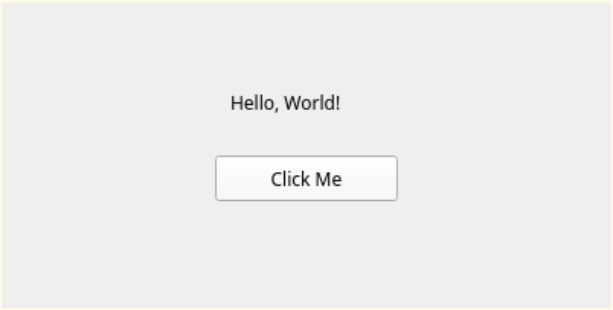
Ovo poglavlje uvodi komponentnu programsku paradigmu kroz jednostavne zadatke koji za cilj imaju dizajn aplikacija koji se sastoji od najmanje dva dela — *prednjeg* (eng. *frontend*) dela korisničkog interfejsa i *zadnjeg* (eng. *backend*) dela koja implementira logiku aplikacije. Delovi aplikacije se razvijaju nezavisno, uz dogovoreni interfejs i interakcije između njih. Međusloj aplikacije definiše veze između prednjeg i zadnjeg dela povezivanjem elemenata prednjeg dela sa odgovarajućim akcijama zadnjeg dela. Prednji i zadnji deo mogu biti posmatrani kao komponente ili kao skupovi komponenata koje međusobno komuniciraju. Elementi korisničkog interfejsa se najčešće sklapaju od gotovih komponenti[a].

[a] Na primer, popularni radni okvir *React* se najvećim delom zasniva na komponentnom programiranju.

1.1 Osnovni elementi korisničkog interfejsa

Zadatak 1.1.1: *Hello, World!*

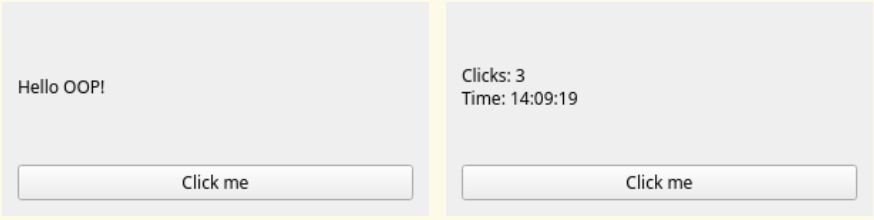
Implementirati program sa komponentom *Label* koja prikazuje tekst *Hello, World!* i dugmetom sa tekstom `Click Me`. Raspored komponenti je prikazan na Slici 1.1. Klikom na dugme `Click Me` se zatvara prozor.



Slika 1.1: Aplikacija *Hello World!*

Zadatak 1.1.2: *Hello, OOP!*

Implementirati program sa komponentom *Label* koja prikazuje tekst `Hello OOP!` i dugmetom sa tekstom `Click Me`, sa izgledom kao na Slici 1.2a. Klikom na dugme `Click Me` se tekst labele menja tako da prikazuje broj klikova dugmeta i vreme poslednjeg klika, kao na Slici 1.2b.

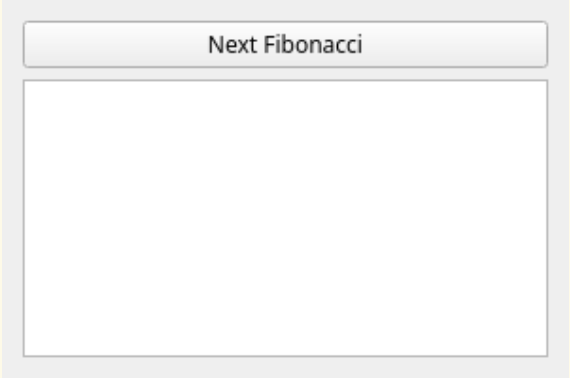


(a) (b)

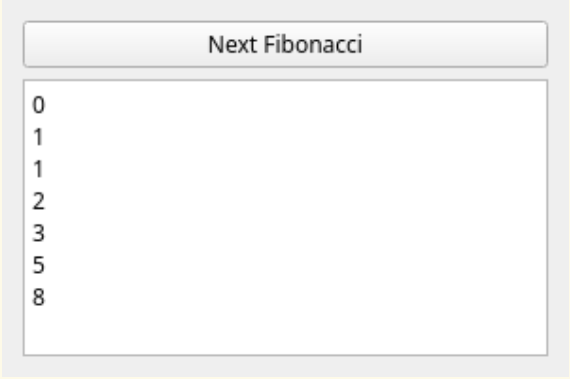
Slika 1.2: Aplikacija *Hello OOP!*

Zadatak 1.1.3: ***Generator Fibonačijevih brojeva***

Implementirati generator Fibonačijevih brojeva sa izgledom kao na Slici 1.3. Aplikacija se sastoji od dugmeta `Next Fibonacci` i polja za čitanje (*read-only*) sa rasporedom kao na slici 1.3a. Klikom na dugme `Next Fibonacci` se generiše naredni Fibonačijev broj i dodaje u polje za čitanje, kao na Slici 1.3b.



(a)

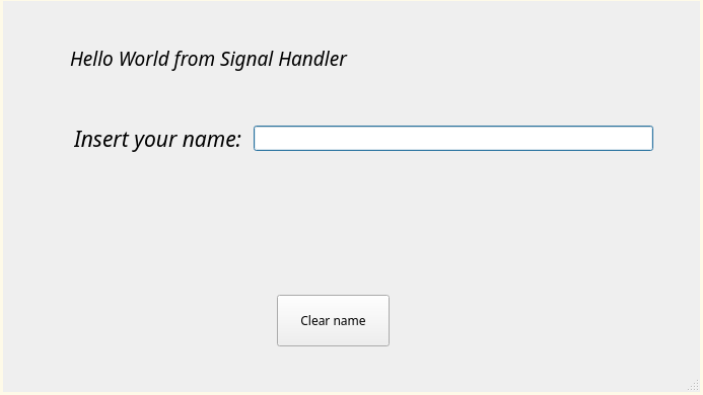


(b)

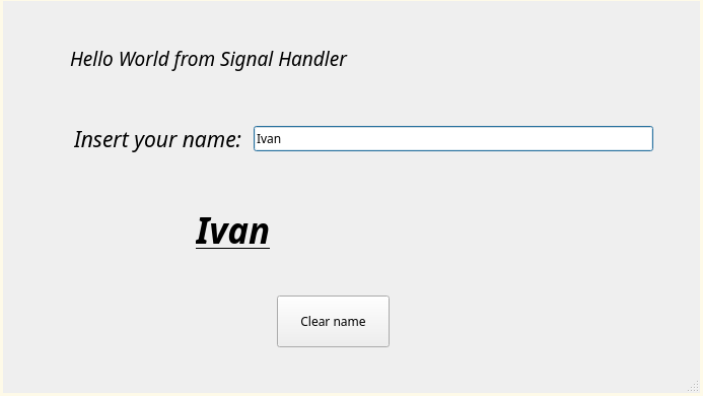
Slika 1.3: Generator Fibonačijevih brojeva

Zadatak 1.1.4: ***Hello World from Signal Handler***

Implementirati program sa rasporedom komponenti kao na Slici 1.4a. U jednolinijsko polje za unos se unosi tekst koji aplikacija ispisuje u labelu ispod jednolinijskog polja za unos koja se ažurira u realnom vremenu kako se tekst u polju za unos menja (Slika 1.4b). Klikom na dugme `Clear name` se briše tekst unet u polje za unos i izgled aplikacije se vraća na početni izgled (Slika 1.4a).



(a)

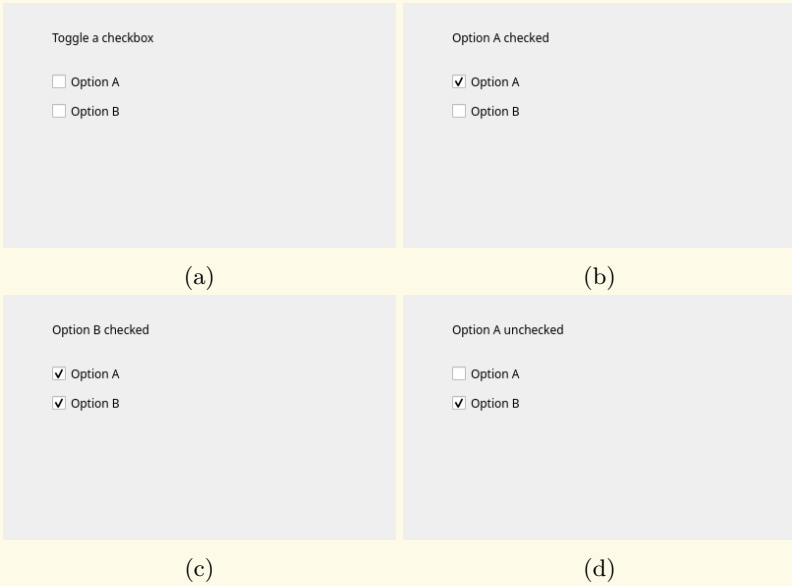


(b)

Slika 1.4: Aplikacija *Hello World from Signal Handler*

Zadatak 1.1.5: ***Checkbox toggle***

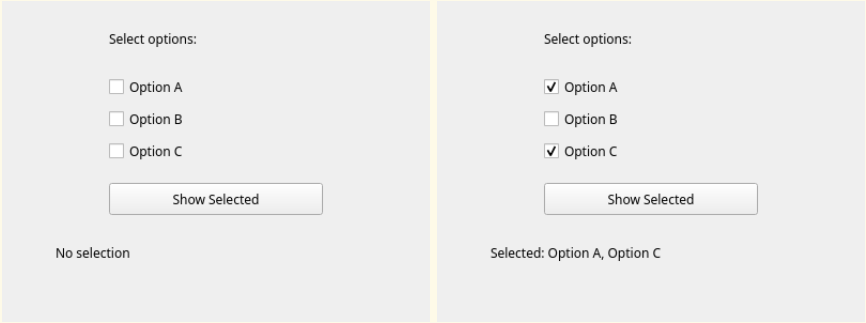
Implementirati program *Checkbox toggle* sa dve komponente za odabir kao na Slici 1.5a. Odabirom opcija se ažurira tekst labele tako da opisuje poslednju interakciju. Na primer, prilikom odabira opcije `A` (Slika 1.5b), tekst se ažurira na `Option A checked`. Prilikom odabira opcije `B` (Slika 1.5c), tekst se ažurira na `Option B checked`. Prilikom uklanjanja odabira, tekst se ažurira na `Option A unchecked` (Slika 1.5d), tj. `Option B unchecked`.



Slika 1.5: Aplikacija *Checkbox toggle*

Zadatak 1.1.6: *Multibox toggle*

Implementirati program *Multibox toggle* sa rasporedom komponenti kao na Slici 1.6a. Klikom na dugme `Show Selected` se prikazuju odabrane opcije redom, kao na Slici 1.6b.

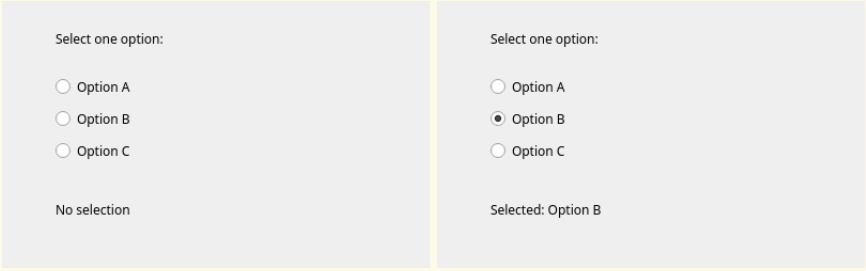


(a) (b)

Slika 1.6: Aplikacija *Multibox toggle*

Zadatak 1.1.7: *Radio toggle*

Implementirati program *Radio toggle* sa rasporedom komponenti za jedinstveni odabir kao na Slici 1.7a. Prilikom odabira neke od opcija, tekst labele se ažurira tako da prikaže odabranu opciju, kao na slici 1.7.

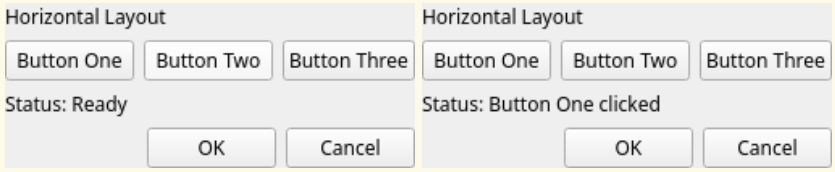


(a) (b)

Slika 1.7: Aplikacija *Radio toggle*

Zadatak 1.1.8: *Horizontal layout*

Implementirati program *Horizontal layout* sa horizontalnim rasporedom komponenti kao na Slici 1.8. Klikom na bilo koje od tri dugmeta se ažurira tekst labele po uzoru na Sliku 1.8b. Klikom na dugme `OK`, aplikacija uspešno završava izvršavanje, dok klikom na dugme `Cancel` aplikacija neuspešno završava izvršavanje.



(a) (b)

Slika 1.8: Aplikacija *Horizontal layout*

Zadatak 1.1.9: *Grid layout*

Implementirati program *Grid layout* sa rasporedom komponenti kao na Slici 1.9. Klikom na bilo koje dugme se ispisuje identifikator dugmeta na standardni izlaz kao u primeru ispod.

Slika 1.9: Aplikacija *Grid layout*

Primer izlaza, u slučaju gde su dugmad kliknuta redom od broja 1 do broja 5:

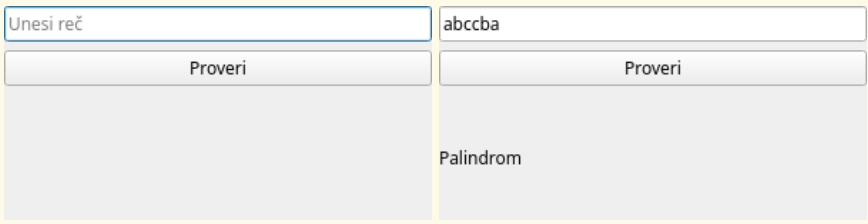
Primer 1

```
Izlaz: 1
       2
       3
       4
       5
```

1.2 Interaktivne komponente

Zadatak 1.2.1: *Palindrom*

Implementirati aplikaciju sa rasporedom komponenti i izgledom kao na Slici 1.10a, koja testira da li je uneta reč palindrom klikom na dugme `Proveri`. Prikazati rezultat po uzoru na Sliku 1.10b.



(a) (b)

Slika 1.10: Aplikacija *Palindrom*

Zadatak 1.2.2: *Štoperica*

Implementirati štopericu sa rasporedom komponenti i izgledom kao na Slici 1.11a. Rezultat prikazati po uzoru na sliku 1.11b.

(a) (b)

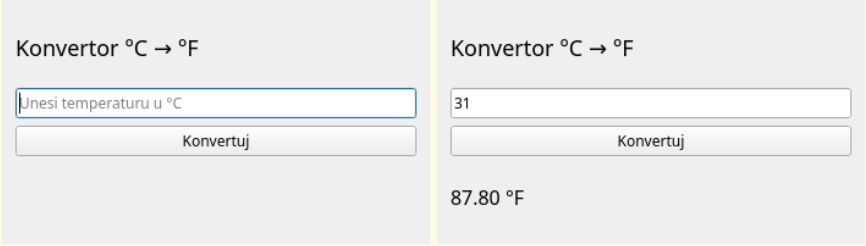
Slika 1.11: Aplikacija *Štoperica*

Zadatak 1.2.3: ***Konverter temperature***

Implementirati konvertor temperature sa rasporedom komponenti i izgledom kao na Slici 1.12a. Temperatura se unosi u stepenima Celzijusa i konvertuje u stepene Farenhajta klikom na dugme `Konvertuj`. Rezultat prikazati po uzoru na Sliku 1.12b.
Formula za konverziju:

$$F = C \cdot \frac{9}{5} + 32$$

.



(a) (b)

Slika 1.12: Aplikacija *Konverter temperature*

Zadatak 1.2.4: ***Brojač klikova***

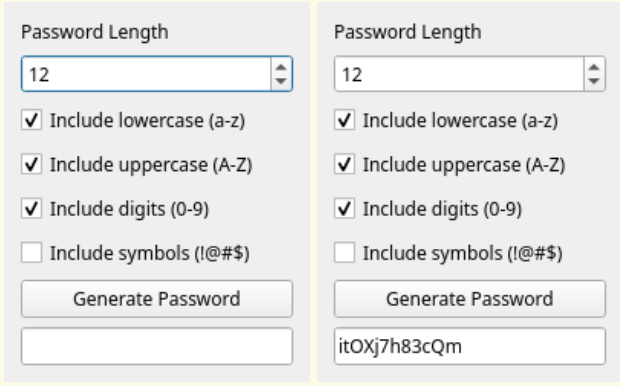
Implementirati aplikaciju sa rasporedom komponenti i izgledom kao na Slici 1.13a. Svaki klik dugmeta `Klikni me` povećava brojač ispisan u labeli iznad dugmeta. Posle 20 klikova dugme postaje onemogućeno i tekst `Dovoljno!` se ispisuje u labeli ispod dugmeta (Slika 1.13b).

(a) (b)

Slika 1.13: Aplikacija *Brojač klikova*

Zadatak 1.2.5: *Generator lozinki*

Implementirati generator lozinki sa rasporedom komponenti i izgledom kao na Slici 1.14a. Klikom na dugme `Generate Password` se ispisuje nasumično generisana lozinka u labeli ispod dugmeta po uzoru na Sliku 1.14b.

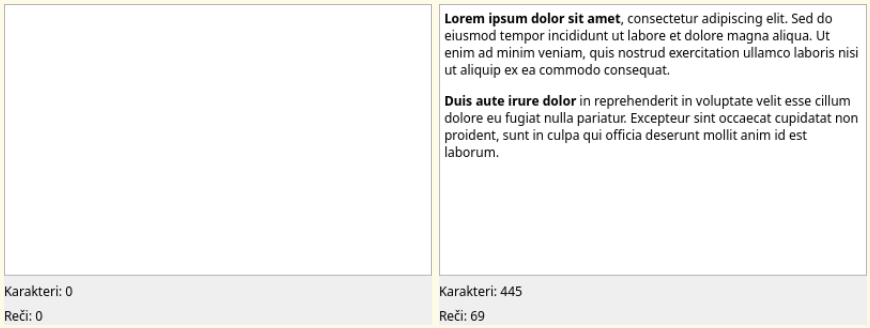


(a) (b)

Slika 1.14: Aplikacija *Generator lozinki*

Zadatak 1.2.6: *Analizator teksta*

Implementirati aplikaciju sa rasporedom komponenti i izgledom kao na Slici 1.15a, koja broji karaktere i reči u unetom tekstu i ispisuje rezultat po uzoru na Sliku 1.15b. Aplikacija treba da podržava bogati unos teksta (*rich text*), uključujući masna (*bold*), kurziv (*italic*), i podvučena slova (*underline*).

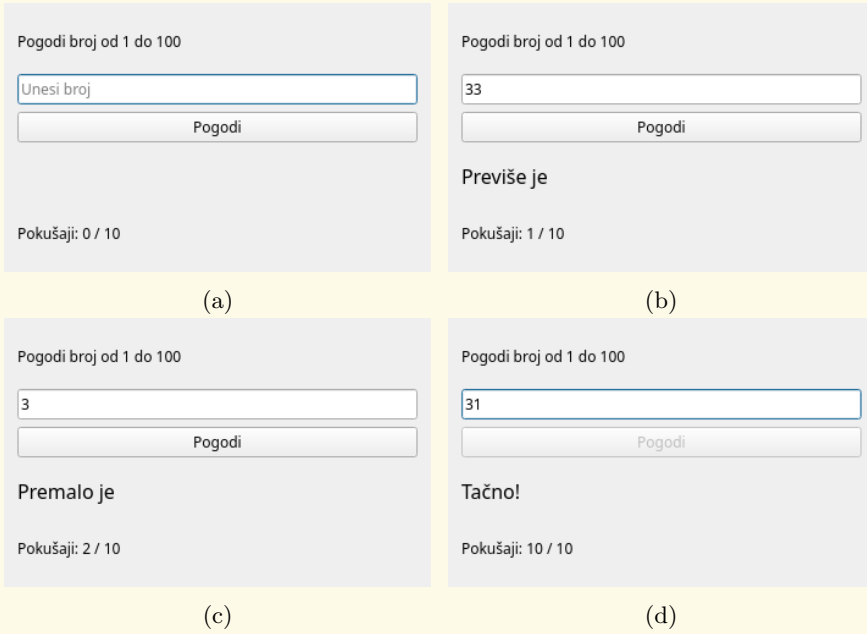


(a) (b)

Slika 1.15: Aplikacija *Analizator teksta*

Zadatak 1.2.7: ***Igra „Pogodi broj”***

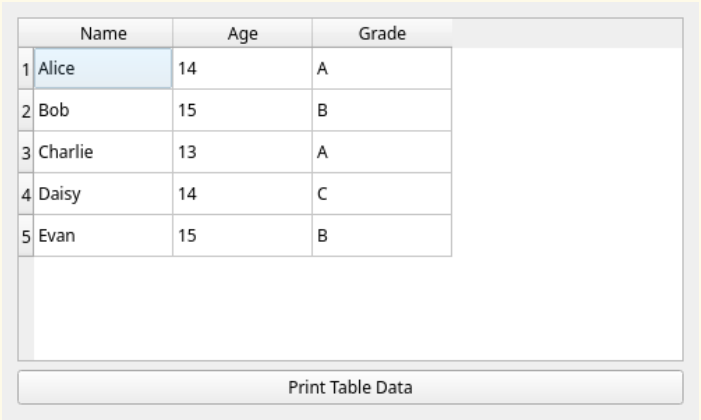
Implementirati igricu *Pogodi broj* sa rasporedom komponenti i izgledom kao na Slici 1.16a. Aplikacija određuje nasumično jedan broj u intervalu $[1, 100]$ kao zamšljeni broj. Korisnik unosi broj u polje za unos i klikom na dugme `Pogodi` se proverava da li je uneti broj jednak zamišljenom. Pri svakom pokušaju, aplikacija odgovara sa `Previše je` (Slika 1.16b), `Premalo je` (Slika 1.16c), ili `Tačno!` (Slika 1.16d) u labeli ispod dugmeta. Igra prestaje ukoliko korisnik pogodi broj ili ukoliko iskoristi 10 pokušaja bez pogotka. Broj neuspelih pokušaja se ispisuje u labeli na dnu aplikacije.



Slika 1.16: Igrica *Pogodi broj*

Zadatak 1.2.8: ***Tabelarni editor***

Implementirati aplikaciju za tabelarni unos podataka o studentima sa izgledom kao na Slici 1.17. Klikom na dugme `Print Table Data` se ispisuje stanje tabele na standardni izlaz u formatu *CSV*. Omogućiti unos i izmenu podataka u realnom vremenu.



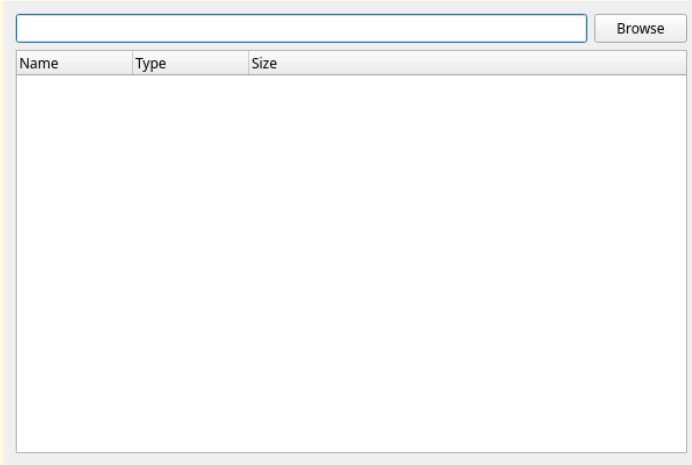
Slika 1.17: Aplikacija *Tabelarni editor*

Primer 1

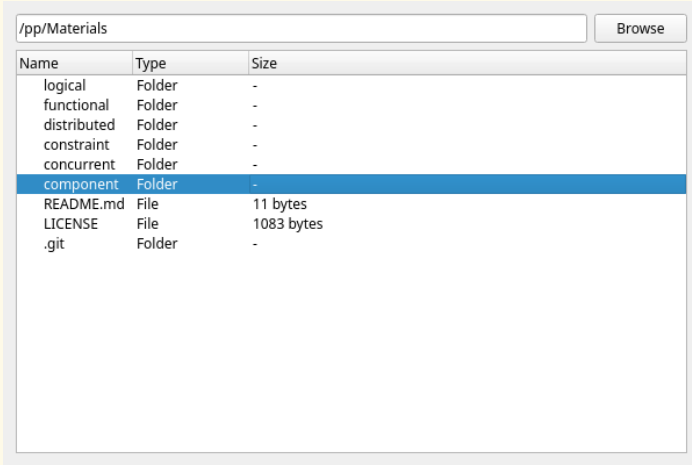
```
Izlaz: Name, Age, Grade
       'Alice', '14', 'A'
       'Bob', '15', 'B'
       'Charlie', '13', 'A'
       'Daisy', '14', 'C'
       'Evan', '15', 'B'
```

Zadatak 1.2.9: *Navigator fajlsistema*

Implementirati aplikaciju za navigaciju fajlsistema sa izgledom kao na Slici 1.18a. Klikom na dugme `Browse` se otvara dijalog za odabir početnog direktorijuma. Nakon odabira, sadržaj odabranog direktorijuma se prikazuje u glavnom polju za prikaz po uzoru na Sliku 1.18b.



(a)

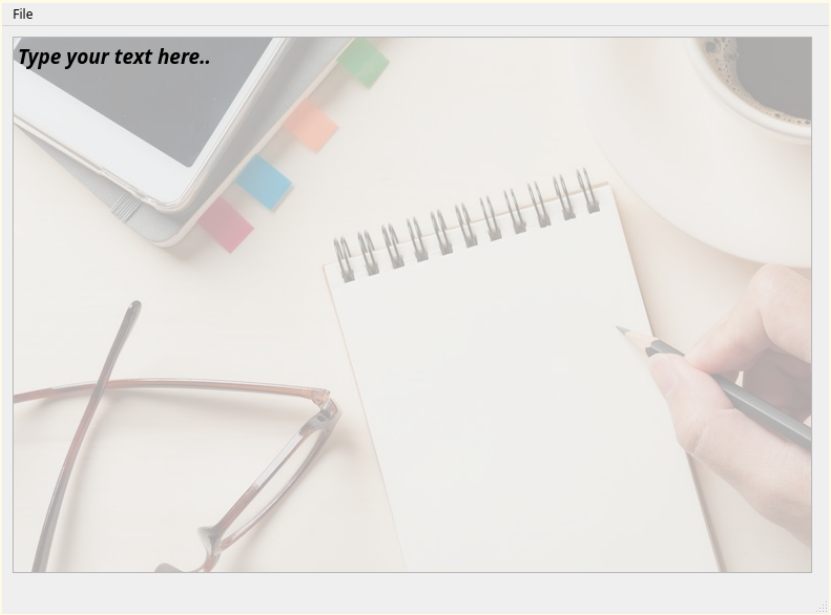


(b)

Slika 1.18: Aplikacija *Navigator fajlsistema*

Zadatak 1.2.10: *„Simple Pad" editor*

Napisati program koji implementira tekstualni editor sa nazivom *SimplePad*, sa rasporedom komponenti i izgledom kao na Slici 1.19. *SimplePad* treba da podržava bogati unos teksta (*rich text*), uključujući masna (*bold*), kurziv (*italic*), i podvučena slova (*underline*). *SimplePad* treba da pruži osnovne komande (kreiranja, otvaranja i čuvanja dokumenata), kao i opcije kopiranja (`Copy`), isecanja (`Cut`), i lepljenja (`Paste`) delova teksta. U padajućem meniju `File` se nalaze osnovne komande, dok se prečice `Ctrl+C`, `Ctrl+X`, i `Ctrl+V` koriste za odabir opcija kopiranja, isecanja, i lepljenja teksta, redom.

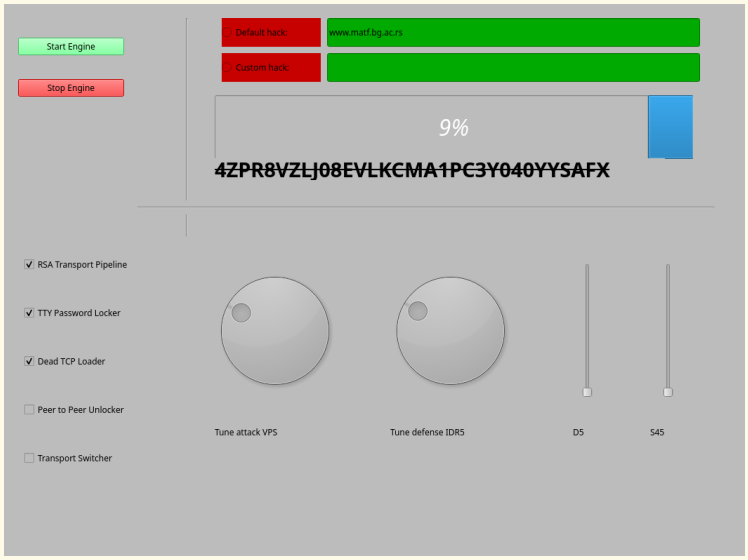


Slika 1.19: Aplikacija *SimplePad*

Zadatak 1.2.11: *HackSim*

Napisati aplikaciju *HackSim* koja simulira hakerske napade, sa izgledom kao na Slici 1.20. Aplikacija klikom na dugme `Start Engine` ažurira progres bar (od 0 do 100%), svake sekunde napredujući za po jedan procenat. Dodatno, aplikacija podržava odabir dodatnih opcija, kao i prodešavanje kontinualnih parametara.

(a)

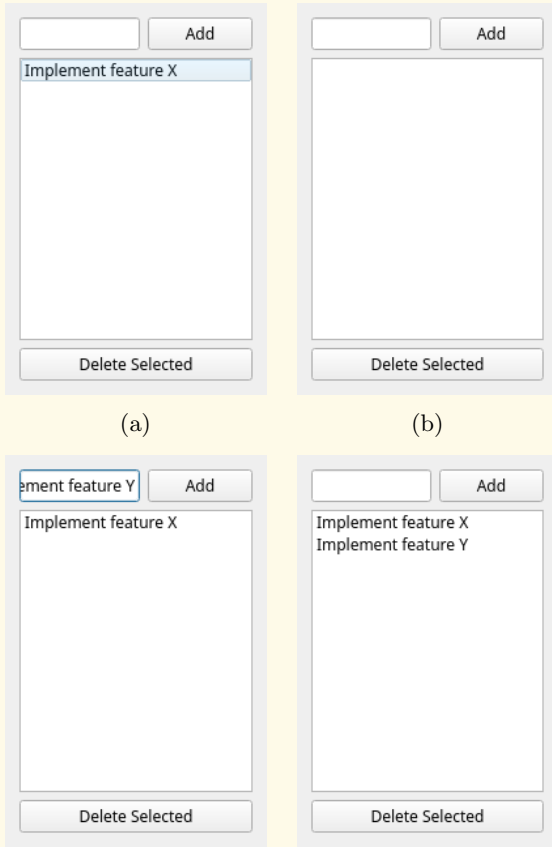


(b)

Slika 1.20: Aplikacija *HackSim*

Zadatak 1.2.12: *„TODO" aplikacija*

Implementirati aplikaciju *TodoApp* za održavanje liste zadataka sa izgledom kao na Slici 1.21a. Klikom na dugme `Add` se u listu dodaje zadatak sa opisom unetim u polje za unos teksta (Slika 1.21b). Aplikacija prikazuje sve zadatke u glavnom polju za prikaz. Klikom na dugme `Delete Selected` se briše odabrani zadatak iz liste zadataka (Slika 1.21c prikazuje odabir stavke za brisanje, dok Slika 1.21d prikazuje stanje nakon brisanja odabrane stavke). Aplikacija čuva listu zadataka u fajlu `tasks.json`, u formatu *JSON*, sa sadržajem kao u primeru ispod.



Slika 1.21: Aplikacija *TodoApp*

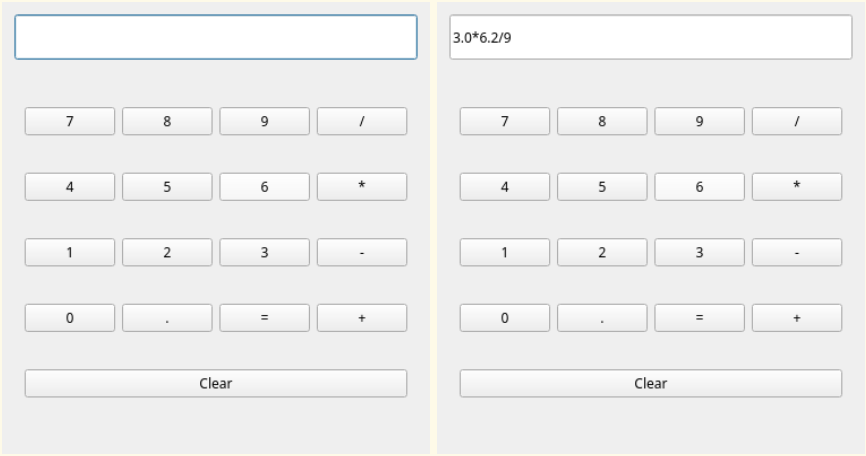
Primer 1

DATOTEKA tasks.json:

```
["Implement feature X", "Review feature Y"]
```

Zadatak 1.2.13: *Kalkulator*

Implementirati kalkulator sa rasporedom komponenti i izgledom kao na Slici 1.22a. Kalkulator podržava osnovne aritmetičke operacije sabiranja, oduzimanja, množenja i deljenja (Slika 1.22b). Klikom na `Clear` dugme se resetuje stanje kalkulatora.

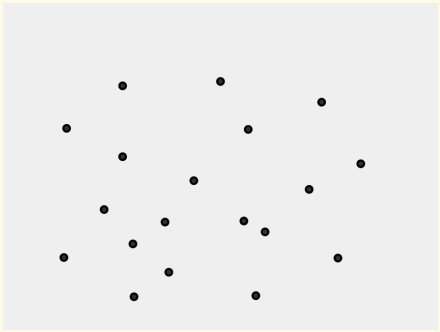


(a) (b)

Slika 1.22: Aplikacija *Kalkulator*

Zadatak 1.2.14: *Igra „Bullet Hole”*

Implementirati igricu *BulletHole*[a] sa izgledom kao na Slici 1.23, koja omogućava igraču da, klikom na površinu unutar aplikacije, ispaljuje *metke*. Svaki klik kreira *metak* simuliran slikom rupe metka učitanom iz vektorskog fajla `hole.svg` sa diska.



Slika 1.23: Igrica *BulletHole*

[a]Igrica *BulletHole* je inspirisana popularnom igricom *Desktop Destroyer*.

Zadatak 1.2.15: *Igra ,,Brzi refleksi"*

Implementirati igru *Brzi refleksi* sa rasporedom komponenti i izgledom kao na Slici 1.24a. Igrica prikazuje dugme koje menja svoju poziciju svakih 0.5 sekundi. Korisnik treba da klikne dugme pre nego što ono promeni poziciju. Finalni skor se računa kao broj uspešnih klikova u 10 sekundi (Slika 1.24b).

(a) (b)

Slika 1.24: Igra *Brzi refleksi*

Zadatak 1.2.16: *Igra ,,Vešala"*

Napisati mini desktop aplikaciju koja predstavlja najjednostavniju verziju igre ,,Vešala" (eng. *Hangman*). Aplikacija treba da zadovolji sledeće uslove:

1. **Nasumičan izbor reči**
   Program treba da ima unapred definisanu listu reči, od koje se pri pokretanju nasumično bira jedna reč koja se pogađa.

2. **Prikaz reči**
   Nepogođena slova treba da budu prikazana kao donje crte (,,_"), a pogođena slova se u njima zamenjuju odgovarajućim slovima.

3. **Unos slova**
   Korisnik unosi jedno slovo u polje za unos i potvrđuje pritiskom na dugme „Pogodi".

4. **Provera pogodaka**
   Ako uneseno slovo postoji u reči, program ažurira prikaz reči. Ako slovo ne postoji, brojač grešaka se povećava.

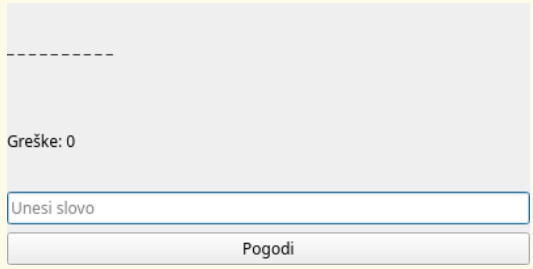
5. **Ograničen broj grešaka**
   Dozvoljeno je najviše 6 pogrešnih pokušaja. Nakon šest grešaka, igra se završava porukom o porazu i prikazuje se tačna reč.

6. **Pobeda**
   Kada su sva slova pogođena, program prikazuje poruku o pobedi.

7. **Jednostavan interfejs**
   Interfejs treba da sadrži:

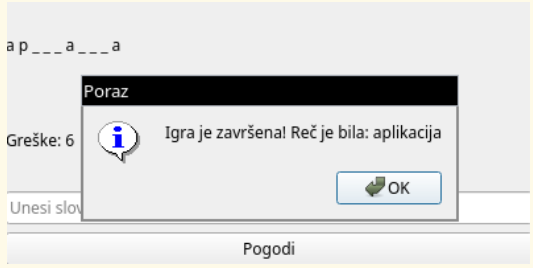
   - prikaz trenutnog stanja reči
   - prikaz broja pogrešnih pokušaja
   - polje za unos slova
   - dugme „Pogodi“

   Primer interfejsa je prikazan na Slici 1.25.



(a)

(b)

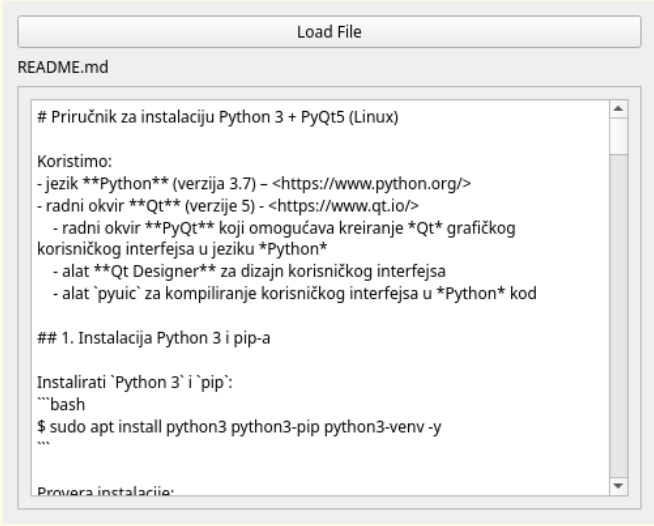


(c)

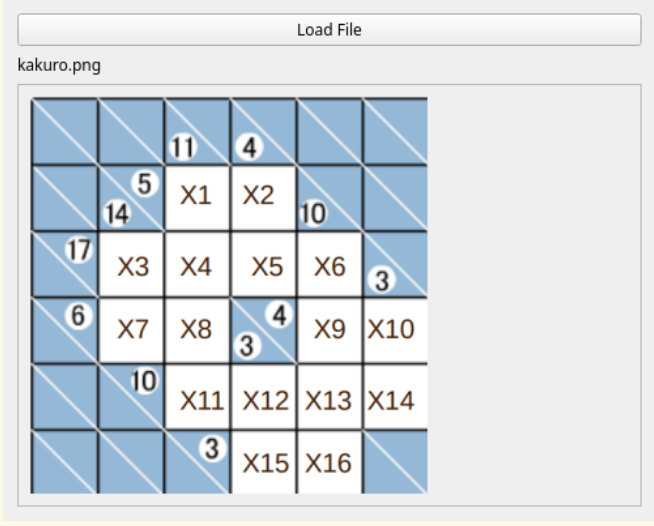
Slika 1.25: Igra „*Vešala*”

Zadatak 1.2.17: ***Pregledač tekstualnih fajlova i slika***

Implementirati pregledač tekstualnih fajlova i slika sa rasporedom komponenti i izgledom kao na Slici 1.26. Pregledač prikazuje fajlove u polju za prikaz. Tekstualni fajlovi (Slika 1.26a) se prikazuju tako što se njihov sadržaj ispiše u polje za prikaz. Slike (Slika 1.26b) se prikazuju unutar polja za prikaz.



(a)

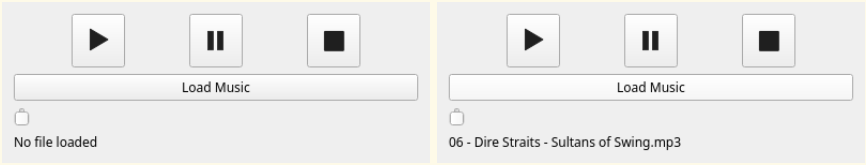


(b)

Slika 1.26: Pregledač tekstualnih fajlova i slika

Zadatak 1.2.18: ***Muzički plejer***

Implementirati muzički plejer sa rasporedom komponenti i izgledom kao na Slici 1.27.



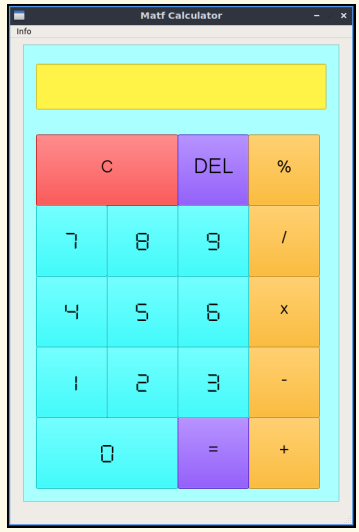
(a) (b)

Slika 1.27: Muzički plejer

1.3 Ispitni zadaci

Zadatak 1.3.1: ***MATF kalkulator***

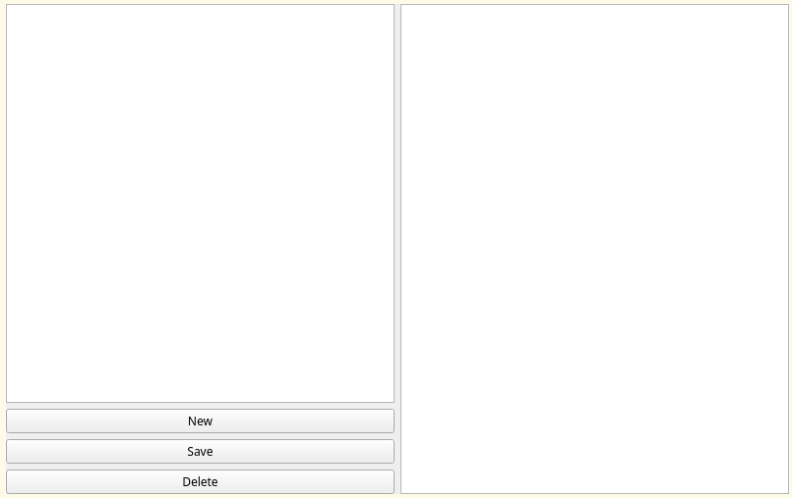
Implementirati aplikaciju *Matf Calculator* sa rasporedom komponenti i izgledom kao na Slici 1.28. *Matf Kalkulator* podržava osnovne aritmetičke operacije sabiranja, oduzimanja, množenja i deljenja. Implementirati brisanje unetih brojeva (*back* dugme) kao i resetovanje rada kalkulatora klikom na odgovarajuće dugme.



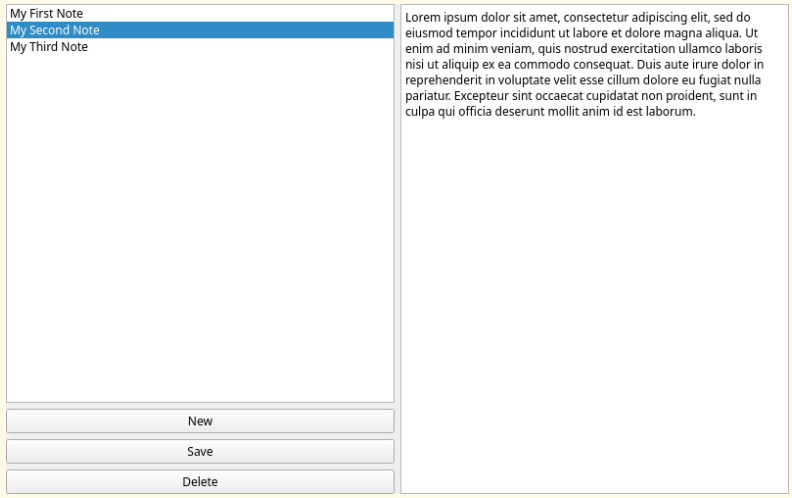
Slika 1.28: Aplikacija *MATF Calculator*

Zadatak 1.3.2: *Beleške*

Implementirati aplikaciju za čuvanje beleški sa izgledom kao na Slici 1.29. Aplikacija čuva beleške u memoriji. Dugme `Add` dodaje novu belešku u spisak beleški, sa imenom koje se unosi u iskačući dijalog za unos teksta. Dugme `Save` ažurira tekst odabrane beleške (u protivnom, izmene trenutne beleške se gube nakon odabira druge beleške iz spiska). Dugme `Delete` briše odabranu belešku iz spiska.



(a)

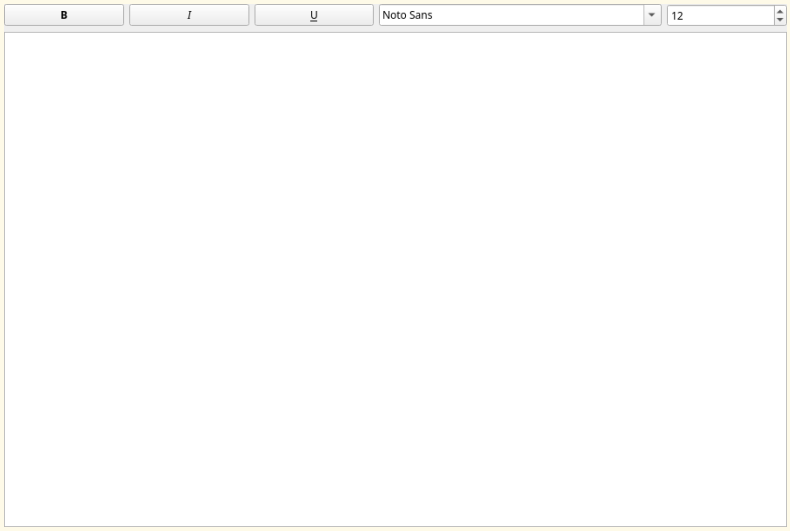


(b)

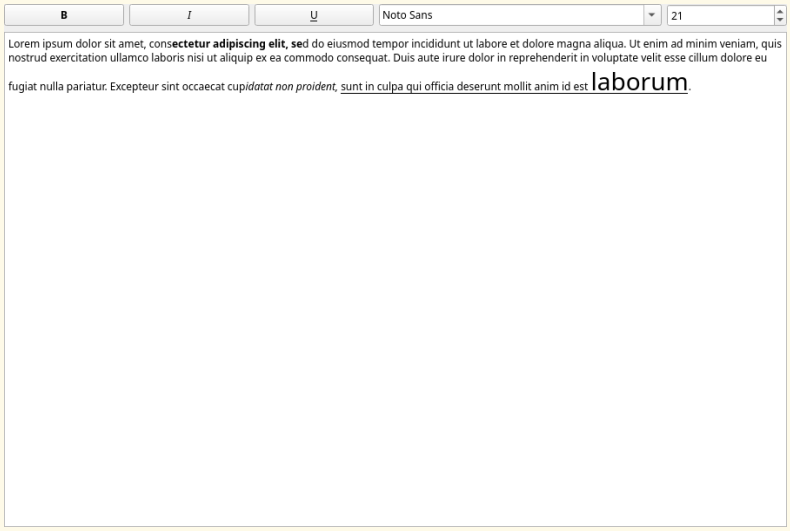
Slika 1.29: Aplikacija za čuvanje beleški

Zadatak 1.3.3: ***Formatiranje teksta***

Implementirati aplikaciju za formatiranje teksta sa izgledom kao na Slici 1.30. Omogućiti podebljanje, italicizovanje, podvlačenje i promenu fonta odabranog raspona teksta.



(a)

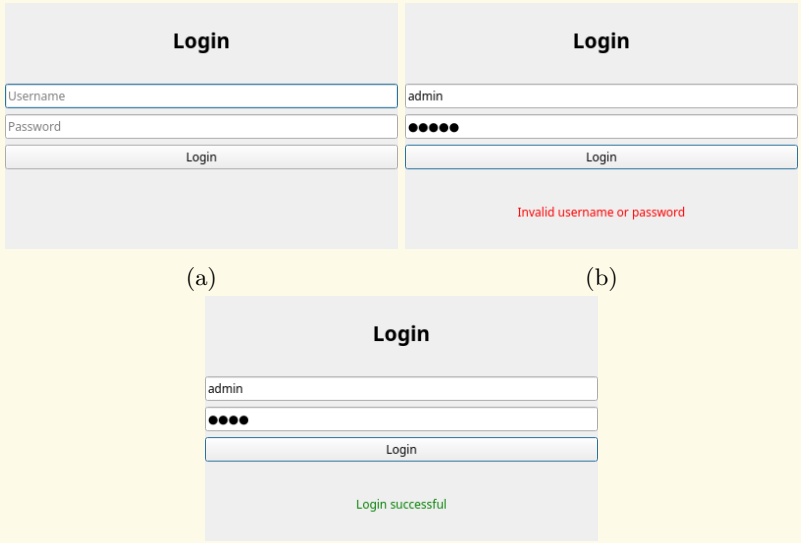


(b)

Slika 1.30: Aplikacija za formatiranje teksta

Zadatak 1.3.4: *Prijava*

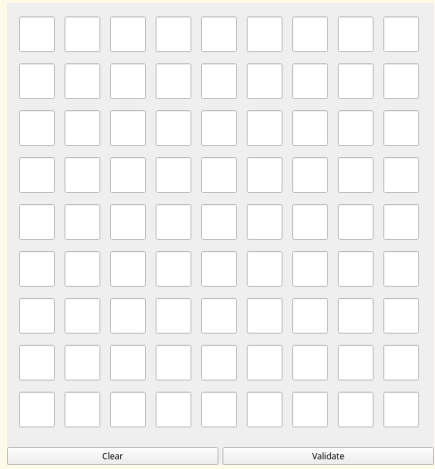
Implementirati aplikaciju za prijavu korisnika sa izgledom kao na Slici 1.31. Pretpostaviti da je ispravno korisnicko ime `admin` i lozinka `1234`. Prilikom pokušaja prijave, u zavisnosti od unetih podataka, ispisati odgovarajući tekst kao u primeru ispod.



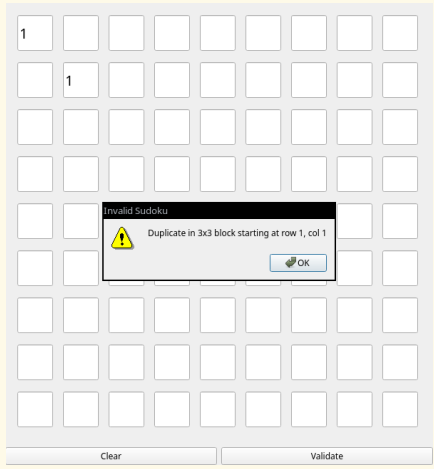
Slika 1.31: Aplikacija za prijavu

Zadatak 1.3.5: ***Igra „Sudoku”***

Implementirati aplikaciju za igru *Sudoku* sa izgledom kao na Slici 1.32. Cilj igre *Sudoku* je popuniti kvadratnu mrežu 9×9 brojevima od 1 do 9 tako da svaki red, svaka kolona i svaka 3×3 pod-mreža (blok) sadrže sve brojeve od 1 do 9, bez ponavljanja. Prilikom pokretanja aplikacije, sva polja su prazna. Igra je završena kada je mreža potpuno popunjena i kada su sva pravila ispoštovana. Dugme `Clear` čisti unete brojeve iz svih ćelija, dok dugme `Validate` proverava da li uneti brojevi zadovoljavaju *Sudoku* pravila, i vraća odgovarajuću poruku u iskačućem prozoru.



(a)



(b)

1

Slika 1.32: Igra *Sudoku*

Zadatak 1.3.6:

U Septembru 2003. godine razvojni tim Matematičkog fakulteta razvio je paket aplikacija koji je podržavao intenzivna numerička izračunavanja u programskim jezicima C i Fortran. Najprodavanija aplikacija računala je (najbrže do tada!) Fibonačijeve i Markus-Lukasove brojeve.
Fibonačijevi brojevi se računaju po formuli:

$$F_0 = 0,\ F_1 = 1,\ F_n = F_{n-1} + F_{n-2}$$

Markus-Lukasovi brojevi se računaju po formuli ($c_0, c_1, a, b \in \mathbb{R}$, $m, l \in \mathbb{N}$):

$$L_0 = c_0,\ L_1 = c_1,\ L_n = a^m L_{n-1} + b^l L_{n-2}$$

Napisati program koji računa n-ti Fibonačijev ili Markus-Lukasov broj u zavisnosti od izabrane opcije, po uzoru na Sliku 1.33. U slučaju računanja Markus-Lukasovog broja, potrebno je učitati i odgovarajući konfiguracioni `JSON` fajl.
Grafički interfejs se sastoji od:

- labele sa nazivom programa,
- radio dugmadi pomoću kojih se selektuje algoritam,
- dela za učitavanje parametra n i konfiguracionog fajla u slučaju odabira Kvazi-Lukasovih brojeva,
- dugmeta koje pokreće računanje i
- polja za ispis rezultata

Primer 1

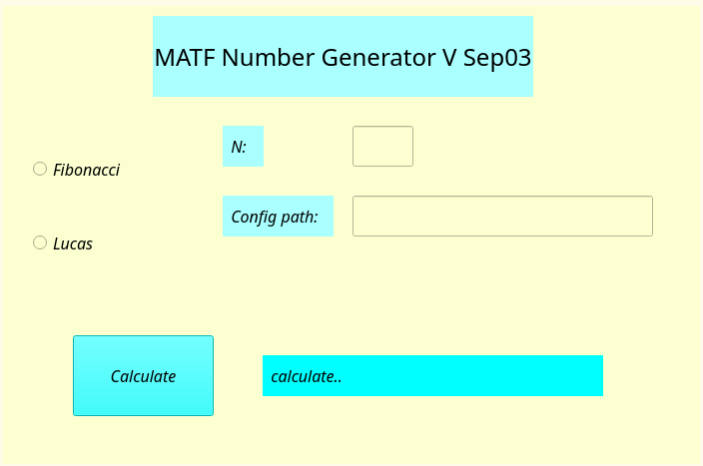
```
Ulazna datoteka config_lucas.json:
        {"c0":0.63, "c1":-5,60, "a":1.23, "b":1.88, "m": 2, "l": 1}
```

Primer 2

```
Poziv: python 1.py F 50
Izlaz: 7778742049
```

Primer 3

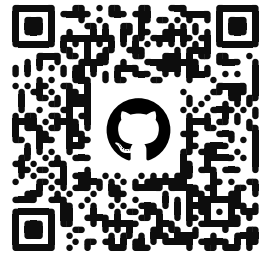
```
Poziv: python 1.py L 7
Izlaz: 32672.409281
```



Slika 1.33: Aplikacija MATF Generator brojeva

Glava 2

Programiranje ograničenja



Uvod

Programiranje ograničenja (engl. *constraint programming*) je programska paradigma zasnovana na definisanju uslova ili ograničenja koja problem mora da ispuni, umesto da se eksplicitno navode koraci za njegovo rešavanje. Programer definiše skup pravila ili zahteva, a sistem automatski pronalazi rešenja koja zadovoljavaju ta ograničenja. Ovaj pristup je posebno koristan za rešavanje kombinatornih problema, optimizaciju i zadatke u kojima treba pronaći rešenja sa složenim uslovima. Razvoj programiranja ograničenja počeo je tokom 1980-ih godina, kao prirodni nastavak i dopuna logičkog programiranja, sa ciljem da se pojednostavi rešavanje komplikovanih problema sa mnogo uslova i promenljivih. Programiranje ograničenja danas je podržano u različitim programskim jezicima, među kojima su Prolog i Pajton (engl. *Python*).

2.1 Uvodni zadaci

Zadatak 2.1.1

Napisati program koji na standardni izlaz ispisuje sve kombinacije oblika `x y z`, gde je `x` $\in$ {`a, b, c`}, `y` $\in$ {`1, 2, 3`} i `z` $\in$ {`0.1, 0.2, 0.3`} tako da važi uslov `10 · z = y`.

Primer 1

```
Izlaz: a 3 0.3
```

Primer 2

```
Izlaz: b 2 0.2
```

Primer 3

```
Izlaz: c 1 0.1
```

Zadatak 2.1.2

Napisati program koji pronalazi trocifren broj `ABC` (gde su `A`, `B` i `C` cifre tog broja) tako da je količnik `ABC / (A + B + C)` minimalan. Pri tome, cifre `A`, `B` i `C` moraju biti međusobno različite. Na standardni izlaz ispisati pronađeni trocifren broj `ABC`.

Primer 1

```
Izlaz: 189
```

Zadatak 2.1.3

Napisati program koji pronalazi sve vrednosti promenljivih `X`, `Y` i `Z` za koje važi:

$$X >= Z \quad \text{i} \quad X * 2 + Y * X + Z <= 34$$

pri čemu promenljive pripadaju sledećim domenima:

- $X \in \{1, 2, ..., 90\}$
- $Y \in \{2, 4, 6, ..., 60\}$ (parni brojevi)
- $Z \in \{1, 4, 9, 16, ..., 100\}$ (kvadrati prirodnih brojeva)

Sve rezultate ispisati na standardni izlaz u formatu:
`X = broj, Y = broj, Z = broj`.

Primer 1

```
Izlaz: X = 7, Y = 2, Z = 4
```

Primer 2

```
Izlaz: X = 6, Y = 2, Z = 4
```

Primer 3

```
Izlaz: X = 5, Y = 4, Z = 4
```

Primer 4

```
Izlaz: X = 5, Y = 2, Z = 4
```

Primer 5

```
Izlaz: X = 4, Y = 4, Z = 4
```

Primer 6

```
Izlaz: X = 4, Y = 2, Z = 4
```

Primer 7

```
Izlaz: X = 3, Y = 8, Z = 1
```

Primer 8

```
Izlaz: X = 3, Y = 6, Z = 1
```

Primer 9

```
Izlaz: X = 3, Y = 2, Z = 1
```

Zadatak 2.1.4

Napisati program koji pronalazi sve vrednosti promenljivih `X`, `Y`, `Z` i `W` za koje važe sledeći uslovi:

- $X \geq 2 \cdot W$
- $3 + Y \leq Z$
- $X - 11 \cdot W + Y + 11 \cdot Z \leq 100$

Domeni promenljivih su:

- $X \in \{1, 2, \ldots, 110\}$
- $Y \in \{1, 3, 5, \ldots, 51\}$
- $Z \in \{10, 20, 30, \ldots, 100\}$
- $W \in \{1, 8, 27, 64, 125, 216, 343, 512, 729, 1000\}$

Sve ispravne kombinacije ispisati na standardni izlaz u formatu:
`X: broj, Y: broj, Z: broj, W: broj`

Primer 1

```
IZLAZ: X: 87, Y: 15, Z: 20, W: 27
```

Primer 2

```
IZLAZ: X: 85, Y: 15, Z: 20, W: 27
```

Primer 3

```
IZLAZ: X: 58, Y: 15, Z: 20, W: 27
```

Primer 4

```
IZLAZ: X: 56, Y: 15, Z: 20, W: 27
```

Primer 5

```
IZLAZ: X: 54, Y: 15, Z: 20, W: 27
```

Primer 6

```
IZLAZ: X: 109, Y: 13, Z: 20, W: 27
```

Primer 7

```
IZLAZ: X: 79, Y: 3, Z: 20, W: 27
```

Primer 8

```
IZLAZ: X: 77, Y: 3, Z: 20, W: 27
```

Primer 9

```
IZLAZ: X: 75, Y: 3, Z: 20, W: 27
```

Primer 10

```
IZLAZ: X: 55, Y: 1, Z: 10, W: 8
```

Zadatak 2.1.5

Dati su novčići od 1, 2, 5, 10 i 20 dinara. Napisati program koji pronalazi sve moguće kombinacije ovih novčića tako da njihov zbir iznosi tačno 50 dinara. Sve kombinacije ispisati na standardni izlaz.

Primer 1

```
Izlaz: 1 din: 0
       2 din: 0
       5 din: 0
       10 din: 1
       20 din: 2
       Ukupno: 50
```

Primer 2

```
Izlaz: 1 din: 0
       2 din: 0
       5 din: 2
       10 din: 0
       20 din: 2
       Ukupno: 50
```

Primer 3

```
Izlaz: 1 din: 1
       2 din: 2
       5 din: 1
       10 din: 0
       20 din: 2
       Ukupno: 50
```

Primer 4

```
Izlaz: 1 din: 3
       2 din: 1
       5 din: 1
       10 din: 0
       20 din: 2
       Ukupno: 50
```

Primer 5

```
Izlaz: 1 din: 5
       2 din: 0
       5 din: 1
       10 din: 0
       20 din: 2
       Ukupno: 50
```

Primer 6

```
Izlaz: 1 din: 0
       2 din: 5
       5 din: 0
       10 din: 0
       20 din: 2
       Ukupno: 50
```

Zadatak 2.1.6

Napisati program koji učitava ceo broj n i ispisuje magičnu sekvencu S brojeva od 0 do $n-1$. Sekvenca $S = (x_0, x_1, \ldots, x_{n-1})$ je magična ako za svako $i = 0, 1, \ldots, n-1$ važi da se broj i pojavljuje tačno x_i puta u sekvenci S. Sve rezultate ispisati na standardni izlaz.

Primer 1

```
Ulaz : Duzina sekvence: 5
Izlaz: [2, 1, 2, 0, 0]
```

Primer 2

```
Ulaz : Duzina sekvence: 7
Izlaz: [3, 2, 1, 1, 0, 0, 0]
```

Primer 3

```
Ulaz : Duzina sekvence: 4
Izlaz: [2, 0, 2, 0]
Izlaz: [1, 2, 1, 0]
```

Primer 4

```
Ulaz : Duzina sekvence: 6
Izlaz: Ne postoji magicna
       sekvenca duzine 6
```

2.2 Kriptoaritmetike

Zadatak 2.2.1

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

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

Svako slovo mora predstavljati jedinstvenu cifru od 0 do 9 (nema ponavljanja cifara), a prva cifra ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u obliku sabiranja brojeva.

Primer 1

```
IZLAZ:   765
        +765
        =1530
```

Primer 2

```
IZLAZ:   734
        +734
        =1468
```

Primer 3

```
IZLAZ:   867
        +867
        =1734
```

Primer 4

```
IZLAZ:   846
        +846
        =1692
```

Primer 5

```
IZLAZ:   836
        +836
        =1672
```

Primer 6

```
IZLAZ:   928
        +928
        =1856
```

Zadatak 2.2.2

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SEE + YOU = SOON$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Primer 1

```
IZLAZ: 199 + 803 = 1002
```

Primer 2

```
IZLAZ: 199 + 804 = 1003
```

Primer 3

```
IZLAZ: 199 + 805 = 1004
```

Zadatak 2.2.3

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$GREEN + ORANGE = COLORS$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.
NAPOMENA: *Ovaj zadatak ima jedinstveno rešenje.*

Primer 1

```
IZLAZ: 83446 + 135684 = 219130
```

Zadatak 2.2.4

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$NO + GUN + NO = HUNT$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.
NAPOMENA: *Ovaj zadatak ima jedinstveno rešenje.*

Primer 1

```
IZLAZ: 87 + 908 + 87 = 1082
```

Zadatak 2.2.5

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SIX + SIX + SIX = NINE + NINE$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.
NAPOMENA: *Ovaj zadatak ima jedinstveno rešenje.*

Primer 1

```
IZLAZ: 942 + 942 + 942 = 1413 + 1413
```

Zadatak 2.2.6

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$COMPLEX + LAPLACE = CALCULUS$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.7

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$THIS + IS + VERY = EASY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.8

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$CROSS + ROADS = DANGER$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.9

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FATHER + MOTHER = PARENT$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.10

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$WE + WANT + NO + NEW + ATOMIC = WEAPON$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.11

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$EARTH + AIR + FIRE + WATER = NATURE$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.12

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SATURN + URANUS + NEPTUNE + PLUTO = PLANETS$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.13

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$WHEN + IN + ROME + BE + A = ROMAN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.14

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$DONT + STOP + THE = DANCE$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.15

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$HERE + THEY + GO = AGAIN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.16

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$OSAKA + HAIKU + SUSHI = JAPAN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.17

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$MACHU + PICCHU = INDIAN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.18

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SHE + KNOWS + HOW + IT = WORKS$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.19

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$COPY + PASTE + SAVE = TOOLS$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.20

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$THREE + THREE + ONE = SEVEN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.21

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$NINE + LESS + TWO = SEVEN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.22

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$ONE + THREE + FOUR = EIGHT$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.23

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$THREE + THREE + TWO + TWO + ONE = ELEVEN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.24

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SEVEN + SEVEN + SIX = TWENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.25

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$ONE + ONE + ONE + THREE + THREE + ELEVEN = TWENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.26

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$EIGHT + EIGHT + TWO + ONE + ONE = TWENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.27

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$ELEVEN + NINE + FIVE + FIVE = THIRTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.28

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$NINE + SEVEN + SEVEN + SEVEN = THIRTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.29

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$TEN+SEVEN+SEVEN+SEVEN+FOUR+FOUR+ONE = FORTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.30

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$TEN + TEN + NINE + EIGHT + THREE = FORTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.31

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FOURTEEN + TEN + TEN + SEVEN = FORTYONE$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.32

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$NINETEEN + THIRTEEN + THREE + TWO + TWO + \\ ONE + ONE + ONE = FORTYTWO$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.33

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FORTY + TEN + TEN = SIXTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.34

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SIXTEEN+TWENTY+TWENTY+TEN+TWO+TWO = SEVENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.35

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SIXTEEN + TWELVE + TWELVE + TWELVE + NINE + NINE = SEVENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.36

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$TWENTY + TWENTY + THIRTY = SEVENTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.37

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FIFTY + EIGHT + EIGHT + TEN + TWO + TWO = EIGHTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.38

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FIVE + FIVE + TEN + TEN + TEN + TEN + THIRTY = EIGHTY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.39

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$SIXTY + EIGHT + THREE + NINE + TEN = NINETY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.40

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$ONE + NINE + TWENTY + THIRTY + THIRTY = NINETY$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.41

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$MEN * AND = WOMEN$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.42

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$COGITO = ERGO * SUM$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.43

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$FERMAT * S = LAST + THEOREM$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.44

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$WINNIE/THE = POOH$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

Zadatak 2.2.45

Napisati program koji dodeljuje različite cifre različitim karakterima tako da je sledeća jednačina tačna:

$$TWO * TWO + EIGHT = TWELVE$$

Svakom slovu odgovara cifra od 0 do 9. Cifra koja odgovara prvom karakteru ne može biti nula. Pronaći sve moguće kombinacije i ispisati ih na standardni izlaz u odgovarajućem obliku aritmetičke operacije.

2.3 Matematičke slagalice

Zadatak 2.3.1

Magičan kvadrat je kvadrat dimenzija `3x3` takav da je suma svih brojeva u svakom redu, svakoj koloni i svakoj dijagonali jednaka `15`, a svi brojevi od `1` do `9` su različiti. Program treba da pronađe sve moguće magične kvadrate i ispiše ih na standardni izlaz.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Primer 5

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

Primer 6

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

Primer 7

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

Primer 8

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

Primer 9

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

Zadatak 2.3.2

Napisati program koji raspoređuje brojeve od 1 do 9 u dve linije koje se ukrštaju u jednom broju. Svaka linija treba da sadrži tačno 5 brojeva, svi brojevi moraju biti međusobno različiti i iz skupa $\{1, 2, \ldots, 9\}$. Brojevi unutar svake linije moraju biti poređani u rastućem redosledu, a zbir svih brojeva u svakoj liniji mora biti jednak 25. Sve ispravne kombinacije treba ispisati na standardni izlaz u obliku koji odgovara obliku slova `X`:

```
A       G
  B   F
    C
  D   H
E       I
```

Primer 1

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

Primer 2

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

Primer 3

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

Zadatak 2.3.3

Napisati program koji pronalazi vrednosti promenljivih 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 obliku heksagona 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 brojeva duž svake horizontalne i dijagonalne linije bude 38, tj:

$$\begin{aligned} 38 &= A + B + C \\ &= D + E + F + G \\ &= H + I + J + K + L \\ &= M + N + O + P \\ &= Q + R + S \end{aligned}$$

$$\begin{aligned} 38 &= A + D + H \\ &= B + E + I + M \\ &= C + F + J + N + Q \\ &= G + K + O + R \\ &= L + P + S \end{aligned}$$

$$\begin{aligned} 38 &= C + G + L \\ &= B + F + K + P \\ &= A + E + J + O + S \\ &= D + I + N + R \\ &= H + M + Q \end{aligned}$$

Jedno od rešenja ispisati na standardni izlaz u prikazanom heksagonalnom obliku.
NAPOMENA: *Ovako definisan heksagon se naziva magični heksagon reda tri, i ima jedinstveno rešenje.*

Primer 1

```
IZLAZ:     3 17 18
          19 7 1 11
        16 2 5 6 9
          12 4 8 14
            10 13 15
```

Zadatak 2.3.4

Napisati program koji raspoređuje 8 topova na šahovskoj tabli dimenzija 8×8 tako da se nikoja dva topa međusobno ne napadaju. Topovi se međusobno napadaju ukoliko se nalaze u istoj vrsti ili koloni. Potrebno je pronaći i prikazati sve moguće rasporede topova koji zadovoljavaju uslov da se međusobno ne napadaju.
Sve rezultate ispisati na standardni izlaz, jedan rezultat predstavlja jedno moguće rešenje i sastoji se od osam redova sa po osam karaktera, pri čemu karakter `T` označava poziciju topa, a znak `-` prazno polje.

Primer 1

```
Izlaz: T-------
       -------T
       -T------
       ------T-
       --T-----
       ---T----
       -----T--
       ----T---
```

Primer 2

```
Izlaz: T-------
       -------T
       -T------
       ------T-
       --T-----
       ---T----
       ----T---
       -----T--
```

Zadatak 2.3.5

Napisati program koji raspoređuje 8 dama na šahovsku tablu dimenzija 8×8 tako da se nikoje dve dame međusobno ne napadaju. Dame se međusobno napadaju ukoliko se nalaze u istoj vrsti, koloni ili dijagonali. Potrebno je pronaći i prikazati sve moguće rasporede dama koji zadovoljavaju ovaj uslov. Svaki rezultat treba ispisati na standardni izlaz, pri čemu jedan raspored predstavlja jedno rešenje i sastoji se od osam redova sa po osam karaktera. Karakter `D` označava poziciju dame, a karakter `-` prazno polje.

Primer 1

```
Izlaz: --D-----
       -----D--
       -D------
       ------D-
       D-------
       ---D----
       -------D
       ----D---
```

Primer 2

```
Izlaz: --D-----
       -----D--
       -------D
       D-------
       ---D----
       ------D-
       ----D---
       -D------
```

Zadatak 2.3.6

Napisati program koji učitava tablu za igru Sudoku iz datoteke čije ime se zadaje sa standardnog ulaza i rešava Sudoku zagonetku koristeći ograničenja. Sudoku je kvadratna tabla dimenzija 9×9 podeljena na devet 3×3 kvadrata. Svaki red, svaka kolona i svaki od devet manjih kvadrata mora da sadrži brojeve od 1 do 9 bez ponavljanja.
Ulaz programa predstavlja ime datoteke u kojoj se nalazi tabla, gde su nule predstavljene kao prazna polja, a izlaz rešena Sudoku tabla ispisana na standardni izlaz.

Primer 1

```
ULAZ : Unesite ime datoteke sa tablom za sudoku: sudoku2.json
ULAZNA DATOTEKA sudoku2.json:
       [[6, 3, 8, 7, 0, 0, 0, 0, 0],
        [0, 2, 7, 0, 0, 0, 5, 0, 0],
        [5, 4, 0, 6, 0, 9, 2, 0, 0],
        [0, 1, 0, 5, 9, 0, 3, 0, 0],
        [0, 0, 9, 0, 7, 0, 4, 0, 0],
        [0, 0, 6, 0, 2, 4, 0, 1, 0],
        [0, 0, 5, 3, 0, 1, 0, 2, 7],
        [0, 0, 2, 0, 0, 0, 8, 3, 0],
        [0, 0, 0, 0, 0, 7, 9, 4, 5]]
IZLAZ: ==================
       |638 | 752 | 194 |
       |927 | 418 | 563 |
       |541 | 639 | 278 |
       ------------------
       |714 | 596 | 382 |
       |289 | 173 | 456 |
       |356 | 824 | 719 |
       ------------------
       |895 | 341 | 627 |
       |472 | 965 | 831 |
       |163 | 287 | 945 |
       ==================
```

2.4 Optimizacioni problemi

Zadatak 2.4.1

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 20 kg brašna, koliko komada hleba i kifli treba da se umesi kako bi se ostvarila maksimalna zarada (pod pretpostavkom da će pekara sve prodati)? Na standardni izlaz ispisati maksimalnu zaradu koju je moguće ostvariti, kao i koliko komada hleba a koliko komada kifli je potrebno umesiti da bi se ostvarila maksimalna zarada.

Primer 1

```
IZLAZ: Maksimalna zarada je 876 dinara, za nju je potrebno 39 hlebova
       i 67 kifli.
```

Zadatak 2.4.2

Kompanija *Start* ima 250 zaposlenih radnika. Rukovodstvo kompanije je odlučilo da svojim radnicima obezbedi dodatnu edukaciju. Za obuku u programskom jeziku *Elixir* potrebno je platiti 100 evra po osobi, a nakon obuke radnik ostvaruje 150 projekat/sati mesečno, što kompaniji donosi dobit od 5 evra po satu. Za obuku u programskom jeziku *Dart* potrebno je platiti 105 evra po osobi, a nakon obuke radnik ostvaruje 170 projekat/sati mesečno, što kompaniji donosi dobit od 6 evra po satu. Kompanija ima na raspolaganju ukupno 26000 evra za edukaciju i 51200 projekat/sati mesečno koje može iskoristiti. Napisati program koji određuje koliko radnika treba poslati na kurs *Elixir*, a koliko na kurs *Dart*, tako da ukupna dobit bude maksimalna. Pretpostaviti da kompanija može obučiti najviše 250 radnika.

Primer 1

```
IZLAZ: Maksimalna zarada je 166000, broj radnika koje treba poslati
       na kurs Elixir je 50, a broj radnika koje treba poslati na
       kurs Dart je 200.
```

Zadatak 2.4.3

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 određenu količinu deterdženta: 120 ml po stanu, odnosno 100 ml po kući. Mica zaradi 1000 dinara po svakom stanu, odnosno 1500 dinara po kući. Ukoliko Mica radi 40 sati nedeljno i ima 5 l deterdženta na raspolaganju. Na standardni izlaz ispisati koliko stanova i kuća je potrebno da očisti kako bi imala najveću zaradu kao i kolika je najveća moguća zarada.

Primer 1

```
IZLAZ: Najveca moguca zarada je 40000 i ona se ostvaruje kada se
       ocisti 1 stan(ova) i 26 kuca.
```

Zadatak 2.4.4

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 g, dok se za tanjir potroši 100 g 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. Na standardni izlaz ispisati koliko šolja i tanjira Marija mora da napravi kako bi ostvarila maksimalnu zaradu.

Primer 1

```
IZLAZ: Najveca moguca zarada je 600.00 i ona se ostvaruje kada se
       napravi 0 solja i 400 tanjira.
```

Zadatak 2.4.5

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 dm^3 prostora i mesečno može da nabavi robu u iznosu od najviše 14000 evra. Na standardni izlaz ispisati koliko računara, a koliko štampača komšija treba da proda kako bi se maksimalno obogatio.

Primer 1

```
IZLAZ: Maksimalna zarada je: 14680, broj racunara: 59, broj stampaca: 36
```

2.5 Ispitni zadaci

Zadatak 2.5.1

Napisati program koji pronalazi sva rešenja jednačine:

$$((JE + PENSE) - DONC) + JE = SUIS^{a}$$

Različitim slovima odgovaraju različiti brojevi. Brojevi ne mogu početi cifrom 0. Na standardni izlaz ispisati sva rešenja. Rešenja je potrebno numerisati pri ispisu.

Primer 1

```
IZLAZ: 1. ((47 + 17587) - 9653) + 47 = 8028
       2. ((47 + 17587) - 9053) + 47 = 8628
       3. ((47 + 17597) - 8652) + 47 = 9039
       4. ((47 + 17597) - 8052) + 47 = 9639
       5. ((35 + 15065) - 8709) + 35 = 6426
       6. ((35 + 15065) - 8409) + 35 = 6726
       7. ((83 + 13473) - 6542) + 83 = 7097
       8. ((83 + 13473) - 6042) + 83 = 7597
       9. ((35 + 15085) - 6207) + 35 = 8948
       10. ((35 + 15085) - 6907) + 35 = 8248
       11. ((30 + 10740) - 5976) + 30 = 4824
       12. ((30 + 10740) - 5876) + 30 = 4924
       13. ((52 + 12072) - 4309) + 52 = 7867
       14. ((52 + 12072) - 4809) + 52 = 7367
```

[a] *Je pense, donc je suis*, fr. *Mislim, dakle postojim.*

Zadatak 2.5.2

Napisati program koji pronalazi (i ispisuje na standardni izlaz) jedno rešenje jednačine:

$$MANET + MATISSE + MIRO + MONET + RENOIR = ARTISTS.$$

Različitim slovima odgovaraju različiti brojevi. Nijedan broj ne sme počinjati cifrom 0. Na standardni izlaz ispisati jedno (bilo koje ispravno) rešenje u abecednom redosledu.

Primer 1

```
IZLAZ: [1, 2, 0, 3, 4, 5, 6, 7]
```

Primer 2

```
IZLAZ: [1 ,2, 0, 3, 4, 5, 6, 8]
```

Primer 3

```
IZLAZ: [1, 2, 0, 3, 4, 5, 6, 9]
```

Primer 4

```
IZLAZ: [1, 2, 0, 3, 4, 5, 7, 6]
```

Zadatak 2.5.3

Na Milanovoj farmi uzgajaju se samo zečevi i kokoške. Na farmi ima ukupno 18 glava i 56 nogu. Napisati program koji korišćenjem programiranja ograničenjem određuje i na standardni izlaz ispisuje koliko ima životinja koje vrste na farmi.

Primer 1

```
Izlaz: Kokoske: 7
       Zečevi: 11
```

Zadatak 2.5.4

Ivan kreće put Male Azije da proda svoje proizvode (viski, parfem, čaše). Njegov ranac ima 9 delova. Boca viskija zauzima 4 mesta, boca parfema 3, dok jedna čaša zauzima 2 dela. Zarada ostvarena prodajom viskija jeste 15 evra, parfem donosi 10 evra, a čaše 7 evra.
Napisati program koji korišćenjem programiranja ograničenjem određuje i na standardni izlaz ispisuje koliko proizvoda koje vrste Ivan mora poneti na put ukoliko želi da ostvari maksimalni profit (pod pretpostavkom da proda sve proizvode koje ponese).

Primer 1

```
Izlaz: Vino: 8
       Parfem: 1
       Čaša: 0
```

Zadatak 2.5.5

Hyper-6 predstavlja šestougao koji pored šest spoljašnjih temena ima i jedno unutrašnje teme, koje je unutrašnjim ivicama povezano sa svakim od njih. Ukoliko su sve ivice ove figure jedinične, onda se ona naziva *ravni Hyper-6*. Primer ravnog Hyper-6 dat je na slcii ispod, pri čemu su sa `A...G` označena njegova temena.

```
  A   B
C   D   E
  F   G
```

Napisati program koji u ivice *ravnog Hyper-6* upisuje brojeve od 1 do 7, pri čemu je zbir brojeva u temenima svakog od jediničnih jednakostraničnih trouglova manji od 11, dok je zbir brojeva u temenima svakog od jediničnih rombova manji ili jednak 16.

Primer 1

```
Izlaz:  2 7
       5 1 3
        4 6
```

Primer 2

```
Izlaz:  3 5
       7 1 4
        2 6
```

Zadatak 2.5.6

Hyper-6 predstavlja šestougao koji pored šest spoljašnjih temena ima i jedno unutrašnje teme koje je unutrašnjim ivicama povezano sa svakim od njih. Ukoliko su sve ivice ove figure jedinične, onda se ona naziva *ravni Hyper-6*. Glavna paralela *Hyper-6* jeste jedina njegova paralela koja sadrži tri temena i paralelna je osi Ox (podrazumevamo položaj u ravni tako da su dve stranice paralelne osi Ox.
Napisati program koji raspoređuje različite brojeve od 1 do 7 u svako teme ravnog *Hyper-6*, pri čemu važi da zbir svakog od spoljašnjih temena sa susedima nije veći od 15. Pronaći takav raspored gde su elementi na glavnoj paraleli takvi da im je zbir maksimalan a proizvod minimalan. Pri čemu su sa `A...G` označena temena kao na slici ispod (pri ovim oznakama, D je unutrašnje teme a glavna paralela je $C - D - E$). Hyper-6 sa prethodno navedenim imenovanjem temena dat je sa:

```
  A   B
C   D   E
  F   G
```

Primer 1

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

Zadatak 2.5.7

Čuveni kuvar Gordon Remzi stigao je u Beograd na poslovni sastanak sa ciljem da otvori svoj novi restoran. U sred noći probudio se gladan i rešio je da sam sebi pripremi obrok — proteinsku palačinku. Otišao je do obližnje radnje i shvatio da ima dva problema — radnja nije tako bogata sastojcima, a u džepu ima samo 10 evra, koje može da zameni po kursu od 117 dinara (iako je zvanični kurs Narodne banke Srbije 117.52 dinara za evro!).
Sastojke kupuje u kesicama od po 100 grama. Može da kupi najviše 10 kesica, pri čemu želi da napravi palačinku sa što više proteina, a što manje masti i šećera. Ukupna količina masti mora biti manja od 500 grama, dok šećera ne sme biti više od 150 grama.
Koristeći podatke iz tabele, izračunati maksimalnu količinu proteina koju Gordon može da postigne, poštujući sva ograničenja. Na standardni izlaz ispisati **samo** jedan broj: maksimalnu količinu proteina. Gordon ima ukupno 1170 dinara (10 evra × 117 RSD).

Sastojak	Cena (din/kesica)	Broj kesica u radnji	Protein (g)	Masti (g)	Šećer (g)
brašno	30	10	20	30	5
plazma	300	20	15	10	30
jaja	50	7	70	150	2
mleko	170	5	40	32	15
višnja	400	3	23	3	45
nutela	450	9	7	15	68

Primer 1

```
Izlaz: 355
```

Zadatak 2.5.8

Projekat *DataForge* koristi razne *Python* biblioteke. Svaka biblioteka može zavisiti od drugih biblioteka, ali može imati i konflikte sa njima. Cilj je instalirati što je više biblioteka moguće, uz maksimalnu kompatibilnost među bibliotekama (tj. ispoštovati sve zavisnosti i izbeći konflikte), a pri tome maksimizovati sumu prioriteta instaliranih biblioteka. Ako je biblioteka instalirana, sve biblioteke od kojih ona zavisi takođe moraju biti instalirane. Dve biblioteke koje su u konfliktu ne mogu biti instalirane istovremeno. Sve biblioteke sa sobom nose određenu cenu resursa koje troše, a ukupan trošak resursa (cena) instaliranih biblioteka ne sme prelaziti 15. Takođe, želimo da instaliramo bar dve biblioteke. Koristeći podatke iz naredne tabele pronaći kombinaciju instaliranih biblioteka tako da se postigne maksimalna moguća vrednost sume njihovih prioriteta. Na standardni izlaz ispisati instalirane biblioteke.

Biblioteka	Zavisi od	Konflikti	Cena	Prioritet
NumPy	-	*Keras*	3	2
Keras	-	*NumPy*, *Pandas*	5	3
Pandas	-	*Keras*	2	1
Seaborn	*NumPy*	-	4	2
OpenCV	*Keras*, *Pandas*	-	3	3

Tabela 2.1: Zavisnosti i konflikti softverskih biblioteka

Primer 1

```
IZLAZ: Instalirane biblioteke: NumPy Pandas Seaborn.
```

Zadatak 2.5.9

Problem ranca predstavlja jedan od najpoznatijih problema oblasti matematičke optimizacije. Potrebno je u ranac kapaciteta W staviti predmete, od kojih je svaki vrednosti v_i tako da zbir svih v_i u rancu bude maksimalan. Za svaki predmet i poznata je njegova vrednost v_i, njegova težina w_i i koliko komada tog predmeta nam je dostupno c_i (za predmet i u ranac nije moguće staviti više od c_i komada predmeta).

1. Napisati predikat `get(X, I, Y)` koji iz liste `X` vraća element `Y` koji se nalazi na indeksu `I`. Pretpostaviti da indeksiranje počinje od 0.
2. Napisati predikat `ranac(X, Brojaci, Tezine, Vrednosti, Kapacitet)` koji rešava problem ranca tako što u listu `X` upisuje koliko instanci kog predmeta će biti odabrano i ispisuje kolika su zarada i težina ranca pri odabiru tih predmeta.

Sledeća tabela predstavlja jednu instancu opisanog problema koju je potrebno rešiti. Pretpostaviti da postoje tačno 4 predmeta: laptop, dijamant, pantalone i četkica za zube.

ime	laptop	dijamant	pantalone	četkica
i	0	1	2	3
v	3400	1800	200	10
w	500	7	55	1
c	2	3	5	4

Na primer, za $W = 128$, rešenje je uzeti 3 dijamanta, 1 pantalone i 4 četkice za zube.

Primer 1

```
Ulaz : | ?- ranac(X, [2, 3, 5, 4], [500, 7, 55, 1],
       [3400, 1800, 200, 10], 128)
Izlaz: Zarada 5640
       Tezina 80
       X = [0,3,1,4]
```

Zadatak 2.5.10

Doktor Mile kreira novu terapiju za jačanje imuniteta. Doktor želi da unošenjem vitamina i minerala poveća količinu hemoglobina u organizmu. Na raspolaganju ima 9 tableta vitamina B, 19 tableta vitamina C, 6 tableta vitamina D, 4 tablete magnezijuma, samo dve kapsule selena i 8 tableta cinka. Potrebno je odabrati, za svaki dan u nedelji po najviše jednu tabletu, kojom će se povećati količina hemoglobina u krvi. Trajanje sprint terapije vitaminima i mineralima jeste sedam dana.
Doktor Mile na raspolaganju ima 100 evra (koje može da zameni po kursu od 118 dinara za evro), pri čemu tableta vitamina B košta 130 dinara, vitamina C 800 dinara, vitamina D 150 dinara, magnezijuma i selena 370 odnosno 490 dinara redom, dok tableta cinka košta isto koliko i tableta vitamina D. Kako unos vitamina/minerala može dovesti do dizbalansa u organizmu, doktor mora voditi računa da njegova terapija ne poveća sintezu proteina za više od 100 g, niti stimuliše sintetisanje više od 200 g amino-kiselina. Količina proteina i amino-kiselina koje tableta svakog od pomenutih suplemenata sintetiše (u gramima) data je u tabeli:

Sastojak	Stimulacija sinteze proteina	Simulacija sinteze amino-kiselina
vitamin B	15	33
vitamin C	11	31
vitamin D	10	20
magnezijum	22	18
selen	1	21
cink	13	16

Potrebno je kreirati sprint trerapiju tako da se maksimizuje sinteza hemoglobina, pri čemu jedna tableta vitamina B sintetiše 92.5 g hemoglobina, vitamina C 155.5 g, vitamina D 79.6 g, magnezijuma 156.2 g, selena čak 413 g i cinka 137.7 g. Na standardni izlaz ispisati, zaokruženo na dve decimale, maksimalnu količinu hemoglobina koju je moguće sintetisati pri navedenim uslovima.

Primer 1

```
Izlaz: 1605.60
```

Zadatak 2.5.11

Marko se zaljubio. Srećom, on radi u fabrici čokolade gde na raspolaganju ima nekoliko vrsta čokolade. Svojoj devojci Marko želi napraviti najslađu čokoladu koju može spakovati u torbu. Da devojka ne bi primetila čokoladu ukoliko se slučajno sretnu pre nego što Marko pripremi poklon, on može kupiti najviše 3kg čokolade. U torbu može staviti najviše 1 dm^3 čokolade. Za kupovinu čokolade Marko na raspolaganju ima 300 evra.
Koristeći podatke iz sledeće tabele, izračunati koliko koje čokolade Marko mora da uzme kako bi napravio poklon maksimalne slatkoće, vodeći računa o ograničenjima mase, zapremine i cene. Na standardni izlaz ispisati ukupnu slatkoću Markoovog poklona, kao i njegov sastav.

Tip čokolade	Masa (g)	Dimenzije (cm)	Cena (EUR)	Slatkoća
Čokolada A	100	$8 \times 2.5 \times 0.5$	20	8
Čokolada B	45	$7 \times 2 \times 0.5$	16	6.8
Čokolada C	10	$3 \times 2 \times 0.5$	9	4
Čokolada D	25	$3 \times 3 \times 0.5$	7	3

Primer 1

```
Izlaz: Ukupna slatkoća: 365.0
       Čokolada A: 27
       Čokolada B: 2
       Čokolada C: 16
       Čokolada D: 2
```

Zadatak 2.5.12

Nikola se sprema za takmičenje u bodibildingu, i potrebno mu je da svoju ishranu dovede do savršenstva. Na raspolaganju ima četiri suplementa: whey protein, kreatin monohidrat, i anaboličke steroide: testosteron i danazol. Prema planu ishrane, Nikola može pojesti najviše 3 kg suplementacije. Jedna doza whey proteina teži 100 g, kreatina 45 g, a anabolika — testosterona i danazola — 10 g i 25 g. Svaki suplement povećava vezivanje vode u organizmu, koje ne sme preći 1 dm^3. Doza whey proteina vezuje 3 cm^3 vode, kreatin 10 cm^3, testosteron 15 cm^3, a danazol 20 cm^3. Nikola na raspolaganju ima po 20 doza svakog suplementa, i ukupni budžet iznosi 300 EUR. Cena jedne doze je:

- whey protein — 8 EUR
- kreatin — 6 EUR
- testosteron — 14 EUR
- danazol — 11 EUR

Efekat na povećanje mišićne mase po dozi je:

- whey protein — 5 g
- kreatin — 11 g
- testosteron — 20 g
- danazol — 15 g

Korišćenjem programiranja ograničenja izračunati i na standardni izlaz ispisati koliko doza kog suplementa Nikola treba da unese kako bi maksimizirao ukupnu dobijenu mišićnu masu, poštujući ograničenja mase, zapremine, cene i broja doza.

Primer 1

```
Izlaz: Maksimalno povecanje misicne mase: 454
       Plan ishrane:
       1 doza whey proteina,
       19 doza kreatina monohidrata,
       0 doza testosterona,
       16 doza danazola
```

Zadatak 2.5.13

Ferarijevi stručnjaci rade na poboljšanju formule za ovogodišnju F1 trku na *Hungaroringu*. Potrebno je na bolid dodati nove komponente koje bi povećale njegovu snagu i brzinu. Na raspolaganju su:

- 4 katalizatora `D17`,
- 31 filter za sagorevanje `R35`,
- 14 turbo pumpi `Z`,
- 3 pogonska motora *Nitro Scalar*.

Ubrzanja koje daju komponente su sledeća:

- katalizator (D17): 9.3 sekundi,
- filter za sagorevanje (R35): 9.9 sekundi,
- turbo pumpa (Z): 2.17 sekundi,
- Nitro Scalar motor: 303.5 sekundi.

Trkački kolegijum doneo je sledeća ograničenja:

- Ukupna težina komponenti ne sme preći 20 000g, pri čemu su težine komponenti redom:

$$480\text{g}, \quad 3980\text{g}, \quad 290\text{g}, \quad 6600\text{g}.$$

- Ukupna zapremina dodatnih komponenti ne sme preći 3500cm^3, pri čemu su dimenzije komponenti (u cm):

$$6 \times 14, \quad 2.4 \times 7.2, \quad 2 \times 3, \quad 24.9 \times 110.$$

- Budžet je ograničen na 20 000 evra, sa cenama po komadu:

$$800, \quad 1300, \quad 120, \quad 9000 \text{ evra}.$$

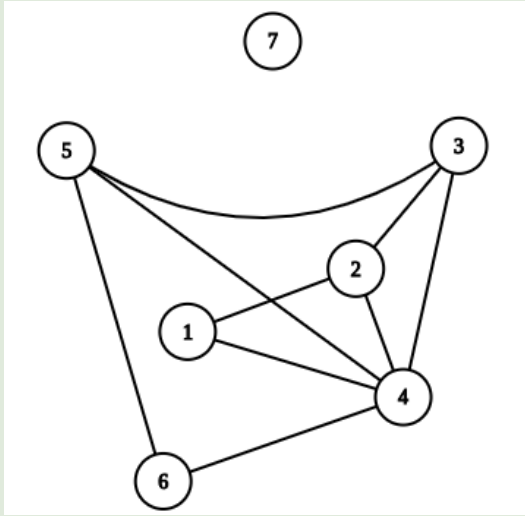
Napisati program korišćenjem programiranja ograničenjem koji računa maksimalno moguće ukupno ubrzanje bolida (u sekundama) koje se može postići korišćenjem dostupnih komponenti, a pri tome poštuje sva navedena ograničenja. Na standardni izlaz ispisati maksimalno moguće ubrzanje kao i odgovarajuće potrebne komponente.

Primer 1

```
Izlaz: Moze se postici ubrzanje od 386.54s
       Potrebne komponente:
       Nitro Solar: 1,
       Katalizator D17': 4,
       Turbo Pumpa: 12,
       Filter R35: 2
```

Zadatak 2.5.14

Mnogi problemi u računarstvu (poput socijalnih mreža, kompilatorskih internih reprezentacija, struktura podataka, modela neuronskih mreža itd.) mogu se modelovati grafovima. Problem bojenja grafa izučava se od 70-tih godina prošlog veka. Zbog eksponencijalne složenosti jedan je od težih problema koji se rešava uz pomoć raznih heuristika.
Milan pokušava da reši problem PP-bojenja grafa sa slike 2.1 koristeći logičko programiranje. Napisati program koji pronalazi sva validna PP-bojenja zadatog grafa (boje kodirati brojevima: `0` – crvena, `1` – plava, `2` – bela, a rezultat ispisati kao vektor gde indeksi u vektoru odgovaraju rednim brojevima čvorova grafa).



Slika 2.1: Graf koji Milan želi da oboji.

Napomena: PP-bojenje *grafa sa slike 2.1 podrazumeva bojenje grafa sa 3 boje (crvenom, plavom i belom), pri čemu je svakom od čvorova dodeljena boja i nikoja dva susedna čvora nisu obojena istom bojom. Dodatno, čvor broj* `7` *mora biti obojen crvenom bojom, dok čvor broj* `3` *ne sme biti obojen zelenom bojom.*

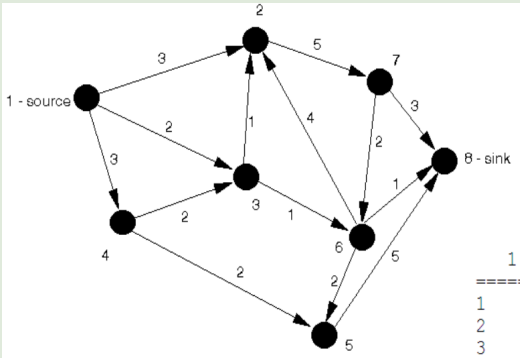
Primer 1	Primer 2	Primer 3
`Izlaz: [1,1,0,2,1,2,0]`	`Izlaz: [1,1,2,0,1,2,0]`	`Izlaz: [2,1,2,0,1,2,0]`

Zadatak 2.5.15

Neka je G usmereni graf $G = (V, E)$, gde je V skup njegovih čvorova a E skup usmerenih grana između njih ($E \subseteq V \times V$). Kapaciteti grana definišu se funkcijom $c : V \times V \longrightarrow R^+$. U grafu G definišu se dva istaknuta čvora: izvor (eng. source, s) i ponor (eng. sink, t). Protok kroz graf G definiše se kao funckija $f : V \times V \longrightarrow R$ koja zadovoljava:

- $(\forall u, v \in V) f(u, v) \leq c(u, v)$ (ograničenje kapaciteta)
- $(\forall u, v \in V) f(u, v) = -f(v, u)$
- $(\forall u \in V \setminus \{s, t\}) \sum_{v \in V} f(u, v) = 0$

Vrednost protoka kroz graf G definiše se kao: $|f| = \sum_{v \in V} f(s, v)$. Problem pronalaska maksimalnog protoka definiše se kao problem pronalaska maksimalnog protoka koji može biti poslat između izvora (čvora s) i ponora (čvora t).
Napisati program koji korišćenjem programiranja ograničenja rešava problem maksimalnog protoka. Program na standardni izlaz ispisuje maksimalni protok koji se može postići za konkretan graf definisan na slici 2.2.



Capacity matrix

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

Slika 2.2: Protok kroz test graf.

Primer 1

```
Izlaz: 7
```

Zadatak 2.5.16

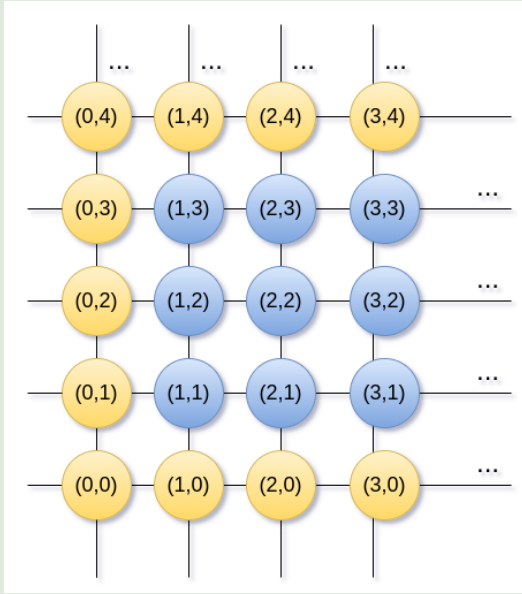
Celobrojna rešetka predstavlja jedan od osnovnih pojmova geometrije brojeva. Definisana je formulom:

$$C = \{ f(m,n) \in \mathbb{R} \mid m,n \in \mathbb{Z} \}$$

gde je $f : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$. Podmreža celobrojne rešetke C_1 dimenzije 3×3 definiše se kao:

$$C_1 = \{ g(m,n) \in \mathbb{R} \mid m,n \in \mathbb{Z}, 1 \leq m,n \leq 3 \}$$

gde je g proizvoljna funkcija $g : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$ (označena plavom bojom na narednoj slici).



Slika 2.3: Primer podrešetke C_1

Napisati program koji raspoređuje brojeve $1, 2, \ldots, 9$ u čvorove rešetke C_1, pri čemu je u svaki čvor upisan broj **različite parnosti** od zbira koordinata tog čvora. Na izlaz ispisati jedno validno popunjavanje rešetke u tri reda.

Primer 1

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

Zadatak 2.5.17

Korišćenjem programiranja ograničenja rešiti maksimizacioni problem:

$$\max(\|X\|_2),$$

pri uslovu

$$AX < b,$$

gde je

$$A = \begin{bmatrix} 2 & 3 \\ 45 & -34 \end{bmatrix}, \quad b = \begin{bmatrix} 23 & 12 \end{bmatrix}^T$$

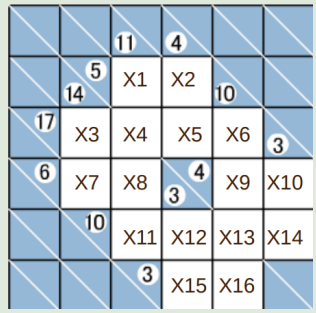
i gde je X dvodimenzioni vektor iz celobrojne rešetke $[0, 10]^2$. Na standardni izlaz ispisati vektor X.

Primer 1

```
IZLAZ: X = [0, 7]
```

Zadatak 2.5.18

Kakuro, igra nastala u Japanu neposredno nakon *Sudokua*, jeste matematička verzija ukrštenice koja koristi zbira cifara umesto reči. Cilj Kakura je popuniti bele kvadrate ciframa 1-9 tako da svaka "reč"(vrsta odnosno kolona) ima dati zbir. Na primer, na slici 2.4 mora važiti X1+X2=5, X3+X4+X5+X6=17, X3+X7=14, itd. Niti jedna cifra se ne može koristiti više od jednom u svakoj "reči".
Napisati program koji rešava Kakuro zagonetku prikazanu na slici 2.4. Na standardni izlaz ispisati vektor u kome vrednost na indeksu i odgovara vrednosti promenljive X_i sa slike 2.4.



Slika 2.4: Kakuro zagonetka.

Primer 1

```
IZLAZ: [2,3,9,5,1,2,5,1,3,1,3,1,4,2,2,1]
```

Zadatak 2.5.19

Sudoku kvadrat jeste kvadrat veličine 9×9*, podeljen na* 9 *manjih kvadrata dimenzija* 3×3*. Svaki od ovih kvadrata, kao i svaka kolona i svaki red, mora da sadrži svih 9 brojeva od 1 do 9 bez ponavljanja.* Korišćenjem programiranja ograničenja pronaći i na standardni izlaz ispisati **sva** rešenja sudoku zagonetke date sledećom početnom konfiguracijom (nepopunjene ćelije označene su zvezdicom $*$):

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

Prilikom ispisa rešenja razdvojiti ispisivanjem novog reda.

Primer 1

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

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

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