

Prilog

B

SINTAKSNA I ALGORITAMSKA NOTACIJA

B.1 Sintaksna notacija sa zagradama

U osnovi svake sintaksne notacije nalaze se sledeći elementi:

- sintaksni pojam: leksička konstrukcija koja se definiše;
- metasimbol: element sintaksne notacije koji služi za konstrukciju definicije sintaksnog pojma;
- terminalni simbol: znak ili ključna reč kao najniži i finalni element u sastavu definicije sintaksnog simbola.

U ovom odeljku razmotrićemo kao najprikladniju sintaksnu notaciju sa zagradama. Kao prvo, naglasimo da se u toj notaciji terminalni simboli pišu podvučeno a sintaksni pojmovi obično. Na primer, BEGIN i ; su terminalni simboli dok **Naziv** to nije. Drugo, definicija bilo kakvog sintaksnog pojma se formira primenom pogodno odabranog skupa metasimbola čije objašnjenje sledi:

- ::=** ima značenje “definiciono jednako”; svaka definicija je forme **Pojam ::= ... ;**
- []** ono što se nalazi između zagrada može se pojaviti jednom ili nijednom;
- { }** ono što se nalazi između zagrada mora se pojaviti jednom;
- |** uzima se ono levo ili ono desno od znaka **|** koji označava izbor;
- ...** ono što se nalazi levo od **...** uzima se 0 ili više puta.

Između navedenih metasimbola ne postoje nikakvi prioriteti. Željeni “redosled” primene ovih simbola postiže se grupisanjem delova definicije između para vitičastih ili uglastih zagrada. Sledi nekoliko ilustrativnih primera:

Definicija označenog celog broja

OznaceniCeoBroj ::= [+ | -] Cifra Cifra ...
Cifra ::= 0|1|2|3|4|5|6|7|8|9

Definicija naziva varijable - simbola

Simbol ::= Slovo { Slovo | Cifra | _ } ...

Definicija C sekvence naredbi

SekvencaNaredbi ::= { Naredba ; { Naredba ; } ... }

Definicija C prototipa funkcije

PrototipFunkcije ::=
Tip Funkcija (Tip Simbol [, Tip Simbol] ...) ;

Situacija da se neki sintaksni pojam pojavljuje jednom ili više puta odvojeno zarezima je toliko česta da je za to uveden posebni metasimbol:

,... sve što se nalazi levo od **,...** uzima se jednom ili više puta odvojeno zarezima kao terminalnim simbolima.

Sa takvom notacijom, definicija prototipa C funkcije je mnogo jednostavnija:

PrototipFunkcije ::= Tip Funkcija ({ Tip Simbol } ,...) ;

B.2 Algoritamska notacija akcionih dijagrama

Notacija akcionih dijagrama predstavlja jednostavno sredstvo za strukturiranu specifikaciju algoritama. Prednost te notacije u odnosu na grafičke strukturirane dijagrame i blok-dijagrame toka je dvojaka:

- u osnovi se radi o tekstuelnoj notaciji za koju nije potreban poseban editor;
- postupak postepenog razvoja algoritma ostaje vidljiv na finalnoj specifikaciji.

Razradu nekog postuka **Postupak** na podpostupke **Postupak1** i **Postupak2** predstavljamo pomoću znaka dvotačke i uvlačenjem, kako je prikazano:

```

Postupak
    ->      Postupak
           : Postupak1
           : Postupak2

```

Time se ujedno formira i sekvenca postupaka. Podrazumevani redosled u sekvenci je od gore na dole i, ako je u jednom redu navedeno više postupaka, s leva na desno.

Uslovno grananje **IF** u svim varijantama predstavlja se na sledeći način:

```

IF ( LogickiIzraz )      IF ( LogickiIzraz )      IF ( LogickiIzraz1 )
| PostupakIf             | PostupakIf             | PostupakIf1
                        +-
                        | PostupakElse             +- ( LogickiIzraz2 )
                        |                           | PostupakIf2
                        |                           +-
                        |                           | PostupakElse

```

Ponavljjanje **DO** u svim varijantama predstavlja se kako je prikazano:

while petlja “dokle god”

```
DO WHILE ( LogickiIzraz )  
| Postupak
```

until petlja “sve dok ne”

```
DO  
| Postupak  
+-- UNTIL ( LogickiIzraz )
```

for petlja “za svako i”

```
DO FOR ( i = Poc,Zad,Korak )  
| Postupak
```

for petlja “za svaki element u skupu”

```
DO FOR EACH ( Element IN Skup )  
| Postupak
```

for petlja “zauvek” (podrazumeva se da u telu postoji iskok)

```
DO FOREVER  
| Postupak
```

Iskok iz proizvoljno ugnježđenog dela algoritma predstavlja se kao

```
<---EXIT
```

pri čemu se vrh strelice povlači do nivoa do koga se iskače. Sam iskok ima smisla samo ako je uslovljen, odnosno ako se nalazi neposredno unutar nekog **IF**.

Napomenimo i to da se kvalifikacija argumenata podprograma kao ulaznih, izlaznih i ulazno-izlaznih postiže oznakama **>**, **<** i **<>** s leva, kao i da se ulaz podataka sa standardnog ulaza i izlaz podataka na standardni izlaz predstavljaju pomoću procedura **INPUT()** i **OUTPUT()** gde se između zagrada odvojeno zarezima navodi šta se učitava odnosno ispisuje.

Na kraju, neka nam kao ilustrativni primer posluži postupak nalaženja nule funkcije metodom bisekcije. Polazni interval $[A, B]$ se polovi sve dok ne nastupi jedna od sledećih situacija:

- sredina intervala je egzaktna nula;
- sredina intervala je približna nula koja zadovoljava uslov $B-A < \text{Eps}$;
- ni tačna ni približna nula nisu dostignuti za **Maxit** iteracija.

Postupak kao rezultat vraća ishod (kako je završen), dostignute granice intervala i dostignutu nulu funkcije. Specifikacija algoritma je sprovedena postupno. Radi konciznosti nisu navedene deklaracije varijabli niti simbolickih konstanti.

Korak 1.

```
Bisekcija ( < Ishod, <> A, <> B, < X0, > Eps, > Maxit )
```

Korak 2.

```
Bisekcija ( < Ishod, <> A, <> B, < X0, > Eps, > Maxit )
: Inicijalizacija
: Obrada
: Finalizacija
```

Korak 3.

```
Bisekcija ( < Ishod, <> A, <> B, < X0, > Eps, > Maxit )
: Inicijalizacija
: : Postavljanje pocetnog ishoda
: Obrada
: : DO FOR ( i = 1, Maxit, 1 )
: : | Jedna iteracija
: Finalizacija
```

Korak 4.

```
Bisekcija ( < Ishod, <> A, <> B, < X0, > Eps, > Maxit )
: Inicijalizacija
: : Postavljanje pocetnog ishoda
: : : Ishod = NEDOVOLJNO_ITERACIJA
: Obrada
: : DO FOR ( i = 1, Maxit, 1 )
: : | Jedna iteracija
: : | : Odredjivanje sredisne tacke
: : | : Test na egzaktnu nulu
: : | : Test na pribliznu nulu
: : | : Odredjivanje novog intervala
: Finalizacija
```

Korak 5 - konačno.

```

Bisekcija ( < Ishod, <> A, <> B, < X0, > Eps, > Maxit )
: Inicijalizacija
: : Postavljanje pocetnog ishoda
: : : Ishod = NEDOVOLJNO_ITERACIJA
: Obrada
: : DO FOR ( i = 1, Maxit, 1 )
: : | Jedna iteracija
: : | : Odredjivanje sredisne tacke
: : | : : X0 = (B+A)/2.
: : | : Test na egzaktnu nulu
: : | : : IF ( F(X0) == 0. )
: : | : : | Ishod = EGZAKTNA_NULA
: <:--|--:--:--|--EXIT
: : | : Test na pribliznu nulu
: : | : : IF ( B-A < Eps )
: : | : : | Ishod = PRIBLIZNA_NULA
: <:--|--:--:--|--EXIT
: : | : Odredjivanje novog intervala
: : | : : IF ( F(A)*F(X0) < 0 )
: : | : : | B = X0
: : | : : +-
: : | : : | A = X0

```

U ovom primeru, ispostavilo se da je postupak **Finalizacija** prazan pa je zato izpstavljen iz finalne specifikacije.