

Osnovi programiranja

Beleške sa vežbi

Smer *Računarstvo i informatika*
Matematički fakultet, Beograd

Jelena Tomašević i Sana Stojanović

November 7, 2005

Sadržaj

1	5
1.1 L ^A T _E X	5
2	11
2.1 Formalni jezici i formalne gramatike	11
2.1.1 Jezici: zadaci	11
2.1.2 Gramatike i jezici: zadaci	12
3	15
3.1 Pozicioni brojni sistemi - konverzije	15
3.2 Prebacivanje iz sistema sa osnovom b u dekadni brojni sistem . .	15
3.3 Prebacivanje iz dekadnog brojnog sistema u sistem sa osnovom b	16
3.4 Rad sa realnim brojevima	17
3.4.1 Prevođenje realnih brojeva iz sistema sa osnovom B u dekadni brojni sistem	18
3.4.2 Prevođenje realnih brojeva iz dekadnog brojnog sistema u sistem sa osnovom B	18
3.5 Direktno prevođenje iz binarnog u heksadekadni sistem	19
3.6 Direktno prevođenje iz binarnog u oktalni sistem	19
4	21
4.1 Reprezentacija znakovnih podataka	21
4.1.1 Tekst je niz karaktera	21
4.1.2 Zapis karaktera u računaru	21
4.1.3 Skupovi karaktera	21
4.1.4 Kodiranje ostalih jezika	22
4.1.5 Osobine YUSCII koda	23
4.1.6 Kodne strane	23
4.1.7 Kodne strane kod nas	24
4.1.8 Višebajtni karakterski kodovi	24
4.1.9 Karakteri, Glifovi, Fontovi	25
5	27
5.1 Specifikacija sintakse programskih jezika, meta jezici	28
5.1.1 BNF (Backus-Naur form)	28
5.1.2 EBNF (Extended Backus-Naur form)	29
5.1.3 Sintaksni dijagrami	30

Čas 1

1.1 L^AT_EX

Za pravljenje tex dokumenata možemo koristiti bilo koji tekstualni editor. Na primer, u Linux-u, koristimo *emacs*, *kate* ili *kwrite*.

Minimalni tex dokument je sledećeg oblika:

```
\documentclass{article}

\author{Ime i prezime}

\title{Naslov}

\begin{document}

\maketitle

\section{Uvod}

...

\end{document}
```

Ako smo ga sačuvali kao **rad.tex**, možemo ga prevesti naredbom **latex rad.tex** čime dobijamo DVI format. Ako želimo da dobijemo PDF format kucamo

```
dvipdf rad.dvi rad.pdf
```

Ukoliko prevođenje vršimo pod Windows operativnim sistemom pod pretpostavkom da imamo instaliran MikTex onda u Command Prompt-u kucamo sledeće:

```
latex rad
```

Za prevođenje u ps format kucamo

```
dvips rad.dvi
```

a za prevođenje u pdf format kucamo

```
dvipdfm rad.dvi
```

Izgled dokumenta možemo videti pomoću naredbe

```
yap rad.dvi
```

ili
`start rad.ps`
 ili
`start rad.pdf`

Kad napravimo osnovni `tex` dokument dalje ga formiramo po našoj želji. Ukoliko želimo da pređemo na novu stranu onda ukucamo `\newpage`. Naredbom `\section{Naziv poglavlja}` dodajemo *novu poglavlje* koje se prevođenjem automatski numerise u skladu sa svojim rednim brojem. Na sličan način dodajemo i podpoglavljia naredbom `\subsection{Naziv podpoglavljia}`.

Ako u nekom delu teksta želimo da se pozovemo na neko tvrđenje koje smo dokazali u nekom drugom poglavlju, to možemo uraditi označavanjem tog poglavlja (onog u kom se nalazi tvrđenje na koje se pozivamo) *labelom*. To radimo tako što posle naredbe početka poglavlja dodamo naredbu `\label{oznaka}` a u svakom drugom poglavlju koje se poziva na njega stavimo *referencu* naredbom `\ref{oznaka}`. *oznaka* je niska koju sami biramo i jedinstveno je dodeljena tom poglavlju.

U sledećoj tabeli su date naredbe za različite *stilove teksta*:

Boldovan tekst	<code>\textbf{tekst}</code>
<i>Italik</i>	<code>\emph{tekst}</code>
<u>Podvučen</u>	<code>\underline{tekst}</code>
<u>Nadvučen</u>	<code>\overline{tekst}</code>

Specijalne karaktere, one koji učestvuju u naredbama, ispisujemo dodavanjem `\` ispred određenog karaktera. To su `$`, `&`, `%`, `#`, `-`, `{`, `}`. Da bi ispisali `\` kucamo `\backslash`.

Da bi dobili *novi red* potrebno je da u `tex` dokumentu napravimo prazan red (odnosno da dva puta pritisnemo `ENTER`) ili da ukucamo `\\` ili `\newline`.

Za dobijanje *ćiriličnih slova* koristimo naredbe date u sledećoj tabeli:

ć	<code>\v c</code>
ć	<code>\' c</code>
đ	<code>\d</code>
ž	<code>\v z</code>

Sledećom tabelom su date još neke mogućnosti *formatiranja teksta*:

Tekst koji se nikad ne prelama	<code>\mbox{text}</code>
Uvučen tekst	Između <code>\begin{quote}</code> i <code>\end{quote}</code>
Doslovno se prepisuje na izlaz	Između <code>\begin{verbatim}</code> i <code>\end{verbatim}</code>
Matematički tekst	Između <code>\$</code> i <code>\$</code> ili između <code>\begin{math}</code> i <code>\end{math}</code>
Izdvojena matematička formula	Između <code>\begin{displaymath}</code> i <code>\end{displaymath}</code>
Tekst unutar formule	<code>\textrm{text}</code>
Levo poravnan tekst	Između <code>\begin{flushleft}</code> i <code>\end{flushleft}</code>
Desno poravnan tekst	Između <code>\begin{flushright}</code> i <code>\end{flushright}</code>
Centriran tekst	Između <code>\begin{center}</code> i <code>\end{center}</code>

Ako imamo neko nabranje koje hoćemo da završimo sa ... kucamo `\ldots`.

Komentare kucamo iza znaka `%`. Sve u tom redu se neće videti na izlazu.

Grčko slovo Ω kucamo naredbom `\Omega`, slično i za ostala grčka slova. Ako je prvo slovo u naredbi veliko dobićemo veliko grčko slovo, inače dobijamo mala grčka slova. Malo slovo ω dobijamo naredbom `\omega`.

Možemo naznačiti na kojim mestima reči smeju biti podeljene pri prelazu iz jednog reda u drugi (da ne bi došlo do nepravilnih podela) i to radimo naredbom `\hyphenation{lista reči}`. Pri čemu **lista reči** sadrži spisak svih reči za koje želimo da naglasimo na koji način se dele, a to radimo stavljanjem povlake (–) na mestima gde je podela dozvoljena.

Ako želimo *dužu povlaku* kucamo `--`, a ako želimo još dužu kucamo `---`. Latex ih automatski spaja u jednu dugu povlaku i dobijamo `---` odnosno `---`.

Razne *matematičke formule* se mogu otkucati korišćenjem naredbi iz sledeće tabele:

$\frac{1}{2}$	<code>\frac{1}{2}</code>	$\vec{a}$	<code>\vec{a}</code>
a^x	<code>a^x</code>	$\log$	<code>\log</code>
a_n	<code>a_n</code>	$\max$	<code>\max</code>
ϵ	<code>\epsilon</code>	ε	<code>\varepsilon</code>
$\forall$	<code>\forall</code>	$\exists$	<code>\exists</code>

Ako želimo da dodamo *fusnotu*¹ to radimo naredbom `\footnote{tekst fusnote}`

¹ovo je fusnota

na mestu gde želimo da nam se pojavi oznaka fusnote. Tekst fusnote će se automatski smestiti na dno strane.

Liste pravimo na više načina u zavisnosti od toga da li želimo numerisanu, nenumerisanu ili opisnu listu.

Numerisanu listu dobijamo na sledeći način:

```
\begin{enumerate}
\item prva stavka      1. prva stavka
\item druga stavka     2. druga stavka
...
\end{enumerate}
```

Nenumerisanu listu dobijamo sledećom naredbom:

```
\begin{itemize}
\item prva stavka      • prva stavka
\item druga stavka     • druga stavka
\item[-] a može i crtica - a može i crtica
...
\end{itemize}
```

Opisnu listu dobijamo na sledeći način:

```
\begin{description}
\item{(a)} prva stavka   (a) prva stavka
\item{(b)} druga stavka  (b) druga stavka
...
\end{description}
```

Pri čemu umesto (a) i (b) mogu stajati proizvoljne reči.

Tabele pravimo naredbom

```
\begin{tabular}{specifikacija tabele}
```

pri čemu se navode elementi ćelija tabele sleva na desno i odozgo na dole. Prelaz iz jedne kolone u drugu se označava sa & a prelazak u novi red sa \\. Ako želimo horizontalnu liniju dobijamo je sa \hline. Kada završimo sa navođenjem svih elemenata tabele kucamo naredbu \end{tabular}. U *specifikaciji tabele* navodimo slova *r*, *l*, *c* u zavisnosti od toga da li je odgovarajuća kolona desno poravnana, levo poravnana ili centrirana. Pri tome treba da imamo isti broj slova koja specifikuju poravnanje kao i kolona. Ako želimo vertikalne linije njih navodimo u specifikaciji tabele stavljanjem znaka | između slova koja određuju poravnanje kolone ili oko njih u zavisnosti od toga kako želimo, i da li uopšte želimo, da ograničimo tabelu.

Na primer,


```
\begin{tabular}{|ccc|}  
\hline  
1 & 2 & 3 \\  
4 & 5 & 6 \\  
\hline  
\end{tabular}
```

1	2	3
4	5	6

Da bi u dokument ubacili *sliku u EPS* formatu, moramo na početku dokumenta uključiti paket `\usepackage{graphicx}` i onda na mestu gde želimo da nam se pojavi slika kucamo naredbu

`\includegraphics[width=w1, height=h1]{slika.eps}` pri čemu je zadanje širine i visine slike opciono.

Komanda `\tableofcontents` proizvodi *sadržaj* na mestu gde je izdata.

Čas 2

2.1 Formalni jezici i formalne gramatike

2.1.1 Jezici: zadaci

1

Zadatak 1 Dokazati da ne postoji nijedna reč x azbuke $\Sigma = \{a, b\}$ za koju važi jednakost $xa = bx$.

Rešenje:

Izvedimo dokaz matematičkom indukcijom po dužini reči x .

1° Za praznu reč e ne važi jednakost $ea = be$ (jer je $a \neq b$), pa tvrdjenje važi za reč x dužine 0.

Za $|x| = 1$, važi $x = a$ ili $x = b$. Međutim, kako je $aa \neq ba$ i $ba \neq bb$, sledi da tvrdjenje važi za reči x dužine 1.

2° Pretpostavimo da tvrdjenje važi za sve reči dužine $n - 2$ i dokažimo da onda važi i za reči dužine n .

Pretpostavimo suprotno – da postoji reč x dužine n takva da važi $xa = bx$. Dakle, reč x počinje slovom b , a završava se slovom a , pa reč x može biti napisana u obliku $x = bya$, gde je y reč dužine $|x| - 2 = n - 2$. Onda važi:

$$byaa = bbya \quad ,$$

odakle na osnovu zakona skraćivanja sledi

$$ya = by \quad ,$$

tj. reč y je rešenje zadate jednačine i $|y| = n - 2$, što je u kontradikciji sa induktivnom pretpostavkom, odakle sledi da ne postoji reč x dužine n takva da važi $xa = bx$.

Tvrdjenje je dokazano za reči dužine 0 i 1, pa, iz dokazanog induktivnog koraka, sledi da tvrdjenje važi za sve prirodne brojeve, čime je dokazano da ne postoji nijedna reč x azbuke $\Sigma = \{a, b\}$ za koju važi jednakost $xa = bx$.

Zadatak 2 Rešiti nad azbukom $\Sigma = \{a, b, c\}$ jednačinu po x : $ax = xa$.

¹Zasnovano na materijalu "Teorija algoritama jezika i automata" Predraga Janičića

Rešenje:

Skup rešenja jednačine je skup reči oblika a^n , ($n \geq 0$).

($\supseteq$): Za svako $n \geq 0$, reč a^n jeste rešenje, jer $aa^n = a^{n+1} = a^n a$.

($\subseteq$): Dokaz indukcijom po dužini reči x , da je svako rešenje oblika a^n ($n \geq 0$).

(1°) Za $|x| = 0$ tj. $x = e$ i $ax = xa$ važi $x = a^0$.

Za $|x| = 1$ iz $ax = xa$ sledi $x = a$, tj. $x = a^1$.

(2°) Pretpostavimo da je tvrdjenje tačno za sve reči dužine $n-2$ i dokažimo da je tačno i za reči dužine n . Neka je $|x| = n$ i neka je $ax = xa$. Dakle, reč x počinje i završava se slovom a , pa je $x = aya$, gde je y reč nad zadatom azbukom dužine $n-2$. Skraćivanjem se iz $aaya = aya$ dobija $ay = ya$, pa je reč y rešenje zadate jednačine dužine $n-2$. Na osnovu induktivne pretpostavke, reč y je oblika a^n , $n \geq 0$, pa je reč x oblika a^{n+2} , $n \geq 0$.

Dakle, svako rešenje date jednačine je oblika a^n ($n \geq 0$).

Dakle, skup rešenja jednačine je skup reči oblika a^n ($n \geq 0$).

2.1.2 Gramatike i jezici: zadaci

2

Zadatak 3 Odrediti jezik generisan gramatikom $G = (N, \Sigma, P, S)$, gde je $N = \{S\}$,

$\Sigma = \{a, b\}$,

$P = \{S \rightarrow aS \text{ (1°)}, S \rightarrow b \text{ (2°)}\}$.

Rešenje:

(Primer izvodjenja u gramatici G : $S \xrightarrow{1^\circ} aS \xrightarrow{1^\circ} aaS \xrightarrow{1^\circ} aaaS \xrightarrow{2^\circ} aaab$)

Dokažimo da važi:

$$L(G) = \{a^n b \mid n \in \mathbf{N}_0\}.$$

$\subseteq$: Svaka završna reč koja se izvodi u gramatici G je oblika $a^n b, n \in \mathbf{N}_0$.

Dokažimo indukcijom jače tvrdjenje:

Lema 1 Sve reči koje mogu biti izvedene u gramatici G su oblika $a^n S$ ili oblika $a^n b$ ($n \in \mathbf{N}_0$).

Dokaz indukcijom po dužini izvodjenja:

(1) $k = 0$: U izvodjenju nije primenjeno nijedno pravilo izvodjenja, tj. izvodjenje je trivijalno; za početni simbol S , dakle, dobija se takodje reč S . Kako je $S = a^0 S$, tvrdjenje važi za $k = 0$.

(2) Pretpostavimo da tvrdjenje važi za sva izvodjenja čija je dužina manja od k i dokažimo da važi i za izvodjenja dužine k .

Neka je $S \Rightarrow^k w$. Neka je $S \Rightarrow^{k-1} w' \Rightarrow w$. Reč w' je izvedena izvodjenjem dužine $k-1$, pa je, na osnovu induktivne hipoteze, w' oblika $a^n S$ ili oblika

²Zasnovano na materijalu "Teorija algoritama jezika i automata" Predraga Janičića

$a^n b$ ($n \in \mathbf{N}_0$). Reč w se netrivialno (izvodjenjem dužine 1) izvodi iz reči w' , pa w' mora da ima nezavršnih simbola. Dakle, w' je oblika $a^n S$. Ako je u k -tom koraku primenjeno pravilo 1° , onda je w oblika $a^{n+1} S$, a ako je primenjeno pravilo 2° , onda je w oblika $a^n b$.

Dakle, sve reči koje mogu biti izvedene u gramatici G su oblika $a^n S$ ili oblika $a^n b$ ($n \in \mathbf{N}_0$), što je i trebalo dokazati.

Na osnovu leme, svi članovi izvodjenja (sve reči koje mogu biti izvedene u gramatici G) su oblika $a^n S$ ili oblika $a^n b$ ($n \in \mathbf{N}_0$), pa i završne reči – reči bez nezavršnih simbola. Završne reči ne mogu biti oblika $a^n S$ (jer je S nezavršni simbol), pa su sve završne reči oblika $a^n b$ ($n \in \mathbf{N}_0$). Dakle, $L(G) \subseteq \{a^n b \mid n \in \mathbf{N}_0\}$.

$\supseteq$: Svaka reč $a^n b$, $n \in \mathbf{N}_0$, može biti izvedena u gramatici G .

Za proizvoljno n ($n \in \mathbf{N}_0$) reč $a^n b$ može biti izvedena u gramatici G :

$$S \xrightarrow{1^\circ} aS \xrightarrow{1^\circ} \underbrace{\dots \xrightarrow{1^\circ}}_n a^n S \xrightarrow{2^\circ} a^n b$$

Zadatak 4 Odrediti jezik generisan gramatikom $G = (N, \Sigma, P, S)$, gde je $N = \{S\}$,

$\Sigma = \{a, b\}$,

$P = \{S \rightarrow aSb \ (1^\circ), S \rightarrow e \ (2^\circ)\}$.

Zadatak 5 Odrediti gramatiku koja generiše jezik $W = \{a^{2i}b^i \mid i > 0\}$.

Rešenje:

$G = (N, \Sigma, P, S)$, gde je $N = \{S\}$,

$\Sigma = \{a, b\}$,

$P = \{S \rightarrow aab \ (1^\circ), S \rightarrow aaSb \ (2^\circ)\}$

Dokažimo da važi $W = L(G)$.

$\subseteq$: Neka je $w \in W$, tj. neka je $w = a^{2i}b^i$, gde je $i > 0$.

w se može izvesti u gramatici G .

$$S \xrightarrow{2^\circ, i-1} (aa)^{i-1} Sb^{i-1} \xrightarrow{1^\circ} (aa)^i b^i.$$

Dakle, $W \subseteq L(G)$.

$\supseteq$: Indukcijom se može dokazati da je svaki član izvodjenja w oblika $a^{2i}Sb^i$ ($i \geq 0$) ili oblika $a^{2i}b^i$ ($i > 0$). Završna reč w ne može da sadrži simbol S , pa je oblika $a^{2i}b^i$ ($i > 0$), odakle sledi $W \supseteq L(G)$.

Zadatak 6 Odrediti gramatiku koja generiše jezik $W = \{a^n b^{\lfloor \frac{n}{2} \rfloor} \mid n \geq 0\}$.

Zadaci za domaći:

Zadatak 7 Dokazati da ne postoji nijedna reč y azbuke $\Sigma = \{c, d\}$ za koju važi jednakost $cy = yd$.

Zadatak 8 Rešiti nad azbukom $\Sigma = \{a, b, c\}$ jednačinu po y : $yb = by$.

Zadatak 9 *Odrediti jezik generisan gramatikom $G = (N, \Sigma, P, S)$, gde je $N = \{S\}$,
 $\Sigma = \{a, b\}$,
 $P = \{S \rightarrow Sb \text{ (1°)}, S \rightarrow a \text{ (2°)}\}$.*

Zadatak 10 *Odrediti gramatiku koja generiše jezik $W = \{b^i a^{3i} \mid i \geq 0\}$.*

Zadatak 11 *Odrediti gramatiku koja generiše sve palindrome nad azbukom $\Sigma = \{a, b\}$.*

Čas 3

3.1 Pozicioni brojni sistemi - konverzije

Pozicioni brojni sistemi su oni u kojima se težina cifre (njen udeo u celokupnoj vrednosti broja) određuje na osnovu njene pozicije u broju (što veća pozicija to je veći i udeo u vrednosti broja). Dekadni brojni sistem je pozicioni, dok rimski brojevi predstavljaju sistem koji nije pozicioni.

Kako su računari zasnovani na binarnoj aritmetici a mi smo navikli da radimo sa dekadnim brojevnim sistemom potrebno je obezbediti prevođenje brojeva iz sistema sa osnovom 10 u sistem sa osnovom 2 i obratno.

Da bi to uradili prvo ćemo posmatrati opštiji problem prevođenja brojeva iz sistema sa proizvoljnom osnovom b u sistem sa osnovom 10.

U bazi sa osnovom 10, na koju smo mi navikli, cifre koje koristimo su $0, 1, \dots, 9$ odnosno od 0 do $10 - 1$. Znači u proizvoljnoj bazi B korišćemo cifre od 0 do $B - 1$.

Najčešće korišćene baze (sem 10) su stepeni dvojke: 2, 8, 16. U sledećoj tabeli su prikazani nazivi odgovarajućih brojevnih sistema zajedno sa ciframa koje se u njima koriste.¹

Naziv	Osnova	Cifre
Dekadni	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Binarni	2	0, 1
Oktalni	8	0, 1, 2, 3, 4, 5, 6, 7
Heksadekadni	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

3.2 Prebacivanje iz sistema sa osnovom b u dekadni brojni sistem

Pozicioni brojni sistemi imaju svojstvo da niz od n cifara u sistemu sa osnovom B

$$\delta_n = d_1 d_2 \dots d_n$$

predstavlja broj

¹pri čemu u heksadekadnom sistemu slova A-F imaju redom vrednosti od 10 – 15

$$\tilde{\delta}_n = \sum_{i=1}^n \tilde{d}_i * B^{n-i} = (\dots (\tilde{d}_1 * B + \tilde{d}_2) * B + \dots + \tilde{d}_{n-1}) * B + \tilde{d}_n$$

u dekadnom brojnom sistemu, pri čemu $\tilde{d}_i$ predstavlja numeričku vrednost karaktera d_i (odnosno za ako je $d_i = A$ onda je $\tilde{d}_i = 10$).

Na ovaj način su jedinstveno predstavljeni svi brojevi od 0 do $B^n - 1$. Ako prvih i cifara označimo sa $\delta_i = d_1 \dots d_i$ onda važi (za $\delta_0 = 0$)

$$\tilde{\delta}_i = \tilde{\delta}_{i-1} * B + \tilde{d}_i, \quad 0 \leq \tilde{d}_i < B$$

odatle takođe možemo da zaključimo da je:

$$\tilde{\delta}_{i-1} = \tilde{\delta}_i \text{ div } B, \quad d_i = \tilde{\delta}_i \text{ mod } B \quad (3.1)$$

Znači, ako pretpostavimo da su cifre broja u sistemu B redom $d_1, \dots, d_n$ onda se njegova vrednost u dekadnom sistemu (označimo je sa x) može izračunati na sledeći način:

1. Neka je $x = 0$ i neka je i indeks tekuće cifre, $i = 1$ na početku.
2. Sada uračunavamo tekuću cifru u vrednost broja: $x = x * B + d_i$, $i = i + 1$
3. Ako je $i > n$ znači da smo uračunali sve cifre broja i da smo dobili vrednost x , inače treba da uračunamo sledeću cifru i vraćamo se na korak 2.

Primer 1 *Prevodenje iz osnova 2, 16 i 8 u osnovu 10:*

$$\begin{aligned} (1101)_2 &= 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = (13)_{10} \\ (1101)_{16} &= 1 * 16^3 + 1 * 16^2 + 0 * 16^1 + 1 * 16^0 = 4096 + 256 + 1 = (4353)_{10} \\ (F9A)_{16} &= F * 16^2 + 9 * 16^1 + A * 16^0 = 15 * 16^2 + 9 * 16^1 + 10 * 16^0 = \\ &= 3840 + 144 + 10 = (3994)_{10} \end{aligned}$$

Zadatak 12 *Prebacite sledeće brojeve u dekadni brojni sistem (indeks predstavlja osnovu u kojoj su brojevi zapisani): $(110111100)_2$, $(77)_8$, $(FFFF)_{16}$*

3.3 Prebacivanje iz dekadnog brojnog sistema u sistem sa osnovom b

Inverzni algoritam koji računa niz cifara $d_1 \dots d_n = \delta_n$ koje predstavljaju broj $\tilde{\delta}_n$ u pozicionom sistemu sa osnovom B se dobija primenom jednačina 3.1.

Za dati broj $\tilde{\delta}_i = x$ ($0 \leq x < B^n$), njegova poslednja cifra u datoj reprezentaciji (sa osnovom B) se dobija kao ostatak pri deljenju broja x sa B . Da bi dobili ostale cifre potrebno je da izračunamo celobrojni količnik pri deljenju x sa B i da na njega primenimo isti algoritam.

1. Krećemo od broja x i u svakom koraku računamo $(n - i)$ -tu cifru. Na početku $i = 0$.
2. $c_i = x \text{ mod } B$, $x = x \text{ div } B$, $i = i + 1$.

3. Ako je $x = 0$ dobili smo cifre obrnutim redosledom (odnosno $d_i = c_{n-i}$),
inače se vraćamo na korak 2.

Primetimo da ovaj algoritam možemo koristiti i za izdvajanje cifara dekadnog broja (proverite!).

Primer 2 *Prevođenje iz dekadnog u binarni brojni sistem*

$$(26)_{10} = (?)_2$$

Rešenje:

$$26 / 2 = 13 \text{ i ostatak } 0$$

$$13 / 2 = 6 \text{ i ostatak } 1$$

$$6 / 2 = 3 \text{ i ostatak } 0$$

$$3 / 2 = 1 \text{ i ostatak } 1$$

$$1 / 2 = 0 \text{ i ostatak } 1$$

Dakle, rešenje je broj čije se cifre dobijaju tako što se ostaci dobijeni prethodnim postupkom pročitaju obrnutim redosledom tj. $(11010)_2$

Zadatak 13 Odredite binarnu reprezentaciju sledećih brojeva: $(54)_{10}$, $(126)_{10}$, $(332)_{10}$.

Primer 3 *Prevođenje iz dekadnog u oktalni brojni sistem*

$$(181)_{10} = (?)_8$$

Rešenje:

$$181 / 8 = 22 \text{ i ostatak } 5$$

$$22 / 8 = 2 \text{ i ostatak } 6$$

$$2 / 8 = 0 \text{ i ostatak } 2$$

Dakle, rešenje je broj čije se cifre dobijaju tako što se ostaci dobijeni prethodnim postupkom pročitaju obrnutim redosledom tj. $(265)_8$

Zadatak 14 Odredite oktalnu prezentaciju sledećih brojeva: $(67)_{10}$, $(336)_{10}$, $(442)_{10}$

Primer 4 *Prevođenje iz dekadnog u heksadekadni brojni sistem*

$$(181)_{10} = (?)_{16}$$

Rešenje:

$$181 / 16 = 11 \text{ i ostatak } 5$$

$$11 / 16 = 0 \text{ i ostatak } 11(\text{heksadekadna cifra } B)$$

Dakle, rešenje će biti broj čije se cifre dobiju tako što se ostaci dobijeni prethodnim postupkom pročitaju unazad tj. $(B5)_{16}$

Zadatak 15 Odredite heksadekadnu prezentaciju sledećih brojeva: $(48)_{10}$, $(1336)_{10}$, $(332)_{10}$.

3.4 Rad sa realnim brojevima

Kada radimo sa realnim brojevima možemo posebno posmatrati ceo deo broja i razlomljeni deo broja. Kako smo do sada radili sa celim brojevima posmatrajmo brojeve oblika $x = 0.d_1d_2 \dots d_n$, za koje je $0 \leq x < 1$.

3.4.1 Prevođenje realnih brojeva iz sistema sa osnovom B u dekadni brojni sistem

Neka je $\delta = 0.d_1d_2 \dots d_n$ razlomljeni deo broja x u sistemu sa osnovom B . Tada će njegova vrednost u dekadnom sistemu biti:

$$\tilde{\delta} = \sum_{i=1}^n d_i * B^{-i} = \frac{1}{B}(\tilde{d}_1 + \frac{1}{B}(\tilde{d}_2 + \dots + \frac{1}{B}\tilde{d}_n) \dots) \quad (3.2)$$

Pri čemu je $\tilde{d}_i$ numerička vrednost "cifre" d_i . Odatle dobijamo sledeći algoritam:

1. Neka je $x = 0$ (dekadna vrednost broja), $i = 1$ indeks tekuće cifre koju uračunavamo u vrednost broja i $f = \frac{1}{B}$ tekući koeficijent sa kojim množimo cifru.
2. $x = x + d_i * f$, $i = i + 1$, $f = f * \frac{1}{B}$.
3. Ako je $i > n$ znači da smo uračunali sve cifre i da se u x nalazi dekadna vrednost broja, inače se vraćamo na korak 2.

Primer 5 $(0.1101)_2 = 1 * 2^{-1} + 1 * 2^{-2} + 0 * 2^{-3} + 1 * 2^{-4} = (0.6875)_{10}$

Zadatak 16 *Prebacite sledeće brojeve u dekadni brojni sistem (indeks predstavlja osnovu u kojoj su brojevi zapisani): $(0.1011)_2$, $(0.77)_8$, $(0.FF)_{16}$*

3.4.2 Prevođenje realnih brojeva iz dekadnog brojnog sistema u sistem sa osnovom B

Neka je $\delta_i = 0.d_i \dots d_n$. Na osnovu jednačine 3.2 vidimo da važi:

$$\tilde{\delta}_i = \frac{1}{B}(\tilde{d}_i + \tilde{\delta}_{i+1}), \quad 0 \leq \tilde{\delta}_i < 1$$

Odnosno

$$\tilde{d}_i = \text{trunc}(B * \tilde{\delta}_i)$$

$$\tilde{\delta}_{i+1} = B * \tilde{\delta}_i - \tilde{d}_i$$

pri čemu je $\text{trunc}(x)$ ceo deo broja x . Odatle direktno vidimo na koji način možemo izračunati cifre broja u sistemu sa osnovom B i dobijamo sledeći algoritam ²:

1. Neka je x broj čiji zapis određujemo, $i = 1$ indeks tekuće cifre koju računamo.
2. $d_i = \text{trunc}(B * x)$.
3. $x = B * x - d_i$, $i = i + 1$.

²pri čemu primetimo da iz petlje izlazimo kada dobijemo željeni broj cifara a ne kada x postane 0 iz razloga što radimo sa realnim brojevima koji ne moraju imati konačan zapis

4. Ako je $i = n$ dobili smo n cifara razlomljenog dela broja, inače se vraćamo na korak 2.

Ovim algoritmom se cifre dobijaju u željenom redosledu, odnosno od prve ka poslednjoj.

Primer 6 *Odrediti binarni zapis broja $x = (0.867)_{10}$ na 4 decimale.*

$0.867 * 2 = 1.734$, *ceo deo 1*

$0.734 * 2 = 1.468$, *ceo deo 1*

$0.468 * 2 = 0.936$, *ceo deo 0*

$0.936 * 2 = 1.872$, *ceo deo 1*

Dakle rešenje se dobija tako što se cifre čitaju onim redosledom kojim su dobijene tj. $(0.1101)_2$

3.5 Direktno prevođenje iz binarnog u heksadekadni sistem

Za kodiranje heksadekadnih cifara dovoljne su binarne reči dužine četiri ($16 = 2^4$).

Heksadekadna cifra	Binarni kod	Heksadekadna cifra	Binarni kod
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Primetimo da je na ovaj način svakoj heksadekadnoj cifri jedinstveno dodeljen kod dužine četiri u binarnom sistemu što nam omogućava da obavljamo direktno prevođenje iz binarnog u heksadekadni sistem na sledeći način:

Binarne cifre se grupišu u grupe od 4 cifre, počev od bitova najmanje težine. Ako ukupan broj bitova nije deljiv sa četiri, onda se dopisuje potreban broj vodećih nula (one su bez uticaja na promenu vrednosti originalnog zapisa).

Primer 7 $(1111011100001101010000)_2 = (0011\ 1101\ 1100\ 0011\ 0101\ 0000)_2 = (3DC350)_{16}$

Zadatak 17 *Odredite heksadekadni zapis sledećeg binarnog broja $(11010100100)_2$*

3.6 Direktno prevođenje iz binarnog u oktalni sistem

Za kodiranje oktalnih cifara dovoljne su binarne reci dužine tri ($8 = 2^3$).

Oktalna cifra	Binarni kod	Oktalna cifra	Binarni kod
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

Sada smo svakoj oktalnoj cifri jedinstveno dodelili binarni kod dužine tri što nam omogućava direktno prevođenje. Binarne cifre se grupišu grupe od po 3 cifre, počev od bitova najmanje težine. Ako ukupan broj bitova nije deljiv sa tri, onda se dopisuje potreban broj vodećih nula.

Primer 8 $(11111010001010)_2 = (011\ 111\ 010\ 001\ 010)_2 = (37212)_8$

Zadatak 18 *Odredite oktalni zapis sledećeg binarnog broja $(11010100100)_2$*

Čas 4

1

4.1 Reprezentacija znakovnih podataka

4.1.1 Tekst je niz karaktera

- Iako obično tekst zamišljamo kao dvodimenzioni objekat, u računarima se tekst predstavlja kao jednodimenzioni (linearni) niz karaktera.
- Potrebno je, dakle, uvesti specijalne karaktere koji označavaju prelazak u novi red, tabulator, kraj teksta i slično

4.1.2 Zapis karaktera u računar

- Računari su zasnovani na binarnoj aritmetici
- Cele brojeve je moguće predstaviti u binarnom sistemu
- Osnovna ideja je svakom karakteru pridružiti određeni ceo broj na unapred dogovoreni način
- Ove brojeve zovemo kodovima karaktera (character codes)

4.1.3 Skupovi karaktera

- Koliko karaktera želimo da predstavimo u računarima? Tokom razvoja računarstva broj karaktera je postajao sve veći
- Pošto je u početku razvoja englesko govorno područje bilo dominantno osnovno je bilo predstaviti sledeće karaktere :
 - Velika slova engleskog alfabeta : A,B,...,Z
 - Mala slova engleskog alfabeta : a,b,...,z
 - Cifre : 0,1,...,9
 - Interpunkcijske znake : ., ; i slično
 - Kontrolne znake : kraj reda, tabulator i slično

¹Zasnovano na materijalu "Zapis tekstova u računar" Filipa Marića

- Standardni karakterski kodovi: Sedamdesetih godina su se pojavile tabele standardnih karakterskih kodova dovoljne za zapis pomenutih karaktera
Najpoznatiji su
 - EBCDIC IBM-ov standard, pogodan za bušene kartice
 - ASCII Standard iz koga se razvila većina današnjih standarda
- ASCII (American Standard Code for Information Interchange)
ASCII je sedmobitan (broj karaktera koji je njime predstavljen je $128 = 2^7$)
- Npr. kod za A je $(41)_{16}$ tj. $0x41$ što je $(65)_{10}$ tj. $(1000001)_2$,
kod za a je $(61)_{16}$ tj. $0x61$ što je $(95)_{10}$ tj. $(1100001)_2$.
- PRIMER: transformisanje malih slova u velika
- Razmak SP se zapisuje kao $(20)_{16}$ što je $(32)_{10}$ tj. $(0100000)_2$.
- **Osobine ASCII koda:** Prvih 32 karaktera (kodovi $0x00-0x1F$) i poslednji karakter (kod $0x7F$) su kontrolni karakteri. To su karakteri bez grafije, kao CR (kod $0x0D$), LF (kod $0x0A$).
- Prvi karakter sa grafijom je blanko (kod $0x20$). Njegova grafija je belina.
- Skup velikih slova A-Z (kodovi $0x41-0x5A$), kao i skup malih slova a-z (kodovi $0x61-0x7A$), je u alfabetskom redosledu unutar kolacione sekvencije
($0x41 < 0x42$, prema tome $A < B$ to odgovara alfabetskom redosledu).
- Skup cifara 0–9 (kodovi $0x31-0x39$) je u rastućem brojčanom redosledu unutar kolacione sekvencije
($0x31 < 0x32$, prema tome $1 < 2$ što odgovara brojčanom redosledu).
- Skup velikih slova A-Z (kodovi $0x41-0x5A$), skup malih slova a-z (kodovi $0x61-0x7A$) i skup cifara 0-9 (kodovi $0x31-0x39$) su kontingentni unutar kolacione sekvencije (između slova A i slova Z nema drugih karaktera osim onih koji odgovaraju velim slovima engleske abecede). Sve cifre prethode svim velikim slovima, sva velika slova prethode svim malim slovima u kolacionoj sekvenciji. Specijalni i interpunkcijski znaci su izmešani između njih.
- Kod svakog velikog slova je za 32 (ili $0x20$) manji od koda odgovarajućeg malog slova. Na primer, za slovo E važi da je $0x45 + 0x20 = 0x65$, odnosno, $01000101 + 00100000 = 01100101$. Prema tome, binarni kodovi velih i malih slova razlikuju se samo u jednoj cifri, onoj koja odgovara petom stepenu osnove 2.

4.1.4 Kodiranje ostalih jezika

- Razvojem računarstva se javlja potreba kodiranja tekstova i na drugim jezicima
- Kroz istoriju su postojala mnoga rešenja, od kojih su se neka zadržala, a neka su nestala

4.1.5 Osobine YUSCII koda

- ASCII kod je jedna verzija međunarodnog standarda ISO 646 IRV koji predstavlja međunarodnu referentnu verziju za 7-bitni kod. Ovaj standard propisuje da se pozicijama $0x40$, $0x5B - 0x5E$, $0x60$ i $0x7B - 0x7E$ ne pridružuje obavezna grafija već se da se one u nacionalnim verzijama standarda i u određenim aplikacijama mogu slobodno koristiti.
- Standard JUS.B1.002 koristi ovih 10 pozicija za kodiranje slova specifičnih za srpsku latinicu Ž, Š, Đ, Ć, Č.
- Yu-ASCII skup zadržava sve navedene osobine ASCII koda osim jedne, a ta je da ni velika ni mala slova nisu u alfabetskom redosledu unutar kolacione sekvencije. Naime, $0x40 < 0x41$, dakle $\text{Ž} < \text{A}$ što ne odgovara alfabetskom redosledu unutar kolacione sekvencije. Nakon slova Ž slede velika slova engleske abecede, zatim slova Š($0x5B$), Đ($0x5C$), Ć($0x5D$), Č($0x5E$), ž($0x60$), zatim slede mala slova abecede, i na kraju slova š($0x7B$), đ($0x7C$), ć($0x7D$), č($0x7E$).

Zadatak 19 Poredati slova vašeg prezimena koristeći YUSCII kodnu šemu.

4.1.6 Kodne strane

- Pod *kodnom* stranom (Code page) tj. *skupom karaktera* (Character set, charset) podrazumevamo uređenu listu karaktera predstavljenih svojim karakterskim kodovima
- Podaci se u računarima obično zapisuju bajt po bajt
- ASCII je sedmobitni standard
- ASCII karakteri se zapisuju tao što se u svakom bajtu bit najveće težine postavi na 0
- To ostavlja prostor za novih 128 karaktera čiji binarni zapis počinje sa 1
- Ovaj prostor se može popuniti na razne načine
- Rešenje nije univerzalno, jer svakako na svetu postoji više od 256 različitih karaktera
- Postavljeni su razni standardi dopunjavanja ovih 128 karaktera
- Svim ovim kodnim stranama je zajedničko prvih 128 karaktera i oni se poklapaju sa ASCII
- Ovako napravljene kodne strane obično omogućuju kodiranje tekstova na više srodnih jezika (obično i geografski bliskih)
- Nama su uglavnom važne kodne strane napravljene za centralno-evropske (Central European) latinice, kao i ćirilicne kodne strane

4.1.7 Kodne strane kod nas

- Najčešće korišćene kodne strane kod nas (Prve dve su delo međunarodne organizacije za standardizaciju (International Standard organization), dok su naredne dve Microsoft-ovi standardi):
 - ISO 8859-2 (Latin2)
 - ISO 8859-5 (Ćirilicna)
 - Windows 1250
 - Windows 1251 (Ćirilicna)
- **Latin 1:** Poželjno je poznavati i osnovnu kodnu stranu ISO 8859-1 (Latin1) jer je veoma često postavljena kao podrazumevana kodna strana. Ona se koristi za zapis tekstova na zapadno evropskim jezicima (Western European)

4.1.8 Višebajtni karakterski kodovi

- Iako navedene kodne strane omogućuju kodiranje tekstova koji nisu na engleskom jeziku nije moguće npr. u istom tekstu mešati ćirilicu i našu latinicu.
- Azijskim jezicima nije dovoljno 256 mesta za zapis svih karaktera.
- Zbog toga se uvode višebajtni karakterski kodovi
- MBCS: Pre svega zbog potreba istočno azijskih korisnika uvedeni su tzv. višebajtni skupovi karaktera tj. Multi-Byte Character Sets (MBCS)
- Ideja je u tome da se najčešće korišćeni karakteri zapisuju koristeći samo jedan bajt, dok se ostali karakteri zapisuju koristeći dva bajta, tj. koristi se mešavina jednobajtnih i dvobajtnih karakterskih kodova (pod UNIX-om nekad čak i trobajtnih)
- Ovo značajno otežava tumačenje podataka
- **ISO 10646** je zamišljen kao 4 bajtni standard. Pri tome se prvih 65536 karaktera koriste kao osnovni višejezični skup karaktera dok je ostali prostor ostavljen kao proširenje za drevne jezike, celokupnu naucnu notaciju i slično.
- **UNICODE:** svakom karakteru dodeljuje dvobajtni kod
- Prvih 128 karaktera se poklapaju sa ASCII standardom, dok su sledećih 128 napravljeni tako da se poklapaju sa Latin1 standardom
- UCS-2: Unicode standard u suštini predstavlja veliku tabelu koja svakom karakteru dodeljuje broj.
- Standardi koji opisuju kako se niske karaktera onda prevode u nizove bajtova se dodatno definišu

- ISO definiše UCS-2 standard koji jednostavno svaki UNICODE karakter prevodi u odgovarajuća dva bajta
- UTF: A Unicode transformation format (UTF) algoritam koji svakom UNICODE karakteru dodeljuje određeni niz bajtova čija dužina varira od 1 do najviše 6.
- UTF je ASCII kompatibilan, što znači da se ASCII karakteri zapisuju pomoću jednog bajta, na standardni način.
- Najčešće korišćena varijanta ovog algoritma je UTF-8 koja je dovoljna za zapis svih dvobajtnih UNICODE karaktera
- Pored ovoga ISO uvodi i UTF-16, UTF-32, kao i standard UCS-4

4.1.9 Karakteri, Glifovi, Fontovi

- Vrlo često se ne pravi jasna razlika između karaktera i njihove grafičke reprezentacije
- Grafičku reprezentaciju karaktera nazivamo glifovima (*glyph*) Skupove glifova nazivamo fontovima (*font*)
- Korespondencija između karaktera i glifova ne mora biti jednoznačna
- Jedan glif može da predstavi više karaktera (ligature)
- Isti karakter može da se predstavlja različitim glifovima u zavisnosti od svoje pozicije u reči
- Za razliku od tradicionalnih fontova koji u sebi sadže glifove za karaktere jedne kodne strane, TrueType fontovi koji podržavaju WGL4 standard sadrže glifove za sve evropske karaktere

Zadatak 20 Zapisati cifru 3 u ASCII kodu.

Rešenje:

Broj 3 se zapisuje kao $(33)_{16}$ tj. $0x33$ što je $(51)_{10}$ tj. $(1010001)_2$

Zadatak 21 Zapisati reč Fakultet u ASCII kodu.

Zadatak 22 Zapisati reči MATF i lišće u kodnim stranama ISO 8859-2, Windows 1250, Windows 1251.

Rešenje:

Reč *MATF* se zapisuje isto u kodnim stranama ISO 8859-2, Windows 1250 i Windows 1251 zato što su njeni karakteri zapravo ASCII karakteri a svim ovim kodnim stranama zajedničko je prvih 128 karaktera i oni se poklapaju sa ASCII kodovima.

Dakle, reč *MATF* se u ovim kodnim stranama zapisuje preko 4 bajta i to $(4D)_{16}$, $(41)_{16}$, $(54)_{16}$, $(46)_{16}$ a to je isto što i $(77)_{10}$, $(65)_{10}$, $(84)_{10}$, $(70)_{10}$ odnosno $(01001101)_2$, $(01000001)_2$, $(01010100)_2$, $(01000110)_2$.

Reč *lišće* sadrži u sebi karaktere š i ć koji nisu ASCII karakteri pa se različito kodiraju u svakoj od kodnih strana.

U kodnoj strani ISO 8859-2 odgovarajući kod je $(6c)_{16}$, $(69)_{16}$, $(b9)_{16}$, $(e6)_{16}$, $(65)_{16}$ a to je isto što i $(108)_{10}$, $(105)_{10}$, $(185)_{10}$, $(230)_{10}$, $(101)_{10}$ odnosno $(01101100)_2$, $(01101001)_2$, $(10111001)_2$, $(11100110)_2$, $(01100101)_2$.

U kodnoj strani Windows 1250 karakteri š i ć se kodiraju sa $(9A)_{16}$, $(E6)_{16}$.

U kodnoj strani Windows 1251 karakteri š i ć se ne mogu kodirati.

Zadatak 23 Šta predstavlja niz kodova 138 65 111 33 u kodnoj strani ISO 8859-2? A u Latin1?

Rešenje:

U kodnoj strani ISO 8859-2 ovaj niz kodova predstavlja |Ao! a u Latin1 ŠAo!

Čas 5

5.1 Specifikacija sintakse programskih jezika, meta jezici

Za opis programskih jezika često se koriste kontekstno-slobodne gramatike. Bekus-Naurova forma (BNF) je konvencija za zapisivanje pravila kontekstno-slobodnih jezika. Proširena Bekus-Naurova forma (EBNF) dodaje određene sintaksičke izraze BNF notaciji i omogućava jednostavniji zapis pravila gramatike.

5.1.1 BNF (Backus-Naur form)

Meta jezik je jezik koji služi da se pomoću njega opiše neki drugi jezik ili isti taj jezik. Tako se na primer služimo srpskim jezikom da bismo opisali gramatiku srpskog jezika. Bitno je razlikovati term meta jezika od terma jezika koji se opisuje.

BNF (Bekus-Naurova forma) je formalni meta jezik za predstavljanje kontekstno-slobodnih gramatika odnosno gramatika programskih jezika.

U BNF se koriste sledeće konvencije za zapisivanje pravila gramatike:

- Umesto simbola $\rightarrow$ koristi se simbol $::=$
- Pomoćni simboli se navode među zagrada $\langle \text{ i } \rangle$
- Vertikalna crta $|$ (izbor) razdvaja desne strane pravila koja odgovaraju istom simbolu sa leve strane pravila
- Prazna niska ϵ se zapisuje kao $\langle \text{empty} \rangle$.

Sledi nekoliko primera BNF-a :

- BNF za cifru:

```
<cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0
```

- BNF za neoznačen ceo broj:

```
<neoznaceniCeoBroj> ::= <cifra> | <cifra> <neoznaceniCeoBroj>
<cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0
```

Drugo pravilo može da se napiše i kao:

```
<neoznaceniCeoBroj> ::= <cifra> | <neoznaceniCeoBroj> <cifra>
```

- BNF za ceo broj:

```
<ceoBroj> ::= <neoznaceniCeoBroj>
           | + <neoznaceniCeoBroj>
           | - <neoznaceniCeoBroj>
<neoznaceniCeoBroj> ::= <cifra> | <cifra> <neoznaceniCeoBroj>
<cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0
```

- BNF za realne brojeve:

```

<realanBroj> ::= - <neoznaceniRealanBroj>
               | + <neoznaceniRealanBroj>
               | <neoznaceniRealanBroj>
<neoznaceniRealanBroj> ::= <neoznaceniCeoBroj>
                           | <neoznaceniCeoBroj> . <neoznaceniCeoBroj>
<neoznaceniCeoBroj> ::= <cifra> | <cifra> <neoznaceniCeoBroj>
<cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0

```

- BNF za identifikator (niska cifara ili slova koja ne sme počinjati cifrom):

```

<identifikator> ::= <slovo>
                  | <identifikator> <slovo>
                  | <identifikator> <cifra>
<cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0

```

Primetimo da ovde ne smemo da zamenimo mesta pomoćnim simbolima (proverite!).

5.1.2 EBNF (Extended Backus-Naur form)

Proširena Backus-Naurova forma ima istu izražajnu moć kao BNF s tim što su u njoj izvršene izmene koje doprinose čitljivosti zapisu pravila. Zapravo, BNF koristi rekursiju (definisanje pomoćnog simbola preko njega samog) da bi se izrazile relacije a EBNF koristi iteraciju.

Konvencije za EBNF su¹:

- Pomoćni simboli se zapisuju velikim početnim slovom
- Završni simboli se zapisuju pod jednostrukim navodnicima ako se sastoje od jednog karaktera, a crnim slogom (engl. *bold*) ako su višeslovni. Npr. '+' ali **div** ili **integer**
- Oble zagrade () se koriste za grupisanje
- Vitičaste zagrade { } ograđuju deo koji se ponavlja 0 ili više puta
- Uglaste zagrade [] opisuju opcionu konstrukciju
- Značenje ostalih oznaka je isto kao kod BNF.

Sve ove konstrukcije mogu biti izražene i u BNF-u što je pokazatelj toga da BNF i EBNF imaju istu izražajnu moć.

Sledi nekoliko primera EBNF-a :

- EBNF za neoznačen ceo broj.

```

NeoznaceniCeoBroj ::= Cifra{Cifra}
Cifra ::= '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '0'

```

- EBNF za ceo broj.

¹Po nekim konvencijama se koriste sledeći sufiksi: ? umesto [], * umesto {}, + sa značenjem jedno ili više pojavljivanja simbola.

```
CeoBroj ::= [ '+' | '-' ] NeoznaceniCeoBroj
NeoznaceniCeoBroj ::= Cifra{Cifra}
Cifra ::= '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '0'
```

Pri čemu se umesto prva dva pravila može pojaviti pravilo:

```
CeoBroj ::= [ '+' | '-' ] Cifra{Cifra}
```

- EBNF za realne brojeve.

```
RealanBroj ::= [ '+' | '-' ] Cifra{Cifra} [ '.' Cifra{Cifra} ]
Cifra ::= '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '0'
```

- EBNF za identifikator.

```
Identifikator ::= Slovo{Slovo|Cifra}
```

Zadatak 24 Napisati BNF/EBNF za aritmetički izraz.

Rešenje:

BNF:

```
<izraz> ::= <izraz> + <term> | <izraz> - <term> | <term>
<term> ::= <term> * <faktor> | <term> / <faktor> | <faktor>
<faktor> ::= (<izraz>) | <neoznaceniCeoBroj>
```

EBNF:

```
Izraz ::= Term{('+' | '-')Term}
Term ::= Faktor{('*' | '/')Faktor}
Faktor ::= '(' Izraz ')' | NeoznaceniCeoBroj
```

Zadatak 25 (DOMAĆI) Napisati BNF/EBNF za klauzu (nad nekim fiksnim skupom iskaznih slova)

Zadatak 26 Napisati BNF/EBNF za klauzu dužine npr. 5 (nad nekim fiksnim skupom iskaznih slova).

Pomoć:

```
<klauza> ::= <literal> ∨ <literal> ∨ <literal> ∨ <literal> ∨
<literal>
```

5.1.3 Sintaksni dijagrami

Sintaksni dijagrami predstavljaju pravila gramatike posredstvom grafova. Imaju istu izražajnu moć kao BNF i EBNF. Konstruišu se na sledeći način:

- Svakom pravilu gramatike dodeljuje se po jedan graf koji nosi ime pomoćnog simbola sa leve strane pravila

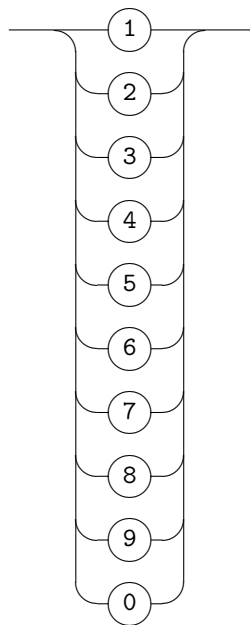
- Čvorovi grafa su obeleženi simbolima desne strane pravila i to pravougaonikom ako odgovaraju pomoćnom simbolu gramatike, a elipsom ako su pridruženi završnom simbolu gramatike. Lukovi u grafu povezuju čvorove na način opisan samim gramatičkim pravilom.
- Jezik predstavljen grafom se dobija obilaskom puteva u grafu.

Pogledajmo kako izgledaju sintaksni dijagrami nekoliko već pomenutih gramatika:

- Sintaksni dijagram za *Cifra*:

Cifra ::= '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '0'

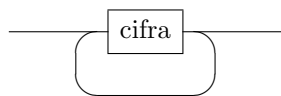
Cifra



- Sintaksni dijagram za *NeoznaceniCeoBroj*:

NeoznaceniCeoBroj ::= *Cifra*{*Cifra*}

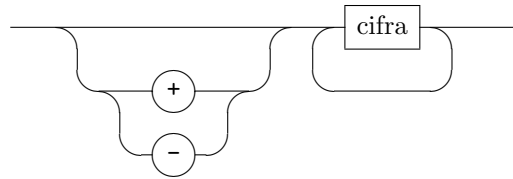
NeoznaceniCeoBroj



- Sintaksni dijagram za *CeoBroj*:

CeoBroj ::= ['+' | '-'] *Cifra*{*Cifra*}

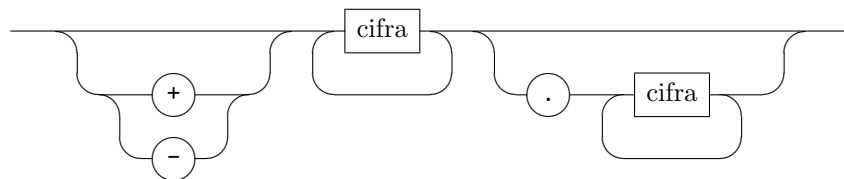
CeoBroj



- Sintaksni dijagram za RealanBroj:

$\text{RealanBroj} ::= [\text{'+'} | \text{'-'}] \text{Cifra}\{\text{Cifra}\} [\text{'.'} \text{Cifra}\{\text{Cifra}\}]$

RealanBroj



- Sintaksni dijagram za Identifikator:

$\text{Identifikator} ::= \text{Slovo} \{ \text{Slovo} \mid \text{Cifra} \}$

Identifikator

