**Zadatak broj 3028:**  **Programski ostvariti parser Pomakni/Pronađi.** Ulaz u program je tablica parsera u slobodnom formatu. Izgraditi tablice parsera za nekoliko primjera gramatika i pokazati ispravnost rada ostvarenog parsera.

**1. Uvod**

**1.1 *Parsiranje od dna prema vrhu***

Parsiranje tehnikom Pomakni/Pronađi je jedno od parsiranja od dna prema vrhu. Parsiranje od dna prema vrhu započinje gradnju generativnog stabla listovima. Listovi stabla označeni su završnim znakovima gramatike. Gradnja stabla nastavlja se primjenom desnih strana produkcija gramatike na prethodno izgrađene čvorove. Budući da se u svakom koraku reducira ili ostaje isti broj znakova u međunizu, produkcije se u postupku parsiranja od dna prema vrhu nazivaju redukcije. Gradnja generativnog stabla nastavlja se do korijena stabla. Ako se izgradi generativno stablo koje ime korijen označen početnim nezavršnim znakom gramatike, niz w je u jeziku koji generira zadana gramatika. Znakovi niza w čitaju se slijedno slijeva nadesno. Na temelju pročitanih znakova izabere se odgovarajuća redukcija. Učinkovitost ovisi o preciznosti kojom je moguće odrediti koju je redukciju potrebno primijeniti u datom koraku parsiranja.

Tijek parsiranja opisuje se na sljedeći način :

\* \* w 🡸 α x 🡸 <*S*>

gdje je w niz završnih znakova , α x je međuniz dobiven postupkom generiranja zamjenom krajnje desnog nezavršnog znaka i <*S*> je početni nezavršni znak gramatike. Međuniz α x čine niz završnih znakova gramatike i niz završnih i nezavršnih znakova α.

Parsiranje od dna prema vrhu moguće je ostvariti primjenom stoga i ulaznog spremnika. Parser čita sve znakove koji su na stogu. Na stog se spremaju završni I nezavršni znakovi gramatike. U ulaznom spremniku je niz w.

***Parser***

**a w'**

Ulazni spremnik Stog

Parsiranje započinje praznim stogom, odnosno na stogu je samo oznaka dna stoga . Parser čita znak a koji je krajnje lijevi znak niza w = aw'.

Na temelju pročitanog znaka i znaka na vrhu stoga, parser primijeni jednu od četiri akcije.

1) Akcija *Pomakni*

Pročitani znak se stavlja na vrh stoga, a kazaljka za čitanje niza znakova miče se za jedan znak udesno.

2) Akcija *Reduciraj*

Ako je na vrhu stoga uzorak za zamjenu, onda se primjenjuje akcija *Reduciraj*. S vrha stoga uzmu se znakovi desne strane produkcije,a na vrh stoga stavi se nezavršni znak lijeve strane produkcije. Primjena akcije *Reduciraj* zahtijeva uspoređivanje znakova vrha stoga i znakova desnih strana produkcija.

3) Akcija *Prihvati*

Ako su pročitani svi znakovi ulaznog niza, odnosno ako parser pročita oznaku kraja ulaznog niza, a na stogu je početni nezavršni znak <*S*> i oznaka dna stoga , parser prihvati niz w. Niz w je u jeziku koji generira zadana gramatika ako i samo ako ispod početnog nezavršnog znaka nema nijednog drugog znaka osim oznake dna stoga .

4) Akcija *Odbaci*

Ako nije moguća daljnja gradnja stabla, primjenjuje se akcija Odbaci. Niz w se ne prihvaća, odnosno niz w nije u jeziku koji generira zadana gramatika. Parser primjenjuje akciju *Odbaci* u dva slučaja:

a) Ako nije zadana nijedna akcija za pročitani znak ulaznog niza i znak vrha stoga, onda se primjenjuje akcija *Odbaci*.

b) Akcija *Odbaci* primjenjuje se tijekom izvođenja akcije *Reduciraj*. Akcija *Reduciraj* uspoređuje znakove vrha stoga i znakove desnih strana produkcija. Ako znakovi na vrhu stoga nisu jednaki znakovima nijedne strane produkcije, onda parser primjeni akciju *Odbaci*.

**1.2 *Parsiranje tehnikom Pomakni – Pronađi***

Tablica *Pomakni/Pronađi* gradi se na temelju vrijednosti dviju relacija: *IspodZnaka*  i *ReduciranZnakom*.

**1.2.1 *Relacija IspodZnaka***

Neka je zadana gramatika koja nema ε-produkcije i prazne nezavršne znakove. Završni i nezavršni znak gramatika *A* jest znak stoga. Završni znak gramatike *x* je znak ulaznog niza. Za znakove *A* i *x* vrijedi relacija :

*IspodZnaka(A , x)*

ako i samo ako je ispunjen jedan od sljedeća dva uvjeta :

1) Znak *A* je izravno ispred znaka *B* na desnoj strani barem jedne produkcije zadane gramatike, odnosno vrijedi relacija *IzravnoIspredZnaka(A , B),* a znak *x* započinje barem jedan niz generiran iz znaka *B*, odnosno vrijedi *x* Є *ZAPOČINJE(B).*

2) *A* je oznaka dna stoga i *x* Є *ZAPOČINJE(*<*S*>*),* gdje je <*S*> početni nezavršni znak gramatike.

**Računanje skupova *ZAPOČINJE* i relacije *IzravnoIspredZnaka***

Skup *ZAPOČINJE* :

Završni znak *b* je u skupu *ZAPOČINJE*(<*X*>) nezavršnog znaka <*X*> ako i samo ako vrijedi relacija : *ZapočinjeZnakom*(<*X*> , *b*)

Relacija *ZapočinjeZnakom*(<*X*> , *b*) vrijedi samo ako je iz znaka nezavršnog znaka <*X*> moguće generirati niz koji počinje znakom b.

Relacija *IzravnoIspredZnaka :*

Za znakove gramatike A i B vrijedi relacija :

*IzravnoIspredZnaka*(*A* , *B*)

ako i samo ako je barem jedna od zadanih produkcija gramatike oblika:

C 🡺 α A β B γ

gdje je β niz praznih znakova, a α i γ su proizvoljni nizovi znakova gramatike.

Računanje relacije *IzravnoIspredZnaka* je pojednostavljeno , jer nema ε-produkcija niti praznih znakova.

**1.2.2 *Relacija ReduciranZnakom***

Neka je zadana gramatika koja nema ε-produkcije i prazne nezavršne znakove. Završni ili nezavršni znak gramatike *A* je znak stoga. Završni znak gramatike *x* je znak ulaznog niza. Za znakove *A* i *x* vrijedi relacija :

*ReduciranZnakom*(*A* , *x*)

ako i samo ako je ispunjen jedan od sljedeća dva uvjeta:

1) Znak <*A*> je krajnje desni znak desne strane produkcije <*L*> 🡺 α*A*, a znak x slijed znak <*L*> u barem jednom nizu generiranom iz početnog nezavršnog znaka gramatike <*S*>, odnosno vrijedi *x* Є *SLIJEDI*(<*L*>).

2) A je početni nezavršni znak gramatike <S>, a x je oznaka kraja niza .

**Računanje skupa SLIJEDI**

Završni znak b je u skupu *SLIJEDI*(*A*), ako i samo ako je završni znak b u barem jednoj produkciji koju može generirati zadana gramatika na mjestu iza završnog ili nezavršnog znaka A.

**Primjer 1.1**

Zadana je gramatika sljedećih produkcija:

1 ) <S> 🡺 a <B><A> 2 ) <S> 🡺 <B> b 3 ) <A> 🡺 c <S> 4 ) <B> 🡺 b

Skup *ZAPOČINJE* :

*ZAPOČINJE*(<*S*>) = {*a* , *b*} , *ZAPOČINJE*(<*A*>) = { *c* } , *ZAPOČINJE*(<*B*>) = { *b* } , *ZAPOČINJE*(*a*) = { *a* } , *ZAPOČINJE*(*b*) = { *b* } , *ZAPOČINJE*(*c*) = { *c* }

Na temelju izračunatih vrijednosti skupova *ZAPOČINJE* određuje se relacija *IspodZnaka* :

Prva produkcija : <S> 🡺 a <B> <A>

i vrijednost skupa *ZAPOČINJE*(<*A*>) = { *c* } određuju relaciju *IspodZnaka*(*B* , *c*). Na taj način određujemo i sve ostale relacije *IspodZnaka* :

*IspodZnaka*(*a* , *b*) , *IspodZnaka*(*B* , *b*) , *IspodZnaka*(*c* , *a*) , *IspodZnaka*(*c* , *b*).

Temeljem pravila (2), za oznaku dna stoga određuju se relacije *IspodZnaka*( , *a*) i *IspodZnaka*( , *b*).¸

Skup *SLIJEDI* :

*SLIJEDI*(<*S*>) = { } , *SLIJEDI*(<*A*>) = { } , *SLIJEDI*(<*B*>) = {*b* , *c*} , *SLIJEDI*(*a*) = { *b* }, *SLIJEDI*(*b*) = { *c* , } , *SLIJEDI*(*c*) = {*a* , *b*} .

Na temelju izračunatih vrijednosti skupova *SLIJEDI* određuje se relacija *ReduciranZnakom*.

Relacija *ReduciranZnakom* za nezavršni znak <*S*>. Budući da je <*S*> početni nezavršni znak gramatike, na temelju pravila (2) vrijedi relacija *ReduciranZnakom*(<*S*> , ) . Budući da je nezavršni znak <*S*> krajnje desni znak produkcije (3) ,a nezavršni znak <*A*> lijeve strane te iste proudukcije u skupu *SLIJEDI* ima samo znak ostaje nam samo prethodna relacija *ReduciranZnakom*(<*S*> , ).

Na taj način određujemo i sve ostale relacije *ReduciranZnakom* :

*ReduciranZnakom*(<*A*> , ), *ReduciranZnakom*(*b* , *b*), *ReduciranZnakom*(*b* , *c*) , *ReduciranZnakom(b , ).*

**1.2.1 *Konstrukcija tablice Pomakni/Pronađi***

Zadanu gramatiku koja nema ε-produkcija moguće je parsirati tehnikom Pomakni - Pronađi. Parser primjenom dvodimenzionalne tablice upravlja akcijama na stogu i ulaznom nizu. Dvodimenzionalna tablica gradi se na temelju vrijednosti relacija *IspodZnaka* i *ReduciranZnakom*.

Ako za znak *A* na vrhu stoga i pročitani ulazni znak *x* vrijedi relacija *IspodZnaka*(*A* , *x*), onda parser stavlja znakove na vrh stoga. Parser nastoji na stog staviti sve znakove desne strane produkcije koja ima podniz *AB*, a za znak *B* vrijedi *x* Є *ZAPOČINJE*(*B*).

Ako za znak *A* na vrhu stoga i pročitani ulazni znak *x* vrijedi relacija *ReduciranZnakom*(*A* , *x*), onda parser znakove na vrhu stoga uspoređuje sa znakovima desnih strana produkcija u cilju primjene akcije *Reduciraj*. Na vrhu stoga je desna strana produkcije <*L*> 🡺 α*A*, a za znak <*L*> vrijedi *x* Є *SLIJEDI*(<*L*>).

Ako za znakove *A* i *x* vrijede obje relacije *IspodZnaka*(*A* , *x*) i *ReduciranZnakom*(*A* *, x*), onda gramatika ima proturječje *Pomakni/Reduciraj*. Parser se gradi na sljedeći način :

1) Znakovi stoga su svi znakovi gramatike i oznaka dna stoga . Reci tablice označavaju se znakovima stoga.

2) Ulazni znakovi su završni znakovi gramatike i oznaka kraja niza . Stupci tablice označavaju se ulaznim znakovima.

3) Tablica *Pomakni/Pronađi* gradi se na sljedeći način :

a) Ako vrijedi relacija *ReduciranZnakom*(*A* , *x*), onda se u redak tablice *A* i u stupac *x* zapiše akcija *Pronađi*.

b) Ako vrijedi relacija *IspodZnaka*(*A* , *x*), onda se u redak *A* i u stupac *x* zapiše akcija *Pomakni*.

c) U sve ostale elemente tablice zapiše se akcija *Odbaci*, *Pomakni* ili *Reduciraj*. Ne dozvoljava se stavljanje oznake kraja niza na stog.

**Primjer 1.2**

Na temelju gramatike i vrijednosti relacija izračunatih u primjeru 1.1 gradi se tablica *Pomakni/Pronađi* na sljedeći način. Budući da za početni nezavršni znak <*B*> vrijedi relacija *IspodZnaka*(*B* , *c*), u redak tablice <*B*> i stupac *c* zapisuje se akcija *Pomakni*. Na temelju relacije *ReduciranZnakom*(<*S*> , ), u redak <*S*> i stupac zapisuje se akcija *Pronađi*.

Na taj način se gradi tablica za sve ostale vrijednosti relacija *IspodZnaka* i *ReduciranZnakom*.

Tablica1 a b c <S> Odbaci Odbaci Odbaci Pronađi <A> Odbaci Odbaci Odbaci Pronađi <B> Odbaci Pomakni Pomakni Odbaci a Odbaci Pomakni Odbaci Odbaci b Odbaci Odbaci Pronađi Pronađi c Pomakni Pomakni Odbaci Odbaci Pomakni Pomakni Odbaci Odbaci

**2. Ostvarenje**

**2.1 *Osnovno pojmovi i moguće pogreške***

Parser Pomakni/pronađi programski je ostvaren pomoću programskog jezika Java.

Program je ostvaren tako da se tablica parsera ucitava iz datoteke, a ulazni niz i produkcije gramatike se zadaju direktno u programu.

U tablici i u ulaznom nizu koriste se samo slova engleske abecede. Oznaka kraja ulaznog niza je $, a oznaka dna stoga je #. Pravilan izgled tablice vidljiv je u **primjeru 1.2** , a jedine promjene su

promjena znakova kraja ulaznog niza i dna stoga. Prilikom granje gramatike i tablice prasera nezavršni znakovi ne smiju se označavati s O ili P zato što su to rezervirani znakovi.

Ulazni niz se zadaje u varijabli ulazni\_niz na sljedeći način:

String ulazni\_niz = "bdbeca$";

Završni znakovi i oznaka kraja niza $ te se sve stavlja pod dvostruke navodnike.

Produkcije gramatike se zadaju u string polju varijabli String[] prod\_gram = **new** String[15]; na sljedeći način:

prod\_gram[0] = "S -> bASB";

prod\_gram[1] = "S -> bA";

prod\_gram[2] = "A -> dSca";

prod\_gram[3] = "A -> e";

prod\_gram[4] = "B -> cAa";

prod\_gram[5] = "B -> c";

Nezavršni znak, razmak, minus, znak ‘veće od’, razmak, niz završnih i nezvaršnih znakova.

Pogrešno zadavanje ulaznog niza produkcija gramatike :

Ulazni niz:

String ulazni\_niz = "bdbeca$;

String ulazni\_niz = bdbeca$;

String ulazni\_niz = "bdb eca$";

String ulazni\_niz = "bdbeca";

Produkcije gramatike:

prod\_gram[0] = " -> bASB";

prod\_gram[0] = "S-> bASB";

prod\_gram[0] = "S -> b ASB";

prod\_gram[0] = "S > bASB";

U slučaju pogrešnog zadavanja ulaznog niza i/ili produkcija gramatike parser neće biti u mogućnosti obaviti parsiranje.

**2.2 *Programsko ostvarenje***

**2.2.1 *Učitavanje tablice parsera i inicijalizacija potrebnih varijabli***

Tablica se učitava pomoću metode *read* koja vraća tablicu u varijabli tipa String.

**public** **static** String

read(String filename) **throws** IOException{

BufferedReader in = **new** BufferedReader

(**new** FileReader(filename));

String s;

StringBuilder sb = **new** StringBuilder();

**while**((s = in.readLine()) != **null**)

sb.append(s + "\n");

in.close();

**return** sb.toString();

}

Tekstualna datoteka koju prima metoda *read* kao argument *String filename* šalje se iz poziva metode *read* iz glavne metode *main().*

Poziv metode *read* iz funkcije *main()*: tablica = *read*("Tablica2.txt");

Tablice(Tablica1 i Tablica2) parsera za pojedine gramatike nalaze se u početnom direktoriju trenutnog java projekta.

U ovoj metodi koristi se objekt BufferedReader in koji u sebi sadrži metodu *readLine()*, koja čita liniju po liniju tekstualne datoteke. Korsitimo i objekt StringBuilder sb koju pomoću metode *append()* sprema linije tekstualne datoteke u varijablu (sb) tipa StringBuilder. Na kraju metode vraćamo varijablu sb koju pomoću metode *toString()* pretvaramo u String.

Inicijalizacija varijabli se događa u glavnoj metodi *main()*.

**public** **static** **void** main(String[] args) **throws** IOException{

String tablica = **new** String();

String[] prod\_gram = **new** String[15];

String[][] matrica\_akcija = **new** String[15][15];

**int** i=0,j=0,m=0,n=0,gran\_matr1=0,provjera = 0,pom1,pom2,pom3,pom4,

pom5;

**int** vrh\_stoga=0,brojac=0;

**int** duljina\_ulaznog\_niza,duljina\_produkcije;

**char**[] z\_i\_n\_znakovi = **new** **char**[15];

tablica = *read*("Tablica2.txt");

System.*out*.println(tablica);

**char**[] k = tablica.toCharArray();

Na početku metode inicijaliziraju se varijable potrebne za tijek programa. String tablica sluzi za spremanje tekstualne datoteke u obliku string pomoću metode *read* u nju. Polje stringova prod\_gram služi kao spremnik za produkcije gramatike čiju tablicu učitavamo. Polje stringova matrica\_akcija služi kao spremnik u kojeg učitavamo akcije iz tablice parsera. Ispod matrice\_akcija je nekolicina pomoćnih varijabli koje služe za ispravno odvijanje programa.

Cjelobrojna varijabla vrh\_stoga služi za praćenje na kojem mjestu od dna je trenutno vrh stoga. Cjelobrojne varijable duljina\_ulaznog\_niza i duljina\_produkcije služe za spremanje duljine ulaznog niza i duljine proukcije koju trenutno gledamo.

Polje znakova z\_i\_n\_znakovi služi za spremanje svih završnih,nezavršnih te znakova dna stoga i kraja ulaznog niza koje čitamo iz tablice parsera. Naredba System.*out*.println(tablica); služi za ispis učitane tablice na zaslon. Pridruživanjem **char**[] k = tablica.toCharArray(); spremamo tablicu u string format u polje znakova zbog lakšeg manipuliranja znakovima.

Inicijalizacija stoga i ulaznog niza :

**char**[] stog = **new** **char**[50];

stog[0] = '#';

String ulazni\_niz = "bebdbecac$";

Stog je polje znakova. U varijablu stog[0] spremili smo oznaku dna stoga #. Ulazni niz je varijabla tipa string.

Spremanje produkcija gramatike u polje stringova :

prod\_gram[0] = "S -> bASB";

prod\_gram[1] = "S -> bA";

prod\_gram[2] = "A -> dSca";

prod\_gram[3] = "A -> e";

Spremanjem produkcija gramatike završila je inicijalizacija glavnog djela potrebnih varijabli te se kreće sa punjenjem polja matrica\_akcija i z\_i\_n\_znakovi koji sun a potrebni za daljnje parsiranje.

**2.2.2 *Punjenje polja matrice\_akcija I z\_i\_n\_znakovi***

Ovaj dio se sastoji od dvije while petlje koje nam omogućuju spremanje podataka u navedena polja.

Prva i puno jednostavnija while petlja:

**while**(k[i]!=' '){

i++;

}

S ovom petljom prelazimo preko prve riječi (Tablica1,Tablica2…) tablice parsera koja je spremljena u polje znakova k.

Druga i puno važnija while petlja preko koje se pune prethodno navedena polja :

**while**(k[i]==' ' || k[i] == '\n' || k[i]=='\t'){

i++;

**if**(k[i]=='O' || k[i]=='P'){

**if**(k[i]== 'O'){

**if**(gran\_matr1 >= n){

matrica\_akcija[m][n]="Odbaci";

n++;

i=i+6;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

**else**{

n=0;

m++;

matrica\_akcija[m][n]="Odbaci";

n++;

i=i+6;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

}

**else**{

**if**(k[i+1]=='o')

**if**(gran\_matr1 >= n){

matrica\_akcija[m][n]="Pomakni";

n++;

i=i+7;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

**else**{

n=0;

m++;

matrica\_akcija[m][n]="Pomakni";

n++;

i=i+7;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

**else**

**if**(gran\_matr1 >= n){

matrica\_akcija[m][n]="Pronadi";

n++;

i=i+7;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

**else**{

n=0;

m++;

matrica\_akcija[m][n]="Pronadi";

n++;

i=i+7;

**if**(provjera == 1 && gran\_matr1 == n-1)

**break**;

}

}

}

**else**

**if**(k[i]!=' ' && k[i] != '\n' && k[i]!='\t'){

**if**(k[i]=='#')provjera = 1;

z\_i\_n\_znakovi[j]= k[i];

j++;

**if**(k[i]=='$')

gran\_matr1 = j-1;

i++;

}

}

While petlja provjerava da li je znak polja (u kojem je spremljena učitana tablica) razmak,novi red ili tabulator. Ako je petlja se nastavlja. Odmah nakon ulaska u petlju povećava se indeks polja te se prvom if petljom provjerava da li je znak u novom indeksu polja O ili P, kako bi se otkrilo da se radi o akcijama parsera Odbaci ili Pomakni/Pronađi.

U prvom redu tablice su samo ime tablice, završni znakovi te znak kraja niza $,a nema rezerviranih znakova O i P. Tako da se tijekom prolaza prvog reda obavlja samo predzadnja if petlja koju vidimo na ovoj stranici. Ona se obavlja zato što prva if petlja **if**(k[i]=='O' || k[i]=='P') nije istinita te se odmah skače na njenu **else if**(k[i]!=' ' && k[i] != '\n' && k[i]!='\t') petlju. U njoj se provjerava da li je pročitan znak dna stoga što znači da se nalazimo u zadnjem redu tablice te se puni polje znakova z\_i\_n\_znakovi. U njoj se još provjerava da li je očitan znak kraja niza te ako je postavlja se granica stupaca matrice matrica\_akcija gran\_matr1.

U sljedećim prolazima while petlje upisuju se znakovi na stogu u polje znakova z\_i\_n\_znakovi, a akcije u matricu\_akcija nakon čega se provjerava da li se radi o zadnjem redu (provjera == 1) i da li se radi o zadnjem stupcu matrice (gran\_matr1 == n-1) što znači da smo prošli kroz cijelu tablicu i upisali sve akcije u matricu\_akcija i sve znakove u polje znakova z\_i\_n\_znakovi.

**2.2.3 *Parsiranje***

Nakon što smo napunili potrebna polja i matrice možemo početi s parsiranjem. Ovaj dio koda nam to i omogućuje.

Pomoću prve dvije naredbe prilagođavamo ulazni niz:

duljina\_ulaznog\_niza = ulazni\_niz.length();

**char**[] ulaz\_niz = ulazni\_niz.toCharArray();

Određujemo duljinu ulaznog niza te ga pretvaramo u polje znakova zbog lakšeg korištenja.

Zatim pomoću tri for petlje ostvarujemo kretanje kroz ulazni niz te traženje potrebnih parametara za parsiranje( znak na vrhu stoga i ulazni znak ).

**for**(pom1 = 0;pom1 < duljina\_ulaznog\_niza;pom1++){

**for**(pom2 = 0;pom2 <= gran\_matr1;pom2++){

**if**(ulaz\_niz[pom1] == z\_i\_n\_znakovi[pom2]){

n=pom2;

}

}

**for**(pom2 = gran\_matr1+1;pom2 < j;pom2++){

**if**(stog[vrh\_stoga] == z\_i\_n\_znakovi[pom2]){

m=pom2-gran\_matr1-1;

}

}

Kao što se vidi prva for petlja u ovom djelu nije zatvorena i ona se prostire do kraja programa, a služi za kretanje kroz ulazni niz. Druga for petlja nam omogućuje da namjestimo indeks stupca matrice\_akcija tražeći ulazni znak u polju z\_i\_n\_znakovi. Treća for petlja nam omogućuje da namjestimo indeks retka matrice\_akcija tražeći trenutni znak na vrhu stoga.

Kada smo namjestili indeks retka i stupca matrice\_akcija provjeravamo o kojoj se akciji radi pomoću niza od tri if petlje. Prve dvije nam određuju akcije Odbaci i Pomakni.

**if**(matrica\_akcija[m][n]=="Odbaci"){

System.*out*.println("Nemoguc nastavak parsiranja:");

System.*out*.println("Na stogu je : "+stog[vrh\_stoga]);

System.*out*.println("Znak ulaznog niza je: "+ulaz\_niz[pom1]);

System.*out*.println("Primjenjuje se akcija Odbaci.");

System.*exit*(0);

}

**if**(matrica\_akcija[m][n]=="Pomakni"){

System.*out*.println("Na stogu je : "+stog[vrh\_stoga]);

System.*out*.println("Znak ulaznog niza je: "+ulaz\_niz[pom1]);

System.*out*.println("Primjenjuje se akcija Pomakni te se" +

" znak ulaznog niza "+ulaz\_niz[pom1]+" stavlja na stog.");

vrh\_stoga++;

stog[vrh\_stoga] = ulaz\_niz[pom1];

}

U slučaju da se radi o akciji Odbaci program javlja da je nemoguć nastavak parsiranja, prikazuje znak na vrhu stoga i znak ulaznog niza i obustavlja izvođenje programa.

U slučaju da se radi o akciji Pomakni program javlja koji je znak na stogu,koji je znak ulaznog nize te javlja da se primjenjuje akcija Pomakni koja stavlja znak ulaznog niza na stog. Nakon toga povećavamo varijablu vrh\_stoga za jedan (vrh\_stoga++) te znak ulaznog niz stavljamo na vrh stoga.

Treća if petlja određuje akciju Pronađi :

**if**(matrica\_akcija[m][n]=="Pronadi"){

System.*out*.println("Na stogu je : "+stog[vrh\_stoga]);

System.*out*.println("Znak ulaznog niza je:"+ulaz\_niz[pom1]);

System.*out*.println("Primjenjuje se akcija Pronadi");

**if**((vrh\_stoga == 1) && (stog[0] == '#') &&

(stog[1] == 'S') && (ulaz\_niz[pom1] == '$')){

System.*out*.println("Na stogu su oznaka dna stoga # i" +" pocetni nezavrsni znak S,a " +

"znak ulaznog niza je oznaka kraja $.");

System.*out*.println("Niz se prihvaca.");

}

**else**{

**for**(pom3 = 0;pom3 < 15;pom3++){

duljina\_produkcije=prod\_gram[pom3].length();

**char**[] polje\_prod = prod\_gram[pom3].toCharArray();

pom4 = duljina\_produkcije-1;

pom5 = vrh\_stoga;

StringBuffer buffer = **new** StringBuffer();

brojac = 0;

**if**(polje\_prod[duljina\_produkcije-1] == stog[vrh\_stoga]){

**if**((duljina\_produkcije-5) < vrh\_stoga+1)

**for**(pom2 = 5;pom2 <= pom4;pom4--){

**if**(polje\_prod[pom4]==stog[pom5]){

brojac++;

pom5--;

buffer.append(polje\_prod[pom4]);

}

}

**if**(brojac == duljina\_produkcije-5){

stog[vrh\_stoga-brojac+1] = polje\_prod[0];

pom5 = 2;

**while**(vrh\_stoga-brojac+pom5 <= vrh\_stoga){

stog[vrh\_stoga-brojac+pom5]= ' ';

pom5++;

}

buffer.reverse();

System.*out*.println("Niz "+buffer+" na stogu " +

"zamjenjuje se nezavrsnim znakom "+polje\_prod[0])

vrh\_stoga = vrh\_stoga-brojac+1;

pom1--;

**break**;

}

}

}

}

U slučaju da se radi o akciji Pronađi program ispisuje koji znak se nalazi na stogu,koji u ulaznom nizu te koja akcija se primjenjuje.

Prva if petlja provjerava da li se radi o završnoj akciji Pronađi u kojoj su na stogu početni nezavršni znak i oznaka dna stoga te na ulazu znak kraja ulaznog niza. U slučaju da je uvjet istinit program ispisuje što se sve nalazi na stogu i u ulaznom nizu te ispisuje da se niz prihvaća.

Ukoliko uvjet prve if petlje nije istinit kreće se na njeno grananje ( else ).

Prva for petlja **for**(pom3 = 0;pom3 < 15;pom3++) prolazi kroz produkcije gramatike. U slučaju da ih nema 15 proći kroz onoliko koliko ih ima. Nakon ulaska u prvu for petlju postavljaju se vrijednosti početnih varijabli koje ovise o produkciji koju trenutačno provjeravamo.

Prva if petlja u for petlji **if**(polje\_prod[duljina\_produkcije-1] == stog[vrh\_stoga]) provjerava da li je krajnje desni znak trenutne produkcije jednak znaku vrha stoga nakon čega možemo početi provjeravati da li je desna strana te produkcije jednaka znakovima na stogu.

Jednakost znakova provjeravamo pomoću for petlje:

**for**(pom2 = 5;pom2 <= pom4;pom4--){

**if**(polje\_prod[pom4]==stog[pom5]){

brojac++;

pom5--;

buffer.append(polje\_prod[pom4]);

}

}

Punimo brojac za svaki točan uvjet. Sljedećom if petljom provjeravamo da li je zaista desna strana produkcije jednaka znakovima na stogu.

**if**(brojac == duljina\_produkcije-5){

stog[vrh\_stoga-brojac+1] = polje\_prod[0];

pom5 = 2;

**while**(vrh\_stoga-brojac+pom5 <= vrh\_stoga){

stog[vrh\_stoga-brojac+pom5]= ' ';

pom5++;

}

buffer.reverse();

System.*out*.println("Niz "+buffer+" na stogu " +

"zamjenjuje se nezavrsnim znakom "+polje\_prod[0]);

vrh\_stoga = vrh\_stoga-brojac+1;

pom1--;

**break**;

U slučaju da je desna strana produkcije jednaka znakovima na stogu sa stoga se briše desna strana trenutne produkcije i na stog se stavlja lijeva strana produkcije odnosno lijevi nezvaršni znak.

**3. Zaključak i primjeri rada**

Programski ostvaren parser Pomakni/Pronađi dobar je primjer rada parsera koji parsira nizove od dna prema vrhu. Parser Pomakni/Pronađi je dobar, ali služi samo kao stepenica prema gradnji učinkovitijih parsera. Njegova neučinkovitost je posljedica učestale usporedbe znakova na vrhu stoga sa znakovima desnih strana produkcija. Taj dio koda najviše usporava program te nakon određenog povećanja broja produkcija koje ima gramatika čiji ulazni niz parsiramo parser postaje neučinkovit.

Koristili smo dvije gramatike koje zorno prikazuju ispravnost rada programski ostvarenog parsera:

**3.1 *Gramatike***

**Gramatika 1:**

1 ) <S> 🡺 a <B><A> 2 ) <S> 🡺 <B> b 3 ) <A> 🡺 c <S> 4 ) <B> 🡺 b

Tablica1 a b c $

S Odbaci Odbaci Odbaci Pronadi

A Odbaci Odbaci Odbaci Pronadi

B Odbaci Pomakni Pomakni Odbaci

a Odbaci Pomakni Odbaci Odbaci

b Odbaci Pronadi Pronadi Pronadi

c Pomakni Pomakni Odbaci Odbaci

# Pomakni Pomakni Odbaci Odbaci

Primjer rada parsera za gramatiku 1:

1) Ulazni niz abcabcbb$

Izlaz:

Na stogu je : #

Znak ulaznog niza je: a

Primjenjuje se akcija Pomakni te se znak ulaznog niza a stavlja na stog.

Na stogu je : a

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz b na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: a

Primjenjuje se akcija Pomakni te se znak ulaznog niza a stavlja na stog.

Na stogu je : a

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz b na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: b

Primjenjuje se akcija Pronadi

Niz b na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz Bb na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz cS na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz aBA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz cS na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz aBA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Na stogu su oznaka dna stoga # i pocetni nezavrsni znak S,a znak ulaznog niza je oznaka kraja $.

Niz se prihvaca.

2) Ulazni niz abcbb$

Izlaz:

Na stogu je : #

Znak ulaznog niza je: a

Primjenjuje se akcija Pomakni te se znak ulaznog niza a stavlja na stog.

Na stogu je : a

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz b na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: b

Primjenjuje se akcija Pronadi

Niz b na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz Bb na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz cS na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz aBA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Na stogu su oznaka dna stoga # i pocetni nezavrsni znak S,a znak ulaznog niza je oznaka kraja $.

Niz se prihvaca.

**Gramatika 2:**

1 ) <S> 🡺 b <A> <S> <B> 2 ) <S> 🡺 b< A> 3 ) <A> 🡺 d <S> c a 4 ) <A> 🡺 e 5 ) <B> 🡺 c <A> a 6 ) <B> 🡺 c

Tablica2 a b c d e $

S Odbaci Odbaci Pomakni Odbaci Odbaci Pronadi

A Pomakni Pomakni Pronadi Odbaci Odbaci Pronadi

B Odbaci Odbaci Pronadi Odbaci Odbaci Pronadi

a Pronadi Pronadi Pronadi Odbaci Odbaci Pronadi

b Odbaci Odbaci Odbaci Pomakni Pomakni Odbaci

c Pomakni Odbaci Pronadi Pomakni Pomakni Pronadi

d Odbaci Pomakni Odbaci Odbaci Odbaci Odbaci

e Pronadi Pronadi Pronadi Odbaci Odbaci Pronadi

# Odbaci Pomakni Odbaci Odbaci Odbaci Odbaci

Primjer rada parsera za gramatiku 2:

1) Ulazni niz bebdbecac$

Izlaz:

Na stogu je : #

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: e

Primjenjuje se akcija Pomakni te se znak ulaznog niza e stavlja na stog.

Na stogu je : e

Znak ulaznog niza je: b

Primjenjuje se akcija Pronadi

Niz e na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: d

Primjenjuje se akcija Pomakni te se znak ulaznog niza d stavlja na stog.

Na stogu je : d

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: e

Primjenjuje se akcija Pomakni te se znak ulaznog niza e stavlja na stog.

Na stogu je : e

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz e na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz bA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: a

Primjenjuje se akcija Pomakni te se znak ulaznog niza a stavlja na stog.

Na stogu je : a

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz dSca na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz bA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz c na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz bASB na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Na stogu su oznaka dna stoga # i pocetni nezavrsni znak S,a znak ulaznog niza je oznaka kraja $.

Niz se prihvaca.

2) Ulazni niz bebecea$

Izlaz:

Na stogu je : #

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: e

Primjenjuje se akcija Pomakni te se znak ulaznog niza e stavlja na stog.

Na stogu je : e

Znak ulaznog niza je: b

Primjenjuje se akcija Pronadi

Niz e na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: b

Primjenjuje se akcija Pomakni te se znak ulaznog niza b stavlja na stog.

Na stogu je : b

Znak ulaznog niza je: e

Primjenjuje se akcija Pomakni te se znak ulaznog niza e stavlja na stog.

Na stogu je : e

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz e na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: c

Primjenjuje se akcija Pronadi

Niz bA na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: c

Primjenjuje se akcija Pomakni te se znak ulaznog niza c stavlja na stog.

Na stogu je : c

Znak ulaznog niza je: e

Primjenjuje se akcija Pomakni te se znak ulaznog niza e stavlja na stog.

Na stogu je : e

Znak ulaznog niza je: a

Primjenjuje se akcija Pronadi

Niz e na stogu zamjenjuje se nezavrsnim znakom A

Na stogu je : A

Znak ulaznog niza je: a

Primjenjuje se akcija Pomakni te se znak ulaznog niza a stavlja na stog.

Na stogu je : a

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz cAa na stogu zamjenjuje se nezavrsnim znakom B

Na stogu je : B

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Niz bASB na stogu zamjenjuje se nezavrsnim znakom S

Na stogu je : S

Znak ulaznog niza je: $

Primjenjuje se akcija Pronadi

Na stogu su oznaka dna stoga # i pocetni nezavrsni znak S,a znak ulaznog niza je oznaka kraja $.

Niz se prihvaca.

**Literatura:**

[1] Jezični procesori 2, Siniša Srbljić , Element, 2002

[2] Internet