GLAVA 2

Programi i izračunljive funkcije

I cijeli ovi besporeci
po poretku nekome sljeduju.
Nad svom ovom grdnom mješavinom
opet umna sila toržestvuje.
Njegoš

3. Sintaksa jezika S

Sada smo spremni da damo jednu precizniju definiciju jezika S.

Simboli

X_1 X_2 X_3 ...

nazivaju se **ulazne varijable**, simboli

Z_1 Z_2 Z_3 ...

nazivaju se **lokalne varijable**, a Y se naziva izlazna varijabla.

Simboli

A_1 B_1 C_1 D_1 E_1 A_2 B_2 C_{12} D_2 E_2 ...

se nazivaju **labelama**.

Kompletan skup instrukcija jezika S je:

$V \leftarrow V + 1$
$V \leftarrow V - 1$
$V \leftarrow V$
IF $V \neq 0$ GOTO L

gdje je V bilo koja varijabla, a L bilo koja labela jezika S.

Za razliku od ranije definicije jezika S, sada smo dodali još instrukciju V←V. Ova instrukcija ne mijenja vrijednost varijabli pa je potpuno bez efekta. Ona liči na instrukciju CONTINUE iz programskog jezika FORTRAN, a razlog za dodavanje ove instrukcije biće jasan nešto kasnije.

Program je lista (konačan niz) instrukcija. Dužina ove liste naziva se dužinom programa.

Programom ćemo smatrati i praznu listu, odnosno listu dužine 0 koja, naravno, ne sadrži ni jednu instrukciju. Takav program nazivamo **praznim programom.**

Pod **izvršavanjem** (egzekucijom) programa podrazumijevaćemo postupak pri kome se, na bilo koji način, obavljaju operacije zadate u instrukcijama programa. Pri tome se instrukcije izvršavaju redom jedna za drugom, osim u slučaju IF V $\neq$ 0 GOTO L instrukcije. Poslije ove instrukcije izvršava se, ili sljedeća instrukcija (ako uslov nije ispunjen), ili instrukcija sa labelom L (ako je uslov ispunjen). Ako instrukcija sa labelom L ne postoji, program završava rad.

Stanje programa je niz jednakosti oblika V = m, gdje je V variabla a m broj, gdje za svaku varijablu imamo samo jednu jednakost. Skup vrijednosti varijabli koji označavaju stanje programa označićemo sa σ.

Trenutnim stanjem programa nazivamo par (i,σ), gdje je $1 \leq i \leq n+1$, a σ stanje programa. Ako zamislimo da su sve instrukcije programa označene brojevima od 1 do n (za program dužine n), tada broj i, u paru (i,σ), označava broj sljedeće instrukcije koja će biti izvršena. Kada i ima vrijednost n+1, smatraćemo da instrukcija koja treba da slijedi ne postoji, odnosno takvo trenutno stanje odgovara "stop" instrukciji.

Trenutno stanje $(1,\sigma)$ odgovara početku izvršavanja programa, sa stanjem programa σ u kojem ulazne varijable imaju početne vrijednosti, a sve ostale (lokalne i izlazna) vrijednost 0.

Trenutno stanje $(n+1,\sigma)$ odgovara kraju izvršavanja programa.

Ako se program nalazi u trenutnom stanju (i,σ) sljedeće trenutno stanje (j,τ) se definiše sa:

Slučaj 1. Instrukcija i je tipa V $\leftarrow$ V + 1.
j = i+1, a τ se dobija kada se u σ vrijednost varijable V=m zamijeni sa vrijednošću m+1.

Slučaj 2. Instrukcija i je tipa V $\leftarrow$ V - 1.
j = i+1, a τ se dobija kada se u σ vrijednost varijable V=m zamijeni sa vrijednošću m-1, ako je m $\neq$ 0, a ako je m=0 onda je $\tau = \sigma$.

Slučaj 3. Instrukcija je tipa V $\leftarrow$ V.
j = i +1, a $\tau = \sigma$.

Slučaj 4. Instrukcija i je tipa IF V $\neq$ 0 GOTO L.

Slučaj 4a. V = 0, $\tau = \sigma$, j = i + 1.

Slučaj 4b. V$\neq$ 0, $\tau = \sigma$. Ako postoji instrukcija sa labelom L, j uzima najmanji redni broj instrukcije sa tom labelom, a ako instrukcija sa labelom ne postoji j = n +1.

Sada smo u stanju da preciznije definišemo izvršavanje (egzekuciju) programa.

Izvršavanje programa je niz trenutnih stanja $s_1,s_2,...,s_k$, takvih da je s_{i+1} sukcesorsko (sljedeće) trenutno stanje poslije stanja s_i, za i = 1,2,...,k-1. Stanje s_k je krajnje (finalno) stanje.