Dodatak

Priprema izraza (detaljnije objašnjenje)

Prije bilo kakvog pisanja programa izraz se mora prilagoditi za program tehnikom rekurzivnog spusta. Autor ovog projekta se odlučio za polje kao „spremnika“ izraza. Ubrzo se došlo do problema kako da program zna što su varijable, a što su zagrade i operatori. Problem je riješen tako da se koristila ASCII tablica za sve znakove osim operatora jednakosti. Program učitava izraz i ulazi u jednu petlju koja bi čitala znak po znak[1]. Na početku te petlje nalaze se petlje koje provjeravaju nalazi li se znak u određenom rasponu ASCII znakova. Sve dok znakovi nisu navedeni operatori program ih tretira kao dio varijable.[2] Zbog petlje se ti znakovi skupljaju u jednu varijablu. Kad znak bude jednak jednom od operatora ili zagradama (operatori jednakosti su riješene uvjetima[3] i pomoćnom varijablom Pom dok su zagrade i ostali operatori riješeni petljama i traženjima u rasponu ASCII kodova[4]) to je znak programu da prestane skupljati varijablu i varijabla se provjerava funkcijom Provjera_varijable. U toj funkciji[5] provjerava se da li varijabla započinje slovom. Također se početni znak varijable provjerava petljama i rasponima ASCII kodova. Ako započinje slovom, funkcija vraća 1 i program može nastaviti, u suprotnom izlazi iz petlje i ispisuje poruku da varijabla nije ispravno napisana. Ako ih ima više, program ispisuje prvu neispravnu varijablu u izrazu. Ako je varijabla ispravna, varijabla se zamjenjuje znakom „v“ zbog unificiranosti svih varijabli (u aritmetičkom izrazu nije bitno kako se zove varijabla i koliko se puta pojavljuje u izrazu). Nakon toga se ta varijabla dodaje spremniku Izraz, a pomoćna varijabla za skupljanje varijable Varijabla se resetira kao i varijabla Pom. Nakon što se spremila varijabla, gleda se koja zagrada ili operator je učitan i navedenim postupkom se pohranjuje u spremnik Izraz. Na kraju se dodaje znak „$“ u spremnik koji označava kraj i koji će biti ključan za prihvatljivost izraza.

Izraz se potom šalje funkciji GlavniProgram koja dobiva izraz prilagođen za tehniku rekurzivnog spusta. Na primjer upisat ćemo ((a-b+f)!=c*e), a funkcija GlavniProgram će dobiti polje

Izraz= ['(', '(', 'v', '-', 'v', '+', 'v', ')', '!=', 'v', '*', 'v', ')', '$'] ili

((pas_1+pas_2)/macka01233)!=konj230_202 će biti

['(', '(', 'v', '+', 'v', ')', '/', 'v', ')', '!=', 'v', '$']

[1] 98. linija u programu bez koraka
[2] Od 99. – 114. linije u programu bez koraka
[3] Od 126. – 184. linije u programu bez koraka
[4] 115. linija u programu bez koraka
[5] Od 70. – 88. linije u programu bez koraka

Zamjena varijable sa znakom „v“

U programu se koristi prvo zamjena bilo koje ispravne varijable sa znakom „v“. Autor ovog projekta se odlučio za to zbog jednostavnosti koda programa i čitljivosti dijagrama (koji su i ovako već kompleksni). Usprkos tome, tehnika rekurzivnog spusta je objašnjena sa samo 4 potprograma i bit projekta je sačuvana.

No ako bi se htjelo parsirati i samu varijablu, gramatika bi se proširila novim nezavršnim i završnim znakovima. Umjesto čitanja završnog znaka **varijabla** („v“), parser bi otišao u potprogram **V** (kao Varijabla) pridružen nezavršnom znaku V u kojem bi se gledalo počinje li varijabla slovom. To se rješava na način da parser odlazi u novi potprogram **S** (kao Slovo), a kad se vrati, provjerava se nastavak varijable potprogramom **N** (kao Nastavak).

V→SN

U potprogramu **S** se provjerava je li jedan od znakova završni znak koji predstavlja slovo (koristi se operator „ili“ {**|**} umjesto pisanja 26 zasebnih produkcija za malo ili veliko slovo):

S→a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z

S→A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

Kad parser pročita jedno od navedenih završnih znakova, vraća se potprogramu **V** i odlazi u potprogram **N** koji provjerava je li sljedeći završni znak varijable (ako ga ima, ako ne onda se vraća) znamenka (potprogram **Z**) ili slovo, i rekurzivno ponavlja postupak.

N→ZN **N→SN**

Z→0|1|2|3|4|5|6|7|8|9

U tom slučaju gramatika bi bila:

G = ({A, B, C, D, V, S, N},T,P,A)

T = {+,-,*,/,>,<,>=,<=,=,!=,(,),a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z, A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z, 0,1,2,3,4,5,6,7,8,9}

Produkcije:

A→B$

B→C+B
B→C-B
B→C*B
B→C/B

C→D<B
C→D>B
C→D<=B
C→D>=B
C→D=B
C→D!=B
C→D

D→(B)
D→V

V→SN

S→a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
S→A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

N→ZN
N→SN

Z→0|1|2|3|4|5|6|7|8|9

Pregledom gramatike može se uočiti duplo više potprograma (8), dodatnih 61 završnih znakova od originalne gramatike. To je razlog korištenja „prečaca“ i tretiranje bilo koje varijable kao jedno slovo „v“.