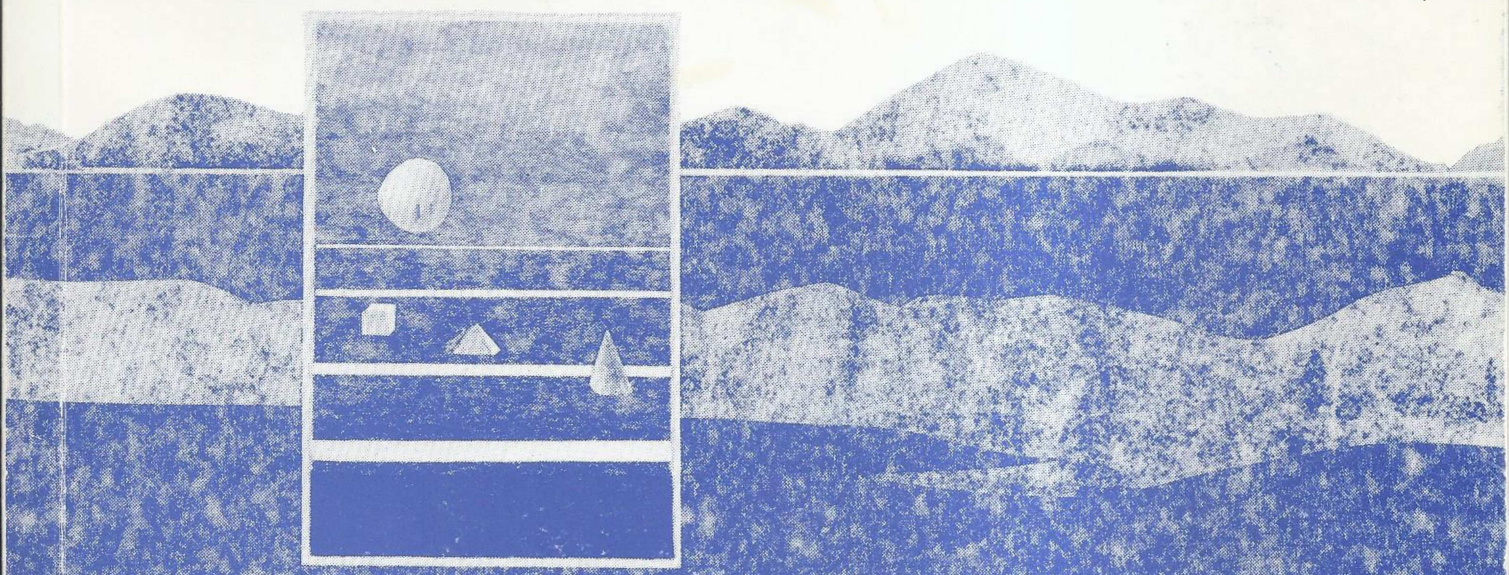
PECOM 32

Kućni računar



UPUTSTVO ZA RUKOVANJE

Ei PECOM 32

Kućni računar

UPUTSTVO ZA RUKOVANJE

S A D R Ž A J :

EI PECOM 32 KUĆNI RAČUNAR

Pred Vama se nalazi kućni ili lični računar, jedan iz familije računara koji se proizvode u Elektronskoj industriji RO „Ei-Računari". Za razliku od ostalih računara ovaj računar je namenjen osobama školskog uzrasta, studentima i širokom krugu korisnika koji već koriste ili imaju nameru da koriste računar za lične potrebe. Spektar namena kučnog računara **PECOM 32** je zaista širok:

- Za obrazovanje – učenje programiranju na različitim programskim jezicima, učenje logičkom razmišljanju, učenje kako se koristi računar i sl.
- Za lične namene – vođenje evidencije o plaćanju računa, stanje tekućeg računa, lični dnevnik i podsetnik.
- Za rešavanje matematičkih problema i upoznavanje sa kolor grafikom.
- Za vođenje kućnih poslova – plaćanje računa (el. energije, gasa, vode, pretplate), kulinarski recepti, telefonski potsetnik, adresar i sl.
- Vođenje poslovanja u malim firmama – planiranje, status izvršavanja poslova, knjigovodstava, komercijalni poslovi, stanje računa kod banke i sl. i
- Za zabavu (TV igre, šah i sl.).

Kućni računar, Ei-**PECOM 32**, je veoma moćan i pouzdan računar. Baziran je na korišćenju 8-bitnog CMOS mikroprocesora CDP 1802. Uz pomoć BASIC-3 programskog jezika primena kućnog računara je jednostavna i za kratko vreme možete ga koristiti za najrazličitije namene. Ovaj kućni računar možete programirati ne samo na BASIC programskom jeziku nego i na mašinskom jeziku mikroprocesora 1802. Na taj način se polje namena **PECOM 32** širi kod projektovanja sistema i uređaja baziranih na mikroprocesorskom dizajnu. Za kreiranje i izradu programa korisnicima stoji na raspolaganju Editor i Asembler programski jezik koji se lako dograđuju u **PECOM 32**.
Da biste koristili kućni računar potrebno je da ga priključite na antenski ulaz TV aparata (u boji ili crno beli) ili monitor jer se na ekranu prikazuju znaci, simboli i grafički simboli. Računar je nakon uključenja odmah spreman da radi na BASIC programskom jeziku, porukom

Ei **PECOM 32**
READY
:

Programe možete da radite sami ili da koristite gotove programe koji se čuvaju na kaseti. To znači da na **PECOM 32** može da se priključi komercijalni kasetofon na čijim kasetama se čuvaju programi korisnika ili gotovi programi. Programi se sa kasete prvo prebace u memoriju **PECOM 32** pa se onda startuju i izvršavaju.

PECOM 32 sadrži 32 KB RAM memorije dostupne korisnicima. Takođe poseduje 16 KB ROM za sistemski softver (12 KB za BASIC-3 i 4KB uslužnog programa) i 4KB statički RAM kao memoriju dispelja sa karakter generatorom. Memorija sistema se može proširiti sa 16KB RAM-a za potrebe korisnika (do ukupno 48 KB RAM) ili sa 16 KB ROM-a za dodatni softver EDITOR i ASEMBLER.

PECOM 32 poseduje sopstveni RF modulator za direktno priključenje na antenski ulaz TV prijemnika preko koga se prenosi video signal i tonski signal tako da preko priključenog TV prijemnika generiše veoma kvalitetan ton.

Preko posebnog priključka se priključuje kasetofon a omogućen je izlaz (serijski komunikacioni interfejs RS232C) za priključenje mikro štampača (sa 40 znakova u redu). Na glavnoj štampanoj ploči je izveden konektor sistemske magistrale koji stoji na raspolaganju korisnicima za priključenje raznovrsnih periferija (mikro flopi diska, modema i sl.).

Za korišćenje kućnog računara kod simulacija i TV igara korisnicima stoji na raspolaganju ručna palica za vođenje kursora.

Tastatura **PECOM-32** je profesionalna. Sadrži set od 55 dirki kojima je obuhvaćen standardni set znakova (uključujući č, ć, ž i š) i simbola. Funkcionalne dirke i četiri strelice ↓, →, ↑, ←, su takođe obuhvaćene setom od 55 dirki.

Na TV ekranu se mogu prikazivati znakovi i simboli u 24 reda sa 40 znakova u liniji. U memoriji se nalazi standardni set od 128 znakova, simbola i grafičkih simbola. Znakovi se mogu softverski modifikovati po veličini i boji. Na ekranu se mogu prikazivati znakovi u 8-boja, s tim što se osnovna boja ekrana može takođe postavljati u 8 boja. Za prikazivanje grafike „niske rezolucije" može se koristiti set od 32 grafička simbola (u okviru karakter generatora). Ukoliko se želi grafika „visoke rezolucije" (240 × 216 tačaka) može se čitav set od 128 znakova programski redifinisati od strane korisnika.

Ton se softverski postavlja naredbom TONE (X, Y, Z) gde je X = 0 – 127 (učestanost tona), Y = 0 – 7 (oktava) i Z = 0 – 15 (jačina zvuka), pa se **PECOM 32** može veoma uspešno koristiti za komponovanje muzike.

Ovim uputstvom su obuhvaćena sva pravila za obavljanje aktivnosti koja treba poštovati za pravilno korišćenje kućnog računara **PECOM 32** od priključenja pa do samostalnog programiranja i izvršavanja programa. U poslednjem poglavlju 6. dati su jednostavni programi koji će pomoći korisnicima u osamostaljivanju kod pisanja sopstvenih programa.

Za programiranje na BASIC programskom jeziku korisnicima stoji na raspolaganju poseban „Priručnik za BASIC programski jezik" a za programiranje na mašinskom jeziku postoji „Priručnik za mikroprocesor 1802"

U prilogu uputstava dati su:

- poruke za greške **PECOM 32**,
- ASCII tabela za znake i simbole,
- uputstvo za rad sa kasetofonom,
- naredbe i funkcije **PECOM 32**,
- osnovni hardver sistema,
- spisak naredbi 1802 za programiranje na mašinskom jeziku.

1. UPOZNAVANJE SA SISTEMOM

1.1. Pakovanje Ei PECOM-a 32

Pakovanje kućnog računara **PECOM 32** sastoji se od:

(1) računara **PECOM-32** sa ugrađenom tastaturom
(2) video kabla sa konektorom za priključenje na antenski izlaz TV aparata
(3) kabla za povezivanje kućnog računara sa kasetofonom
(4) priručnika „UPUTSTVO ZA RUKOVANJE KUĆNIM RAČUNAROM **EI PECOM-32**"
(5) DEMO kasete

Pored gore navedenog biće vam potrebno i sledeće:

(a) standardni TV aparat,
(b) standardni kasetofon.

1.2. Povezivanje računara sa TV aparatom i kasetofonom

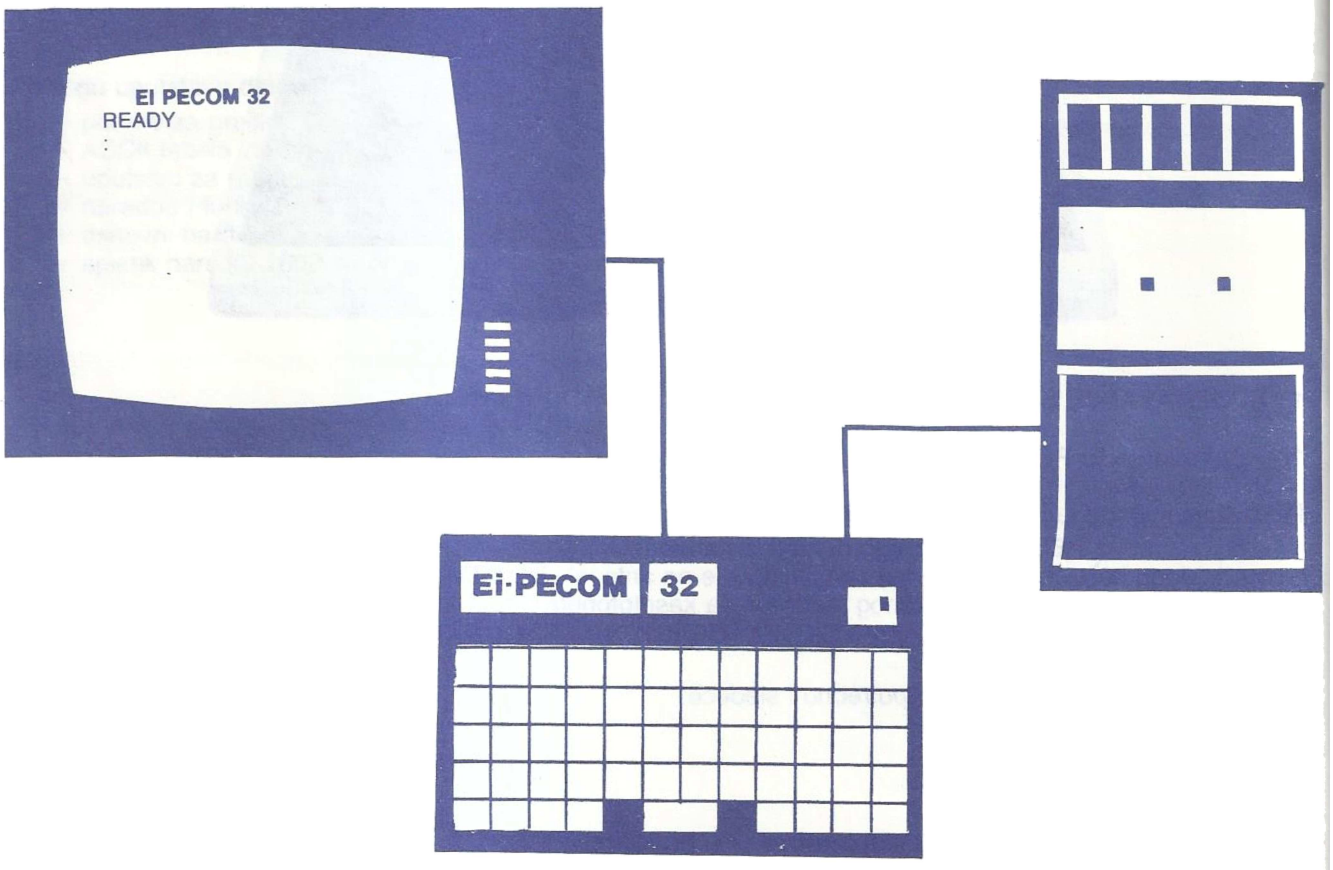
TV aparat se koristi kao jedinica displeja. Kasetofon se koristi za pamćenje programa i podataka. Kada je prekidač na prednjoj strani računara prebačen u krajnji položaj desno (sl. 1.2.1.) računar je isključen, tj. izgubiće se svi privremeno upamćeni programi i podaci. Ako želite da sačuvate ono što ste uradili, to ćete učiniti tako što sadržaj memorije računara prenesete na magnetofonsku traku.

Sl.1.2.1. Kućni računar **PECOM 32** sa priključcima

Da biste TV aparat povezali sa računarom, video kabl povežite za računar (sl. 1.2.2.) i za antenski ulaz TV prijemnika (obično se nalazi na poleđini TV). Nije potreban spoljni RF modulator pošto je isti ugrađen u samom računaru.

Da biste kasetofon povezali za računar, jedan kraj kabla povežite preko priključka koji se nalazi na zadnjoj strani računara (sl. 1.2.2.) a drugi povežite za kasetofon.

Sada priključite računar, TV aparat i kasetofon na mrežu napajanja.



Sl.1.2.2. Minimalna konfiguracija sistema

1.3. **Uključenje**

Pre svega, morate da proverite da li je prekidač na prednjoj strani računara **PECOM-32** u položaju UKLJUČENO.

Odaberite donje VHF područje (kanal 2-3 za PAL sistem) na TV aparatu i podešavajte TV sve dok se ne pojavi sledeća poruka računara **PECOM-32**:

© **EI PECOM 32**
READY
:

Sl. 1.3.1. Poruka o uključenju računara **PECOM 32**

Podešavanje TV aparata na učestanost treba da se vrši polagano. Dok okrećete dugme za podešavanje na TV aparatu, TV slika se gubi a zatim TV daje „snegovitu" indikaciju. Nastavite sa podešavanjem tjunera tako da se „snegovita" slika odjednom izoštri i pojavi poruka sa slike 1.3.1. Sada ste podešeni na „stanicu **PECOM-32**". Podesite kontrast i osvetljaj da biste dobili jasnu sliku.

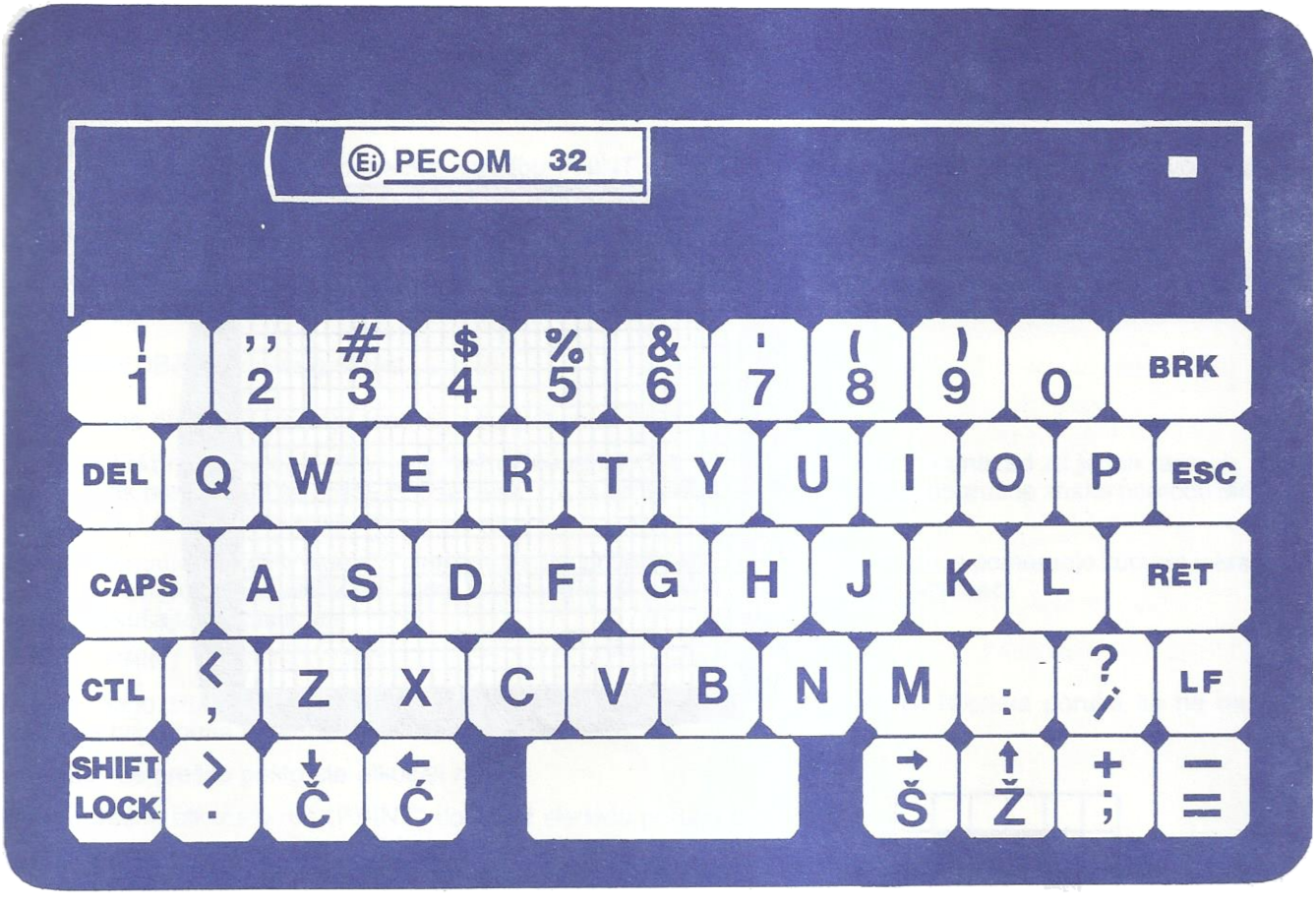
Dve tačke (:) koje se na ekranu pojavljuju posle READY upozoravaju korisnika da računar sada može da prihvati naredbe u BASIC-u i da je računar pod kontrolom BASIC 3 interpretatora.

1.4. **Tastatura računara PECOM 32**

Ei **PECOM 32** je opremljen profesionalnom tastaturom sa 55 dirki kojima je obuhvaćen standardni skup znakova (uključujući Č, Ć, Ž, Š) i funkcionalne dirke: CAPS (CAPS LOCK), ESC (ESCAPE), DEL (DELETE), BRK (BREAK), CTL (CONTROL) i četiri strelice ↓, →, ↑ i ← (vidite sl. 1.4.1.).

Pritiskom na dirku CAPS vrši se izbor korišćenja malih ili velikih slova. Pritiskom na neku dirku sa dva simbola za vreme dok se dirka SHIFT drži pritisnuta, generiše se gornji simbol.

Dirka koja je obeležena sa RET (CARRIAGE RETURN – vraćanje kursora) vraća kursor u krajnji levi položaj na ekranu u sledećem redu. Njome se, takođe šalje poruka računaru koja označava kraj naredbe (vidite odeljak 2.1.).



Sl. 1.4.1. Tastatura računara **PECOM 32**

Dirka BRK (BREAK) omogućava vam da zaustavite program koji se izvršava. Nakon pritiska BRK na ekranu se pojavljuje sledeća poruka:

```
ERR CODE 0
AT LINE 30
READY
:
```

što znači da je računar spreman za nove naredbe u BASIC-u.

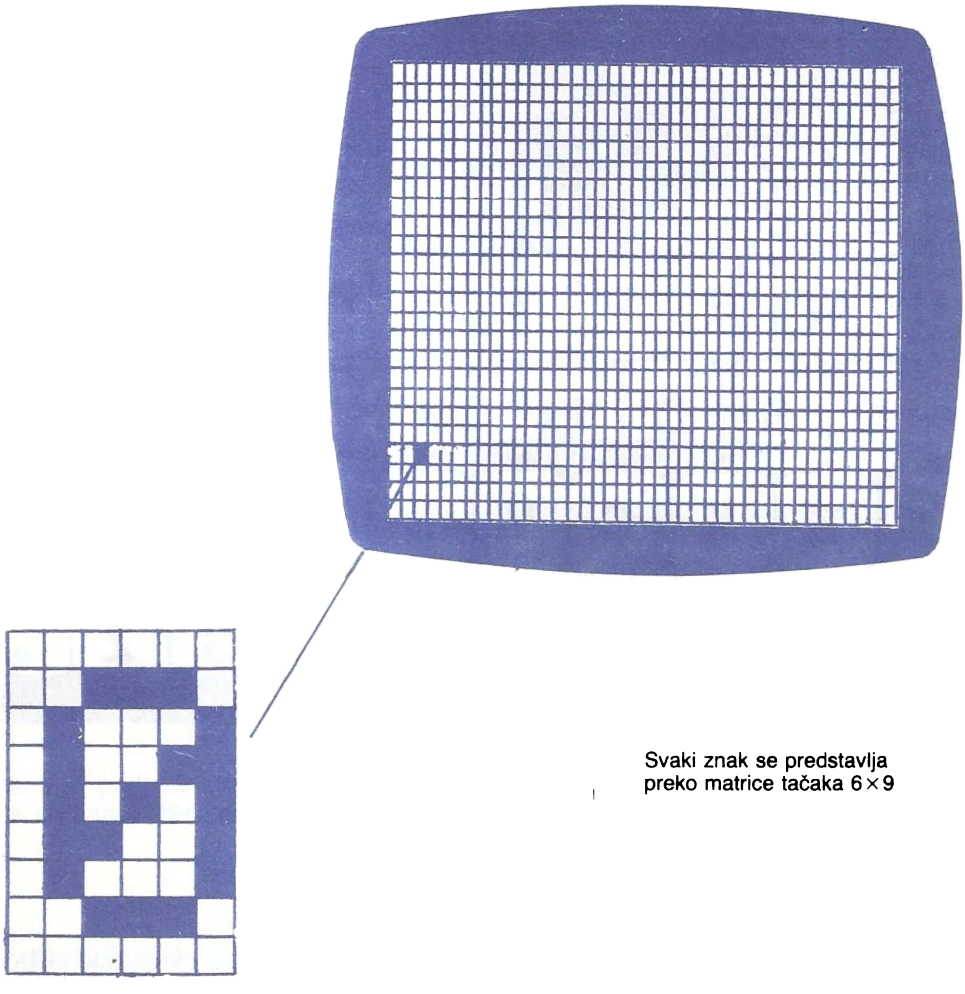
Ako je dirka CTL pritisnuta a naknadno se pritisne neka određena dirka, računar obavlja izvesne akcije.

Npr., pritisak na dirku C dok je dirka CTL pritisnuta, prouzrokovaće da računar ignoriše red koji se kuca (vidi odeljak 2.1.).
Kursor će se pomeriti u krajnji levi položaj u sledećem redu. Ova funkcija je od koristi ako korisnik načini grešku u redu i ako želi da započne novi red. Funkcije dirke CTL detaljnije se razmatraju u odeljku 3.14.

1.5. **Ekran računara PECOM 32**

Ekran računara **PECOM 32** je ustvari ekran priključenog televizora ili minitora. Na ekranu se mogu prikazivati znakovi i simboli u 24 reda sa 40 znakova u liniji (sl. 1.5.1.). Matrica prikazanog znaka na ekranu data je takođe na sl. 1.5.1.

U memoriji računara se nalazi standardni set od 128 znakova, simbola i grafičkih simbola. Znakovi se mogu softverski modifikovati po veličini i boji. Na ekranu se mogu prikazivati znakovi u osam boja s tim što se osnovna boja ekrana može takođe postaviti u osam boja. Ukoliko se želi grafika „visoke rezolucije" (240×216 tačaka) može se čitav set od 128 znakova programski redifinisati od strane korisnika.



Sl. 1.5.1. Ekran sa 40×24 znaka

2 – UPOZNAVANJE SA IZVESNIM NAREDBAMA BASIC-a

2.1. Korišćenje PECOM-a 32 za prikazivanje poruka

Uključite računar da bi ste dobili početnu poruku. Ako naiđete na poteškoće, pogledajte odeljak 1.5. Otkucajte na tastaturi,

PRINT "EI PECOM-32"

nakon čega sledi pritisak na dirku RET. Pritiskom na dirku RET, kursor dovodite na levu stranu ekrana i saopštavate računaru da ste došli do kraja vaše naredbe. Ekran će izgledati ovako:

UNOSITE PRINT "EI **PECOM 32**"
PECOM 32 EI **PECOM 32**
ODGOVARA

ZAKLJUČAK:

Da bi ste prikazali poruku, upotrebite naredbu PRINT posle koje sledi poruka sadržana u paru navodnika. Završite naredbu tako što pritisnete dirku obeleženu sa RET.

Ne morate da otkucate PRINT; koristeći skraćenicu PR postići ćete isti efekat.

Unošenjem PR posle čega sledi pritisak na dirku RET kursor će se pomeriti u sledeći red. Ovo se naziva operacija „novog reda".

PRIMEDBA: Ispravka grešaka

Ako načinite grešku u kucanju pre nego što pritisnete RET:

(1) ... Pritiskom na dirku H za vreme dok držite dirku CTL kursor će se pomeriti unazad za jedan razmak, a znak preko koga se pređe izbrisaće se. Da bi se izvršile ispravke, obrišite pogrešne znake pomoću dirki CTL i H i unesite ispravne.

(2) ... Alternativno, držite dirku CTL pritisnutu dok pritiskate dirku C. Efekat ovoga je pomeranje kursora u krajnji levi položaj u sledećem redu. Ovim red (koji sadrži greške) postaje nevažeći.
Pokušajte da otkucate

PRINT "TEST"

ali, pre nego što pritisnete RET, držite dirku CTL pritisnutu dok pritiskate C. Nikakva poruka se ne daje pošto se nepotpuna naredba jednostavno ignoriše.

Ako načinite grešku pošto ste otkucali red:

Ako pogrešno otkucate reč „PRINT" dobićete sledeću poruku o grešci:

```
ERR CODE 30
READY
:
```

Poruka o grešci sa kodnim brojem 30 znači „neprihvatljiv poslednji znak u naredbi PRINT". Prisustvo: pokazuje da nije načinjena nikakva šteta, i ponovo možete da otkucate naredbu.

Ako ispustite prvi ili poslednji navodnik, računar će prikazati

```
ERR CODE 22
READY
:
```

Poruka o grešci sa kodnim brojem 22 znači „nedostaje navodnik". Ostale poruke o greškama i njihova značenja date su u dodatku. Ako niste dobili nikakav odgovor, možda ste zaboravili da završite naredbu pritiskom na RET.

PRIMERI:

Sada pokušajte sledeće naredbe da biste videli da li dobijate odgovarajuće izlaze:

UNESITE PRINT "MOJA LJUBAV SE NE MOŽE KUPITI NOVCEM – BITLSI"
MOJA LJUBAV SE NE MOŽE KUPITI NOVCEM – BITLSI

UNESITE PRINT "POGREŠNO OBRAZOVANO DETE JE IZGUBLJENO DETE – DŽON KENEDI"
POGREŠNO OBRAZOVANO DETE JE IZGUBLJENO DETE – DŽON KENEDI

Ponovite gornje primere koristeći PR umesto PRINT i videćete da su odzivi računara identični.

2.2. **Upotreba računara PECOM 32 kao kalkulatora**

PECOM 32 može da se upotrebi kao kalkulator.
Pokušajte sledeće na računaru **PECOM 32**

PRINT 5+6

i pritisnite RET. **PECOM 32** će odgovoriti sa:

11

PECOM 32 može da obavlja 5 aritmetičkih operacija:

(a) .. SABIRANJE, pomoću znaka +,
(b) .. ODUZIMANJE, pomoću znaka −. Pokušajte sledeće:

PRINT 34−16

da biste dobili 18. Ne zaboravite da pritisnete RET na kraju vaše naredbe.

(c) .. MNOŽENJE, pomoću znaka ∗
Pokušajte PRINT 7∗8
da biste dobili 56

(d) .. DELJENJE, pomoću znaka /.
Pokušajte PRINT 56/7
da biste dobili 8

(e) .. STEPENOVANJE. Ponekad je potrebno pomnožiti neki broj sa samim sobom određeni broj puta.

Pokušajte PRINT 2∗2∗2∗2∗2
da biste dobili 32

Međutim, umesto da pišete sve ovo, isto možete da predstavite na sledeći način:

Pokušajte PRINT 2↑5
da biste dobili 32

Stoga je znak stepenovanje ↑ jednostavno skraćenica računara za ponovljeno množenje istim brojem; 2 ↑ 5 ima isto značenje kao i 2^5, ali je računar tako projektovan da može da prepozna samo 2 ↑ 5 a ne 2^5. Slično tome, računar može da interpretira 7∗8 i 56/7 ali ne i 7×8 i 56÷7.

Svi gornji primeri sadrže dva broja i jednu operaciju. **PECOM 32** može da rešava i složenije probleme sa više brojeva i mnogo različitih operacija.

Pokušajte	PRINT (5+6) ∗ (9−7)
da biste dobili	22
Pokušajte	PRINT (7−4) ∗ (19−14)/(8−3)
da biste dobili	3
Pokušajte	PRINT (3↑3+2↑4) / (2∗5)
da biste dobili	4,3

Računar će obaviti izvesne operacije pre drugih, npr. prema aritmetičkom pravilu, on će prvo obaviti ∗ i / a onda + i −. Kompletan skup svih pravila dat je u odeljku 5.3. Koristite zagrade „( )" u ovom trenutku koliko vam drago da biste računaru saopštili koja operacija treba prvo da se obavi.

ZAKLJUČAK:

PECOM 32 može da razreši složene aritmetičke probleme koji sadrže +, −, / i ↑. Uvek pritisnite RET na kraju vaše naredbe. Koristite oznake računara ∗ i / umesto oznake × i ÷. Kad god je potrebno, koristite zagrade. Radi detaljnijih informacija, konsultujte odeljak 5.3.

2.3. **Promena boje**

Ukoliko koristite TV u boji ekran računara **PECOM 32** je višebojan. Naredba za bojenje ekrana ima sledeći oblik:

```
SCR X
```

gde je X decimalni broj koji se bira prema tabeli 2.3.1.

Tabela 2.3.1

X	Boja
0	crna
1	zelena
2	plava
3	maslinasta
4	crvena
5	žuta
6	ljubičasta
7	bela

Npr., unesite naredbu

```
SCR 1
```

a zatim pritisnite RET. Primetili ste da je boja ekrana sada zelena.

Ako ovu naredbu stavite u petlju moći ćete da pratite izmenu svih mogućih boja ekrana.

Npr.,

```
10 FOR I=0 TO 7
15 SCR I
20 WAIT (800)
30 NEXT I
```

WAIT naredba u redu 20 obezbeđuje da se boja zadrži određeno vreme na ekranu.

2.4. **Stvaranje zvučnih efekata**

Kod kućnog računara **PECOM 32** zvučni efekt visokog kvaliteta generiše se preko zvučnika televizora.

Naredba TONE (izraz1, izraz2, izraz3) daje kontinualni ton pri čemu:

Izraz1 – određuje učestanost i može da varira od 0 do 127
Izraz2 – vrši izbor jedne od 8 oktava (uzima vrednost od 0 do 7)
Izraz3 – određuje amplitudu tj. jačinu tona i može da varira od 0 do 15.

Korišćenjem naredbe TONE i WAIT u programu mogu se dobiti različite melodije.

Na primer, generisanje tona C u lestvici vrši se na sledeći način:

```
10 TONE  (119,4,8)
20 WAIT  (32)
30 TONE  (0,0,0)
40 END
```

U redu 10 generiše se ton naredbom TONE. Naredbom WAIT određuje se trajanje tona, a naredbom TONE (0,0,0) ukida se generisani ton.

Naredba TONE (izraz1, izraz2) je mala modifikacija prethodne. Izraz1 određuje frekfentni opseg belog šuma i može da varira od 1 do 8.

Izraz2 određuje amplitudu i može da varira od 0 do 15.

Nakon unošenja

TONE (1,1)

i pritiskom na RET čuće se efekat šuma. Sa

TONE (1,2)

osetiće se pojačanje šuma (ili amplitude).
Za ukidanje šuma unosi se:

TONE (0,0).

3 – PISANJE JEDNOSTAVNIH PROGRAMA

3.1. Vaš prvi program u BASIC-u za PECOM 32

Uvešćemo vas u programiranje primenom primera. Pozivamo vas da ukucate primere programa a onda da posmatrate odziv računara. Namena prvog programa je da vam pokaže glavnu strukturu programa i upotrebu izvesnih naredbi u BASIC-u. Ukucajte sledeće:

PROGRAM 3.1.

```
: NEW
: 10        PR    "El PECOM 32"
: 20        PR    2+3
: 30        PR    (5+6) * (9-7)
: 40        END

: RUN


  El PECOM 32
  5
  22
  READY
:
```

Program (i)

Naredba RUN (ii)

Odziv računara (iii)

Indiciranje na ekranu deli se u tri glavna dela:

(i) sam program
(ii) naredbu RUN koja kaže računaru da izvrši naredbe programa, i na kraju
(iii) izlaz računara koji je rezultat izvršenja naredbi programa.

Ispitajmo dalje sam program.

NEW je naredba koja kaže računaru da izbriše sve programe ili podatke koji su prethodno uneti.

Svaki red naredbi u programu počinje brojem reda 10,20, itd. Računar će izvršiti naredbe u sekvenci, tj. redosledu brojeva reda.

Ako pogledate naredbe 10, 20 i 30, možete primetiti da ste ih ranije sreli u odeljku 2. Međutim, u ovom slučaju, naredbe se izvršavaju kao grupa (pošto unesete RUN), umesto da se izvršava jedna po jedna svaki put (pošto ste upotrebili RET). Ovo je osnovna razlika između upotrebe računara **PECOM 32** kao računara za izvođenje programa i upotrebe računara **PECOM 32** kao kalkulatora.

Naredba END ukazuje na kraj programa. Kada računar naiđe na END on završava izvođenje programa.

RUN je naredba koja računaru kaže da počne sa izvršavanjem programa.
Računar traži red sa najmanjim brojem i počinje izvršavanje svakog reda numeričkim redosledom.

PRIMEDBA: Više o brojevima redova

PECOM 32 vam preporučuje da započnete svoj program sa redom broj 10 i dalje sa 20, 30, 40 .. itd. tj. ostavljajući prostore za još 9 instrukcija. Ovo će vam dozvoliti da ubacite nove redove bez potrebe da izmenite prvobitne brojeve redova. Ako pridodate red 15, ova instrukcija će se izvršavati pre reda 20.

Nakon što unesete naredbu sa nekim brojem reda, vi u stvari saopštavate računaru da je ne izvršava sve dok ne ukucate RUN. Računar koristite u režimu rada sa programom. AKo ne koristite broj reda, kao u odeljku 2, izvšavanje će započeti čim ukucate RET. tj. bićete u režimu rada sa kalkulatorom.

Broj reda služi i kao naziv (ili labela) za naredbu. Brojevi redova koriste se i u naredbama GOTO i u pripremi programa. Vidi odeljke 3.2 i 3.3

ZAKLJUČAK: Šta je program?

Program je grupa (ili niz) naredbi. Računar će normalno izvršavati naredbe u sekvenci, tj. u numeričkom redosledu brojeva redova. Naredba NEW na početku programa izbrisaće sve prethodne programe a naredba END prouzrokovaće da računar završi izračunavanje. Naredba RUN saopštava računaru da počne sa izvršavanjem počev od naredbi sa najnižim brojem reda. Neke naredbe se često nazivaju komande. Tako govorimo o komandi END, komandi RUN itd.

3.2 Korišćenje naredbe GOTO

Kao što je objašnjeno u odeljku 3.1, računar će normalno izvršavati naredbe u određenom redosledu. Međutim, primenom naredbe GOTO računar napušta standardni redosled i prelazi na izvršenje naredbi čiji je broj reda dat u naredbi GOTO. Pokušajte da ubacite red „15 GOTO 30" u program u odeljku 3.1; treba da dobijete:

PROGRAM 3.2.

```
NEW
10      PR      "EI PECOM 32"
15      GOTO    30
20      PR      2+3
30      PR      (5+6) * (9-7)
40      END
```

Dodatna komanda GOTO

```
RUN

EI PECOM 32
22
```

Napomena: Naredba u redu 20 nije izvršena.

ZAKLJUČAK: Naredba GOTO

Naredba "GOTO n" saopštava računaru da izvrši naredbu sa sledećim brojem reda n, umesto naredbe sa brojem reda koji sledi. Kaže se da je naredba „bezuslovni skok" pošto je računaru rečeno da skoči ili da se grana na drugu naredbu uvek, tj. bezuslovno. Ako broj reda n ne postoji, generiše se poruka o grešci. „n" može da bude neki izraz.

Prema tome, "GOTO A–B" i "GOTO 50 * (A+B)" su važeće naredbe.

Da biste obrisali broj reda, ukucajte broj reda a zatim RET. Red 15 obrisaćemo pomoću sledećeg:

```
15
LIST
```

Broj reda posle koga sledi RET

Računar će odgovoriti sa:

```
10    PR    "EI PECOM 32"
20    PR    2+3
30    PR    (5+6) * (9-7)
40    END
```

Da biste ubacili red između reda 20 i reda 30, odaberite broj reda, recimo 25, ukucajte 25 GOTO 40, a zatim naredbu LIST.

```
10    PR    "EI PECOM 32"
20    PR    2+3
25    GOTO  40                  Ubačen je novi red
30    PR    (5+6) * (9-7)
40    END
```

3.3 Listanje i izmena programa

Sledeći postupak podrazumeva da ste PROGRAM 3.2 iz odeljka 3.2. uneli u računar. Ako niste, uradite to sada pre nego što nastavite.

Da biste računaru rekli da izlista program, pokušajte

```
LIST
```

posle čega sledi RET i računar će odgovoriti tako što će da izlista poslednji uneti program, u ovom slučaju PROGRAM3.2, koji ovako izgleda:

```
10    PRINT   "EI PECOM 32"
15    GOTO    30
20    PR      2+3
30    PR      (5+6) * (9-7)
40    END
```

Da biste zamenili red, ukucajte broj reda koji treba da se zameni posle koga dolazi nova naredba. Sledeće zamenjuje red 15. Pokušajte

```
15      GOTO    40
LIST
```

Računar će odgovoriti na sledeći način:

```
10      PRINT   "EI PECOM 32"
15      GOTO    40              Ovo zamenjuje stari red
20      PRINT   2+3
30      PRINT   (5+6) * (9-7)
40      END
```

PRIMEDBA: Više o LIST komandi

```
LIST
LIST n
LIST n, m
```

Komanda LIST (posle koje ništa ne dolazi) učiniće da računar indicira ili izlista ceo program. „LIST n" prikazaće samo red obeležen sa n.
„LIST n, m" započeće listanje u redu n a završiće zaključno sa redom m.
„n" i „m" mogu biti neki aritmetički izraz.

Ako je izraz u bilo kom momentu, jednak broju koji je neki nepostojeći broj reda, tada će se prikazati sledeći broj reda.

ZAKLJUČAK: Izmena programa

Izmena programa podrazumeva njegovo korigovanje ili modifikovanje. Možemo zameniti red, obrisati red ili ubaciti red koristeći brojeve redova. Uvek LISTAJTE program ili onaj deo programa koji se menja da biste bili sigurni da je očekivana izmena izvršena (vidi i odeljak 3.11).

3.4. Napišite program koji će prihvatiti podatke sa tastature (koristeći naredbu INPUT)

Počinjući sa poslednjim programom iz odeljka 3.3, pokušajte

```
 5      INPUT   A,B             Ubacite novu komandu INPUT
10                              Obrišite red 10
20      PRINT   A+B
25                              Obrišite redove 25 i 30
30
40      END

LIST
```

Posle izmene računar će odgovoriti sa

PROGRAM 3.4.

```
 5      INPUT   A,B
20      PRINT   A+B
40      END
```

INPUT je ulazna naredba; A i B su „još ne specificirane" ili „nepoznate" veličine koje se zovu promenljive. Kada računar naiđe na ulaznu naredbu, on se zaustavlja i šalje na ekran „?". Zatim čeka da mu korisnik odgovori. Korisnik treba da ukuca onoliko brojeva koliko ima promenljivih. Ovi brojevi specificiraju vrednosti „nepoznatih veličina" ili „promenljivih".

Za A = 2, B = 3, pokušajte

```
RUN
?       2, 3    Ulazni podaci korisnika za A i B kao odgovor na ?
5               Izlaz računara jednako A+B
```

Za A = 1012, B = 4517, pokušajte

```
RUN
?       1012,  4517     Drugi skup podataka za A i B
5529                    A+B
```

Sada imate program koji će vam dati zbir dva broja kada se daju vrednosti ova dva broja.

Pokušajte sa drugim vrednostima za A i B.
Sada zamenite red 20 nekim složenijim izrazom. Pokušajte.

```
20      PRINT    (A * A + B * B)↑0,5
LIST
```

Program postaje

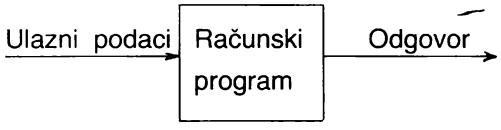
```
 5      INPUT    A, B
20      PRINT    (A * A + B * B)↑0,5
40      END
```

što, kada se izvrši, daje

```
RUN
?       3, 4
5
```

Neki od vas mogu prepoznati da je gornji program aplikacija Pitagorine teoreme na koju ćete svakako naići na časovima geometrije. Ako želite da naučite nešto više o Pitagorinoj teoremi, molimo vas da pročitate sledeću PRIMEDBU. U međuvremenu, važna stvar jeste shvatiti veliku korist ovog programa od tri reda.

Red 5 može da se modifikuje da prihvati podatke za više od dve promenljive. Red 20 može se zameniti složenim algebarskim izrazom sastavljenim od ovih promenljivih. Kada se program izvrši, a vrednosti se dodele promenljivim veličinama kao odgovor na „?", računar će dati rezultat izraza. Isto se možete ponoviti sa drugim skupom vrednosti za ulazne podatke. Ovo je suština računarskog programa i može se ilustrovati na sledeći način:

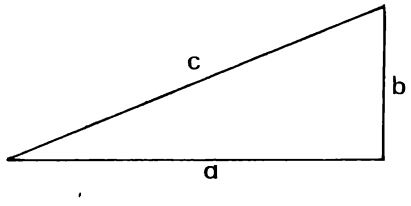


Kada se računar koristi bez brojeva redova (kao u odeljku 2 kada računar radi kao kalkulator), treba da obezbedite postupke izračunavanja za svaki skup podataka.

PRIMEDBA: Pitagorina teorema

Ako je dužina hipotenuze pravouglog trougla c, a dužina druga dva kraka je a i b, kao što se vidi na slici, prema Pitagorinoj teoremi je

$$c=(a^2+b^2)^{1/2}$$



tj. za dato a i b, c može da se izračuna iz gornjeg obrasca. Da biste to uradili koristeći BASIC, algebarski izraz na levoj strani treba da se preuredi u izraz u BASIC-u, tj. (A * A + B * B)↑0,5. U sledećem odeljku dati su primeri kako preurediti algebarski izraz u izraz BASIC-a.

3.5. **Algebarski izrazi i izrazi u BASIC-u**

Svakako da poznajete izvesne algebarske obrasce, npr. obrasce za izračunavanje obima i površine kruga. Međutim, kako računar može da prepozna samo izraze u BASIC-u, ovi algebarski izrazi mora da se preurede u ekvivalentne izraze u BASIC-u pre nego što se unesu u računar. Na primer, „ab", „a÷b", „a^b" mora da se preurede u „a * b", „a/b" i „a↑b". U sledećoj tabeli dato je nekoliko primera:

Tabela 3.5. Pretvaranje algebarskih izraza u izraz u BASIC-u

	Opis	Algebarski izraz	Izraz BASIC-u
1.	Površina kruga, radijus R	πR^2	PI * R↑2
2.	Obim kruga, radijus R	$2\pi R$	2 * PI * R
3.	Površina trougla, data visina H i osnova B	HB/2	H * B/2
4.	Površina trougla, date tri stran. A, B i C i S=1/2 (A+B+C)	$\sqrt{S(S-A)(S-B)(S-C)}$	(S * (S−A) * (S−B)*(S−C))↑0,5
5.	Hipotenuza pravouglog trougla C za katete A i B	$C=(a^2+b^2)^{1/2}$	C=(A * A +B * B)↑0,5 ili C = (A↑2+B↑2)↑0,5
6.	Rešiti X u datoj kvadratnoj jednači $AX^2 + BX + C = 0$	$X=\frac{-B\pm\sqrt{B^2-4AC}}{2A}$	X=(−B+(B * B−4 * A * C)↑ 0.5)/(2 * A) i X=(−B−(B * B−4 * A * C)↑ 0,5)/(2 * A)
7.	„Glavnica plus kamata" po godišnjoj složenoj interesnoj stopi od R%, pri čemu je glavnica P posle n godina	$P(1+R)^n$	P * (1 + R)↑n

Startujući sa modelom programa od tri reda iz odeljka 3.4 i krajnje desnom kolonom iz tabele 3.5, pokušajte da napišete program za neke ili sve gornje probleme.

3.6. **Dovođenje računara PECOM-32 u situaciju da donosi odluke**

Unesite sledeći program za izračunavanje površine trougla, RUN i 3 skupa podataka:

PROGRAM 3.6.

```
NEW
10        INPUT   A,B,C
20        IF  A < 0   THEN  GOTO 50
25        S=(A+B+C)/2
30        PRINT (S * (S−A) * (S−B) * (S−C))↑0,5
40        GOTO 10
50        END
```

```
RUN
?  7, 8, 9
26,8328
?  12, 14, 17
83,0267
?  −1, 2, 3
READY
:
```

Kada računar naiđe na naredbu „GOTO 10" u redu 40, on se uvek grana natrag na naredbu 10 i zahteva drugi skup ulaznih podataka. Ovo se može nastaviti beskonačno (pod uslovom da su vaši podaci korektni, npr. da je neka od dve strane trougla veća od treće strane) i sa ovim malim programom možete izračunati površinu svih trouglova na svetu. Ovo je svrha reda 20. Ova nova naredba zahteva od računara da ispita vrednost za A, i ako je A manje od nule, onda se grana na 50 da bi se završilo izvođenje. Naredba GOTO izvršava se samo ako je neki uslov istinit (u ovom slučaju A<0). Naredba „IF ... THEN GOTO" naziva se uslovno grananje, tj. grananje ako je izvesni uslov istinit.

Takođe, možemo reći da računar **PECOM-32** donosi odluku šta sledeće treba da uradi zavisno od izvesnih uslova.

U mnogim verzijama BASIC-a, naredba „IF ... THEN GOTO" predstavlja jedini oblik „IF" naredbe. Međutim u BASIC-u za **PECOM 32** iza „IF ...THEN" može da sledi grupa naredbi (odvojenih dvema tačkama). Ovo je veoma važna prednost pošto nam fleksibilnost pomaže da načinimo program koji je „strukturisaniji".

Red 25 je naredba „dodeljivanja". Vrednost koja proizilazi iz izračunavanja izraza na desnoj strani data je ili je dodeljena promenljivoj na levoj strani.

Koristeći drugi oblik IF komande, gornji program može se preurediti na sledeći način ali sa identičnim rezultatima.

```
NEW
10      INPUT  A,B,C
20      IF  A>0  THEN  S=(A+B+C)/2:
        PRINT  (S * (S-A) * (S-B) * (S-C))↑0,5: GOTO 10
30      END
```

Red 20, koji se sastoji od mnogo naredbi odvojenih dvema tačkama, izvršiće se samo ako je A>0, tj. ako je A veće od 0. Ako je A=0, red 20 biće preskočen a red 30 završava izvršenje.

PRIMEDBA: IF komanda

Simboli

> veće od
< manje od

nazivaju se realcioni operateri. Ostali relacioni operateri su:

= jednako
<> nejednako
>= veće od ili jednako
<= manje od ili jednako

3.7. Formiranje dobrih programerskih navika

Poboljšajmo program 3.6. Unesite sledeće redove:

PROGRAM 3.7.

```
NEW
  1     REM OVIM PROGRAMOM SE IZRAČUNAVA POVRŠINA TROUGLA
  5     PRINT "OVAJ PROGRAM IZRAČUNAVA POVRŠINU TROUGLA"
 10     INPUT "MOLIMO UNESITE DUŽINE STRANICA A,B,C" A,B,C.
 20     IF A < 0 THEN GOTO 120
 30     S = (A + B + C) /2
 40     IF S < A THEN GOTO 100
 50     IF S < B THEN GOTO 100
 60     IF S < C THEN GOTO 100
 70     T = (S * (S-A) * (S-B) * (S-C))↑ 0,5
 80     PRINT "POVRŠINA TROUGLA = ", T
 90     GOTO 10
100     PRINT "NETAČNI PODACI: DVE STRANICE TROUGLA NISU DUŽE OD
        TREĆE"
115     GOTO 10
120     PRINT "KRAJ POSTUPKA": END
```

```
OVAJ PROGRAM IZRAČUNA POVRŠINU TROUGLA
MOLIMO UNESITE DUŽINE STRANICA
A, B, C? 12, 13, 14
POVRŠINA TROUGLA = 72,3079
MOLIMO UNESITE DUŽINE STRANICA
A, B, C? 12, 13, 26
NETAČNI PODACI: DVE STRANICE TROUGLA NISU DUŽE OD TREĆE
MOLIMO UNESITE DUŽINE STRANICA
A, B, C? -1, 2, 3
KRAJ POSTUPKA
```

Razmotrimo šta ste uradili?

Red 1 koji počinje ključnom reči REM (skraćeno od REMARK) dopušta vam da posle nje ubacite bilo kakvu primedbu da biste objasnili šta program radi ili kako on radi. Naredba REM može se proizvoljno upotrebiti bilo gde u programu. Ona se lista (pomoću LIST) ali je računar ignoriše za vreme izvođenja. Ona je namenjena više za čitaoca programa nego za računar.

Red 10 je sličan naredbi INPUT koja je prethodno upotrebljena osim što vam par navodnika dozvoljava da ubacite poruku koja će biti prikazana pre nego što se znak pitanja „?" pojavi da unesete podatke.

Slično tome, red 80 je sličan naredbi PRINT koja je prethodno upotrebljena, osim što poruku pod navodnicima prati promenljiva T odvojena dvema tačkama. Ovo vam dopušta da ubacite poruku da biste objasnili šta je T.

Redovi 40, 50 i 60 čine da se poruka o grešci u redu 100 prikaže ako su podaci netačni. Takav je slučaj drugog skupa ulaznih podataka gde je $12+13 < 26$.

Uporedite program 3.7 sa programom 3.6; vidi se da prvi ima mnogo više reči, poruka i objašnjenja (koji se nazivaju „dokumentacija"). Svrha ovih je da program i izlaz načine značajnijim i čitljivijim. Za kratak program, i kada vam je misao o programu i dalje u svežem sećanju, takva „dokumentacija" ne mora da izgleda neophodna. Za neki dugačak program, i za nekoga ko program uzima po prvi put, ili čak i za sebe posle nekoliko nedelja, „dokumentacija" bi bila od velikog značaja. Kako ste vi mlad programer, formiranje dobrih navika, tako što u potpunosti dokumentujete svoj program porukama „REM" je veoma poželjno.

3.8. **Proste vežbe programiranja sa rešenjima**

Ne zaboravite naš moto „Naučite programiranje vežbanjem". Uvereni smo da ćete, ako postupite u skladu sa ovim priručnikom i ako pokušate sve korake na računaru **PECOM-32**, veoma ceniti programiranje i da ćete biti u boljem položaju da razumete stručnije udžbenike. „Potpuno razumevanje proizilazi iz iskustva". Sledeće vežbe namenjene su da vam pruže mogućnost da steknete veće iskustvo.

Složeni interes

Problem

Za date P=glavnica prvobitnog iznosa ($)
R=kamata za godinu dana (%)
Y=broj godina
T=koliko se puta u godini kamata izračunava npr. ako se izračunava svakodnevno, T=365.

Od vas se zahteva da napišete program da biste dobili
F=glavnica plus kamata, ili konačni iznos ($).

Predlog

Možete poželeti da napišete svoj program pre nego što pogledate rešenje. Tako se najbolje uči.

Rešenje

```
 10    REM    OVAJ PROGRAM IZRAČUNAVA GLAVNICU PLUS
 20    REM    KAMATU F, ZA DATE
 30    REM    P=GLAVNICU ($)
 40    REM    R=KAMATU ZA GODINU DANA (%)
 50    REM    Y=BROJ GODINA
 60    REM    T=KOLIKO SE PUTA U GODINI KAMATA IZRAČUNAVA
 70    REM    UNESITE P=-999 ZA ZAUSTAVLJANJE.
 75    PRINT "OVAJ PROGRAM IZRAČUNAVA GLAVNICU PLUS SLOŽENI INTERES"
 77    PRINT
 78    PRINT
 80    INPUT"GLAVNICA"P
 85    IF P=-999 THEN GOTO 210
 90    INPUT "KAMATA ZA GODINU DANA" R
100    INPUT "BROJ GODINA" Y
110    INPUT "KOLIKO SE PUTA U GODINI KAMATA IZRAČUNAVA" T
115    REM    REDOVI 120 I 150 DETEKTUJU GREŠKE U ULAZNIM PODACIMA
120    IF P<0 THEN GOTO 190
```

```
130     IF R<0 THEN GOTO 190
140     IF Y<0 THEN GOTO 190
150     IF T<1 TEHN GOTO 190
160     F = P * (1+R / (100 * T))↑ (Y * T)
165     PRINT
170     PRINT "GLAVNICA PLUS KAMATA = $ "; F
180     GOTO 77
190     PRINT
195     PRINT "NETAČNI PODACI; PONOVO POKUŠAJ"
200     GOTO 77
210     PRINT
215     PRINT "KRAJ POSTUPKA": END

          RUN

OVAJ PROGRAM IZRAČUNAVA GLAVNICU PLUS SLOŽENI INTERES
GLAVNICA? 1000
KAMATA ZA GODINU? 15
BROJ GODINA? 5
KOLIKO SE PUTA U GODINI KAMATA IZRAČUNAVA? 365
GLAVNICA PLUS KAMATA = $ 2115,99
GLAVNICA? 15000
KAMATA ZA GODINU DANA? 10
BROJ GODINA? 3
KOLIKO SE PUTA U GODINI KAMATA IZRAČUNAVA? - 12
NETAČNI PODACI; PONOVO POKUŠAJ
GLAVNICA? 15000
KAMATA ZA GODINU DANA? 10
BROJ GODINA? 3
KOLIKO SE PUTA U GODINI KAMATA IZRAČUNAVA? 12
GLAVNICA PLUS KAMATA = $ 20222,6
GLAVNICA? - 999
KRAJ POSTUPKA
```

Komentar rešenja

Vaš program neće biti isti kao i dato rešenje; tamo gde se znatno razlikuje zastanite za momenat i razmislite šta je bolje. Nemojte se iznenaditi ako je vaše rešenje bolje.

Određivanje starosti pomoću radioaktivnog ugljenika 14

Kada se pronađe neki stari predmet, npr. fosil ili mumija, veoma često je važno da se odredi starost tog predmeta. Jedna od metoda je i metoda pomoću radioaktivnog ugljenika.

Detaljnije o metodi određivanja starosti pomoću radioaktivnog ugljenika 14

Naučnici određuju procenat P ugljenika 14 (radioaktivnog izotopa) u nekom predmetu u odnosu na ugljenik 14 u atmosferi, a kako nam je poznata brzina radioaktivnog raspadanja R, na sledeći način može da se odredi starost predmeta T u godinama:

$$\frac{P}{100} = e^{-RT} \qquad \ldots\ldots (1)$$

R se određuje iz H, poluživota, što predstavlja vreme u godinama koje je potrebno da izotop ugljenika izgubi polovinu svoje radioaktivnosti. Tako,

$$R = \log 2 / H \qquad \ldots\ldots (2)$$

Kombinujući (1) i (2), dobijamo

$$T = -H \log \left(\frac{P}{100}\right) / \log 2$$

$$= H \text{ Abs } (\log \frac{P}{100}) / \log 2$$

gde je H jednako 5730 godina. Stoga za zadato P, može da se izračuna T.

Postavljanje problema

Za dato P, procenat „ugljenika 14" u nekom predmetu, naučnici mogu da izračunaju starost T (u godinama) nekog predmeta pomoću sledećeg obrasca:

$$T = 5730 \text{ Abs } (\log \frac{P}{100}) / \log 2$$

Od vas se zahteva da napišete dobro dokumentovan program koji prihvata ulaz P od strane korisnika a daje kao rezultat starost predmeta T.

Rešenje

```
 10    REM      OVAJ PROGRAM ODREĐUJE STAROST NEKOG PREDMETA
 20    REM          METODOM RADIOAKTIVNOG UGLJENIKA 14.
 30    REM              P = PROCENAT UGLJENIKA 14
 40    REM                  U PREDMETU U ODNOSU NA
 50    REM                  UGLJENIK 14 KOJI SE NALAZI
 60    REM                  U ATMOSVERI
 70    REM              T = STAROST PREDMETA U GODINAMA
 80    REM      UKUCAJTE P = -999 ZA ZAUSTAVLJANJE.
 90    PRINT    "ODREĐIVANJE STAROSTI POMOĆU RADIOAKTIVNOG UGLJENIKA 14"
100    PRINT
105    INPUT    "NIVO PROCENTA RADIOAKTIVNOSTI" P
110    IF  P = -999 THEN GOTO 180
120    IF  P < 0 THEN GOTO 160
125    IF  P > 100 THEN GOTO 160
130    T = 5730 * ABS (LOG (P/100)) /LOG(2)
140    PRINT "STAROST NEKOG PREDMETA U GODINAMA = "; T
150    GOTO 100
160    PRINT "NETAČNI PODACI : POKUŠAJTE PONOVO"
170    GOTO 100
180    PRINT "KRAJ POSTUPKA" : END
RUN

ODREĐIVANJE STAROSTI POMOĆU RADIOAKTIVNOG UGLJENIKA 14
NIVO PROCENTA RADIOAKTIVNOSTI ? 52
STAROST NEKOG PREDMETA U GODINAMA = 5405,78
NIVO PROCENTA RADIOAKTIVNOSTI? 120
NETAČNI PODACI : POKUŠAJTE PONOVO
NIVO PROCENTA RADIOAKTIVNOSTI? 80
STAROST PREDMETA U GODINAMA = 1844, 65
NIVO PROCENTA RADIOAKTIVNOSTI? -999
KRAJ POSTUPKA
```

3.9. **Rad sa petljama (upotreba petlje FOR/NEXT)**

Svakako se sećate kako program od 3 reda u odeljku 3.4. može da izračunava površine svih trouglova na svetu. Trik se sastojao u upotrebi naredbe GOTO za grananje nazad na početak programa. Postoje i drugi načini da se računaru kaže da se pokorava bloku naredbi. Takav blok se naziva petlja. Govorimo o okretanju u petlji (tj. bloku instrukcija) n puta ili jednostavno o petljanju n puta.

Pokušajte sledeće

```
10    FOR I    = 1 TO 100 STEP 1
20    PRINT      "PECOM-32"
30    PRINT      "JE PAMETAN"
40    NEXT I
RUN
```

Šta se dešava? Ispunjavate svoj ekran redovima i kolonama poruka **PECOM-32** JE PAMETAN stotinak puta.

Naredba FOR u redu 10 i naredba NEXT u redu 40 rade zajedno kao par „FOR/NEXT". Ovaj par kaže računaru **PECOM-32** da 100 puta izvrši naredbe između njih (red 20 i red 30), tj. za I=1 do I=100 koristeći veličinu koraka 1.

Opet pokušajte

```
10    FOR I = 0 TO 2000 STEP 2
20    PRINT I;
30    NEXT
RUN
```

Ekran je ispunjen sa „024 ... 2000". U ovom primeru veličina koraka je 2 te se, stoga, prikazuje 1001 broj. Ovako izgleda analiza od reda do reda:

Tabela 3.9. Opis akcije petlje FOR / NEXT od rada do rada

Br. reda	Akcija
10	Postavi I = 0
20	Prikaži 0
30	Povećaj I za 2, (veličinu koraka), i vrati se na 10
10	I = 2
20	Prikaži 2
30	Povećaj I za 2, vrati se na 10
10	I = 4
20	Prikaži 4
30	Povećaj I za 2, vrati se na 10
10	I = 6
	.
	.
	.

Ovo će se nastaviti sve do I = 2000 i zaključno s njim petlja će se zaustaviti. Obratite pažnju na to da se u ovom slučaju PRINT naredba izvršava u intervalima, naime red 20 se menja sa I.

Veličina koraka može biti negativna. Pokušajte

```
10    FOR A = 11 TO 8 STEP-1
20    PRINT A
30    NEXT A
40    PRINT "KRAJ"

RUN
11
10
 9
 8
 KRAJ
```

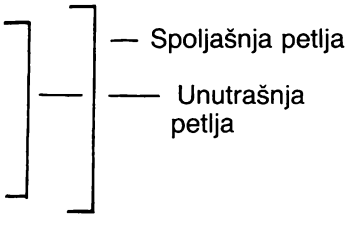
Ako veličina koraka nije specificirana, pretpostavlja se veličina koraka od 1. Takođe, postoji mogućnost da petlja FOR/NEXT bude u okviru neke druge petlje FOR/NEXT. Ova situacija se opisuje kao „PETLJA U PETLJI"

Pokušajte

```
10    FOR    A = 1 TO 3
20    FOR    B = 1 TO 6      STEP 2
30    PRINT A, B

40    NEXT B
50    NEXT A
60    PRINT "KRAJ"

RUN
```



```
1        1
1        3                                    A = 1
1        5
2        1
2        3                                    A = 2
2        5
3        1
3        3                                    A = 3
3        5
KRAJ
```

Unutrašnja petlja ponavlja se tri puta za svaku spoljašnju petlju, tj. za svaku vrednost za A. Sledite gornji program, registrujući kako se menjaju vrednosti za A i B, a posebno kada se A menja od 1 do 2, B počinje sa B = 1.

Kao poslednji primer, pokušajte

```
10      FOR A = 1 TO 10
20      IF A = 4 THEN A = 9
30      PRINT A
40      NEXT A
50      PRINT "KRAJ"

RUN
1
2
3
9
10
KRAJ
```

Obratite pažnju na to kako naredba IF u redu 20 smanjuje petlju od 10 na 5

3.10. **Podela složenog programa u blokove (upotreba potprograma) -GOSUB i RETURN**

Zamislite da treba da napišete program koji će vam omogućiti da igrate igru koja je slična igri „Svemirski osvajač". Zbog toga što su u stanju da napišu tako složen program mnogi su postali bogati i poznati. Sa pametnim BASIC-om računara **PECOM-32**, COLOR grafikom i zvučnim efektima, vi ste u stvari dobro opremljeni da ovo uradite. Potrebno je naime učiniti određeni napor koji je istovremeno i zadovoljstvo i naporan rad pomešan sa maštom. Kako da počnete?

Jedan od načina je da identifikujete raznovrsne poslove koji treba da se ponavljaju u intervalima, npr. posao 1 je nacrtati svemirski brod na bilo kom mestu na ekranu; posao 2 je nacrtati lansiranje rakete; posao 3 je načiniti grafiku u boji i zvučni efekat kada je raketa pogođenja; posao 4 je načiniti grafiku u boji i zvučni efekat kada je pogođen lanser rakete; posao 5 je odrediti da li je svemirski brod pogođen, itd.

Posao 1 je obavljen pomoću bloka programa. Ako ubacimo ovaj blok programa na mesta gde treba da se nacrta svemirski broj, program će stvarno biti veoma dugačak. Drugi način je upamtiti negde kopiju programa za posao 1 (koji se naziva potprogram), i ubaciti „GOSUB" naredbu u glavni program tamo gde treba da se nacrta svemirski brod.

Obratite pažnju na to da je u „GOSUB N" N broj reda prve instrukcije u potprogramui. Takođe, obratite pažnju na to da se potprogram uvek završava naredbom RETURN.

Korak 1: Kada računar naiđe na „GOSUB 5000" u redu 110, on prenosi kontrolu na početak potprograma u redu 5000.

Korak 2: Naredbe u potprogramu izvršavaju se sve dok se ne dostigne naredba RETURN.

Korak 3: Upravljanje se prenosi na red 120, odmah ispod naredbe „GOSUB" u redu 110.

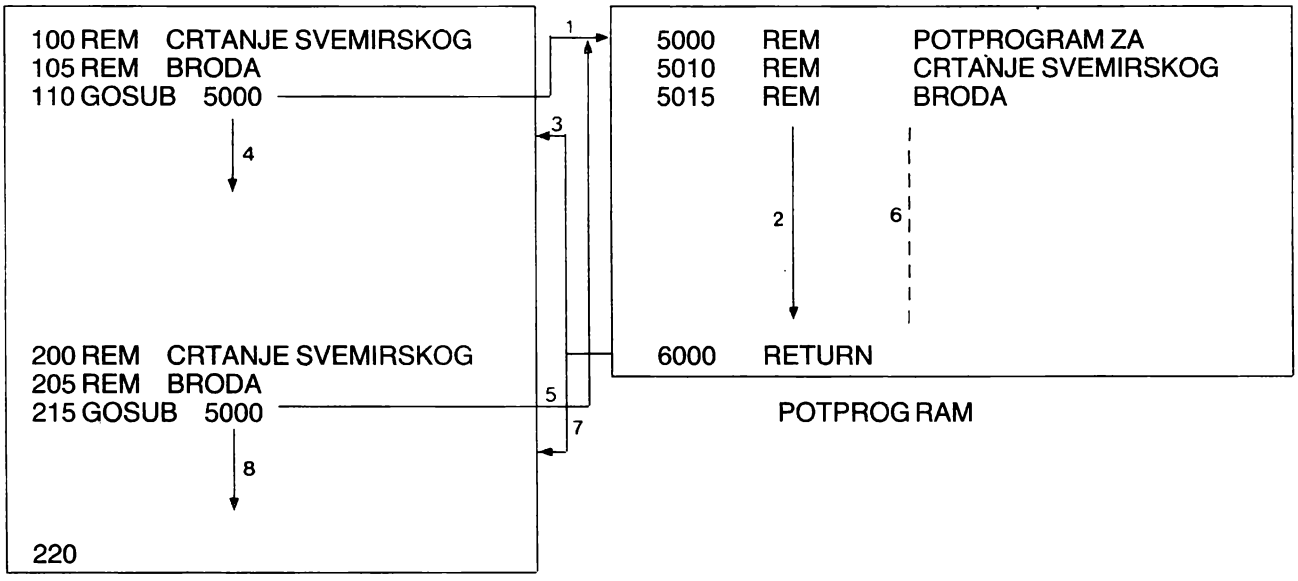
Korak 4: Nastavlja se izvršavanje glavnog programa.

Korak 5: Računar nailazi na drugo „GOSUB 5000" u redu 210 i, kao u koraku 1, kontrola se prenosi na početak potprograma.

Korak 6: Isto kao i za korak 2.

Korak 7: Kontrola se vraća na red 210.

Korak 8: Izvršavanje glavnog programa se nastavlja.



GLAVNI PROGRAM POTPROGRAM

NAJVEĆI ZAJEDNIČKI DELITELJ

```
 10      REM    OVO JE GLAVNI PROGRAM
 20      REM    NAJVEĆI ZAJEDNIČKI DELITELJ ZA CELE BROJEVE
 30      REM    A, B, C
 40      REM    "OVAJ PROGRAM NALAZI NAJVEĆI ZAJEDNIČKI DELITELJ"
 43      PRINT
 50      INPUT "A", A
 55      IF A < 0 THEN GOTO 150
 60      INPUT "B", B
 70      INPUT "C", C
 80      X = A
 90      Y = B
100      GOSUB 1000
110      X = C
120      GOSUB 1000
130      PRINT "NAJVEĆI ZAJEDNIČKI DELITELJ = " ; Y
140      GOTO 50
150      PRINT "KRAJ PROCESA"; END
1000     REM OVAJ POTPROGRAM NALAZI NZD
1010     REM DVA CELA BROJA X I Y
1020     REM I NZD = F
1030     REM
1040        Q = INT (X/Y)
1045     R = X - Q * Y
1050     IF R = 0 THEN GOTO 1100
1060     X = Y
1070     Y = R
1080     GOTO 1040
1100     RETURN
```

3.11. **Dalji saveti u vezi sa izmenom programa**

Pri pokušaju da se napiše duži program, postoji veća mogućnost da se greške češće načine a sredstva za izmenu koja su opisana u odeljcima 2.1 i 3.3 mogu biti neadekvatna. Pored upotrebe dirke „DEL", "CTL" i „brojeva redova" za izmenu (vidi odeljke 2.1 i 3.3), računar **PECOM 32** u stvari pruža bolja sredstva za ispravku grešaka u nekom programu.

Ovo se radi pomoću naredbe EDIT. Ova naredba ima oblik EDIT X gde je X ceo broj koji označava broj reda komande u kojoj se zahteva ispravka.

Kao odgovor na EDIT X, prikazuje se red X. Sada se kaže da je računar u „režimu izmene" (tj. pod kontrolom EDITOR-a umesto u režimu BASIC-a **PECOM-32**). Korisnik može sada da pomeri kursor (pritiskom na dirku za razmak) u položaj ispod prikazanog reda jedan znak ispred znaka koji treba da se modifikuje. U ovom momentu na raspolaganju stoje tri EDIT opcije. Korisnik može da unese I (za ubacivanje), D (za brisanje) ili C (za izmenu). Nakon izbora opcije, znak za opciju (I, D ili C) prikazuje se a kursor je sada direktno ispod znaka koji treba da se modifikuje.

Opcija za ubacivanje dopušta da se znaci unesu ili ubace odmah posle položaja koji se specificira sa I. Opcija za brisanje uklanja znake iz reda sekvencijalno sa svakim pritiskom dirke za razmak tj. pritiskom na dirku za razmak n puta obrisaće se n znakova. Opcija za izmenu zamenjuje sekvencijalno svaki znak novim znakom koji se unese posle C.

Pošto je korisnik obavio željenu modifikaciju, pritiskom na dirku S dok je dirka CTL pritisnuta, red X će se ponovo prikazati ali sada modifikovan. U jednom prolazu dozvoljava se samo jedna opcija (I, D ili C). Ako je potrebno, korisnik može da obavi jednu od tri opcije onoliko puta koliko je potrebno da bi se izvršile sve korekcije. Kada se ovo uradi, opet pritisnite CTL i S da „izađete" iz režima EDIT i da pokažete da je izmena reda X završena.

Gore izneto može se ilustrovati na sledeći način („u" dirka za razmak):

```
10   PRINT    "ZDRAVO"
```

Da biste korigovali red 10 unesite EDIT 10 i dobićete sledeće:

```
: EDIT 10
10   PRINT    "ZDRAVO"
```

gde je 10 red koji treba da se koriguje. Pozicionirajte sada kursor ispred znaka sa kojim počinjete izmenu.

Da biste ubacili KAŽI pre ZDRAVO u redu 10, odaberite opciju I, i unesite KAŽI iza koga dolazi razmak da biste dobili

```
:   EDIT     10
10  PRINT    "ZDRAVO"
             IKAŽIu
```

Sada pritisnite dirke CTL i S da biste dobili

```
10  PRINT    "KAŽI ZDRAVO"
```

Unesite ponovo CTL i S da biste izašli iz režima EDIT.

Da biste obrisali KAŽI i izmenili ZDRAVO u MILAN, odaberite opciju D iza koje dolaze četiri znaka za razmak.

```
:   EDIT     10
10  PRINT    "KAŽI ZDRAVO"
             Duuuu
```

Za svaki razmak posle D, briše se po jedan znak. Pritisnite CTL i S da biste prikazali delimično korigovan red. Pošto izmena još nije završena, ne izlazimo iz režima EDIT u ovom momentu. Nastavljamo i pozicioniramo kursor ispod " u delimično ispravljenom redu i biramo opciju C za izmenu.

```
10  PRINT    "ZDRAVO"
             CMILANu
```

Pritiskom na CTL i S prikazaće se

```
10  PRINT    "MILAN"
```

Šest znakova ZDRAVO zamenjeno je sa pet znakova MILAN i jednim blanko znakom. Sada kada je posao izmene završen, opet pritisnite CTL i S da biste izašli iz režima izmene. Računar će odgovoriti sa:

```
READY
:
```

što pokazuje da se kontrola sada vraća na BASIC intrepretator.

Kad se radi o grešci u nekom kratkom redu, ponovo kucanje celog reda uz upotrebu broja reda predstavlja najjednostavniji način da se izvrši korekcija. Kada je u pitanju dugačak red, postoji mogućnost da se načine nove greške dok se vrši ispravka starih grešaka. U ovom slučaju poželjnije je korišćenje EDIT režima.

Pritiskom na CTL i S dok se nalazite u režimu EDIT, završićete režim izmene i vratićete se u režim BASIC-a sa zadnjom izvršenom izmenom u redu.

3.12. **Generisanje specijalnih grafičkih znakova**

3.12.1. **Ugrađeni „grafički znaci" – upotreba funkcije CHR** $

Da biste dobili uvid u ugrađene grafičke znake, unesite sledeći program:

Program 3.12.1.

```
10    FOR I=0 TO 127
20    PR I, CHR $ (I); "    ";
25    IF I=127 THEN PR
30    NEXT
RUN
```

Primena funkcije CHR $ (X) detaljno je opisana u odeljku 5.5. Ukratko rečeno CHR $ (X) je funkcija koja generiše znak za svaki dat ceo broj X. Svakoj dirki odgovara neki ceo broj ili kod (tačnije rečeno ASCII kod, gde ASCII znači „Američki standardni kod za izmenu informacije) i neka dirka, npr. dirka A ima za kod broj 65 (65 decimalno ili 41 heksadecimalno). Pritiskom na dirku A normalno će se kao rezultat prikazati znak A. Stoga je "PR CHR $ (65) ekvivalentno sa PR "A". Dati program će na taj način prikazati svih 128 znakova čiji su decimalni kodovi brojevi od 0 do 127.

Kada se **PECOM-32** uključi, dirke koje imaju decimalne kodove od 32 do 107 daju standardne znake: SP,!, itd. do 1,2, ... 9, A, B, C ... Z, itd.) kao što je prikazano u Dodatku A.2. Dirke koje imaju decimalne kodove 0 do 32 i 108 do 127 daju ugrađene grafičke znake.

3.12.2. **Znaci koje definiše korisnik – uz upotrebu naredbe CHRGEN**

Postoji mogućnost da se ponovo definišu znaci koji su već ugrađeni u računaru **PECOM-32**. Npr. može se ponovo definisati znak za decimalni kod 65 tako što će se svi znaci A prikazani na ekranu zameniti nekim drugim znakom. Pretpostavimo da umesto znaka A hoćemo da prikažemo neki drugi znak.

Ovo ćemo uraditi primenom naredbe CHRGEN koja ima oblik (za TV koji koristi PAL sistem)

CHRGEN (X, "18heksadecimalnih brojeva")

gde je X ceo broj koji je ustvari decimalan kod znaka. U okviru para dvostrukih navodnika je 9 bajtova (18 heksadecimalnih brojeva) koji zajedno specificiraju boju i oblik znaka. Unesite,

```
10    CHRGEN (65,"48487E6A7E48484848")
RUN
```

(Napomena: CHRGEN se može upotrebiti i u režimu sa direktnim izvođenjem bez broja).

Nakon izvršenja naredbe CHRGEN sva slova A koja su prikazana na ekranu zameniće se znakom ◫.

Kod TV koji koriste NTSC sistem, umesto da koristite 18 heksadecmilanih brojeva, upotrebite 16 heksadecimalnih brojeva da biste genreisali oblik znaka.

Za definisanje 9 parova heksadecimalnih brojeva, razmotrimo pravougaonik sa slike 3.12.

Sl. 3.12. Pisanje naredbe CHRGEN za znak ◫

1	0	1	0	0	1	0	0	0	(48)	heksadecimalno
2	0	1	0	0	1	0	0	0	(48)	
3	0	1	1	1	1	1	1	0	(7E)	
4	0	1	1	0	1	0	1	0	(6A)	
5	0	1	1	1	1	1	1	0	(7E)	
6	0	1	0	0	1	0	0	0	(48)	
7	0	1	0	0	1	0	0	0	(48)	
8	0	1	0	0	1	0	0	0	(48)	
9	0	1	0	0	1	0	0	0	(48)	

bitovi za definisanje boje

bitovi za definisanje oblika znaka

Posmatrajmo svaki red od 8 kvadrata. Dva krajnja leva kvadrata, koji treba da se popune nulom i jedinicom respektivno, određuju boju preostalih šest kvadrata prema tabeli 3.12.2. Svaki od šest kvadrata predstavljen je sa 1 ako je taj kvadrat „obojen" ili „ispunjen" a sa 0 ako nije (tj. ako ima istu boju kao pozadina). Na taj način, prve dve kolone kvadrata (sleva) upotrebljavaju se za definisanje boje, dok se šest kolona kvadrata (6×9 kvadrata) upotrebljavaju za definisanje oblika. Svaki red se kodira sa dva heksadecimalna broja. Kako postoji devet redova, 18 heksadecimalnih brojeva je potrebno za definisanje boje i oblika. Npr. razmotrimo 4. red sa slike 3.12. Pretpostavimo da znak želimo da obojimo u crvenu boju. Dva krajnja leva bita treba da budu „01" (vidi tabelu 3.12.2). Preostala šest bitova su „101010" kao što se vidi na sl. 3.12. i oni kodiraju „ispunjeni kvadrat" sa 1 a „prazan kvadrat" sa „0". Osam bitova „01101010" mogu se predstaviti sa dva heksadecimalna broja tj. sa 6A. Odatle je četvrti par heksadecimalnih brojeva jednak 6A (vidi sl. 3.12).

Razlaganje po znaku kod PAL sistema je 6×9 tačaka. Kod NTSC sistema potrebno je 8 parova heksadecimalnih cifara koji daju 6×8 tačaka. Svaki znak, u oba slučaja, može biti višebojan.

Tabela 3.12.2. Boje jednog reda kvadrata kao funkcija prva dva bita

Dva krajnja leva bita		Boja
0	0	CRNA
0	1	CRVENA
1	0	PLAVA
1	1	LJUBIČASTA

3.13 Upotreba kasetofona za pamćenje programa i podataka

3.13.1. Prenos programa sa računara na kasetofon korišćenjem naredbe PSAVE

Prvo, povežite kasetofon za upotrebu kao što je opisano u odeljku 2.1. Pretpostavimo da imate program u memoriji računara koji želite da upamtite za kasnije korišćenje. Unesite LIST da biste izvršili brzu proveru programa u memoriji da vidite da li ima neželjenih delova programa. Podesite komandu za jačinu na kasetofonu na neki odgovarajući položaj. Postavljanje jačine na maksimum obično zadovoljava za **PECOM-32**. Sada, postupite prema sledećem:

(1) .. Ubacite praznu kasetu u kasetofon i premotajte traku (brojač postavite na 0). Snimanje na već snimljenoj traci će, naravno obrisati prethodno snimljene podatke.

(2) .. Počnite sa snimanjem tako što pritisnete dirke "PLAY" i "RECORD".

(3) .. Unesite naredbu PSAVE za „pamćenje programa" i pritisnite RET da biste započeli sa prenosom programa na traku.

(4) .. Kada se završi postupak pod (3) pojavljuje se poruka „READY" koju prati ":". Ne zaboravite da zabeležite položaj trake koji odgovara kraju zapisanog programa radi kasnijeg korišćenja. Postupak pod (3) je u daljem tekstu detaljnije opisan. Kada se unese PSAVE i pritisne dirka RET, kursor će se pojaviti na početku sledećeg reda i čuće se zvuk visokog tona sa dužim trajanjem koji obeležava početak programa (zaglavlje programa).

Iza toga se čuje šum koji označava da se vrši prenos prve stranice (256 bajtova) programa. Posle ovoga sledi šum visokog tona, zaglavlje strane, koji označava početak druge strane, a zatim drugi šum koji pokazuje da se vrši prenos druge strane programa. Zaglavlje strane visokog tona i šum se smanjuju sve dok se ne prenese ceo program. Kraj prenosa se označava zvukom visokog tona posle koga se čuje zvuk niskog tona. Pošto se navikne na gornji postupak korisnik može da zaključi da li se radi o normalnom prennosu.

PECOM-32 je tako konstruisan da šalje zvučni signal kasetofonu preko kabla koji se povezuje na dva džeka MIC. Signal ide kroz kasetofon i vraća se u računar preko kabla povezanog na dva džeka EAR (slušalice). Ako postoji pogrešan i/ili labav spoj, ili ako kasetofon nije uključen, putanja zvučnog signala se prekida i on se ne čuje. Ovo opominje korisnika da proveri spojeve.

Treba napomenuti da svi kasetofoni neće dati gore pomenuti zvučni redosled dok se vrši snimanje. Ako je ovo slučaj, ne brinite, i dalje možete koristiti svoj kasetofon sa računarom **PECOM-32**, ali se zvuk neće čuti.

3.13.2. Pamćenje podataka korišćenjem naredbe DSAVE

Da biste sačuvali podatke umesto programa, postupite prema (1) i (2) osim što u (3) umesto PSAVE upotrebite DSAVE za „pamćenje podataka". Sadržaj tog dela memorije računara rezervisan za podatke (brojevi, nizovi i/ili polja) biće upamćena na traci.

Isti redosled zvukova koji je opisan u prethodnom odljku ukazuje na pravilno funkcionisanje procesa pamćenja podataka.

3.13.3. **Prenos programa sa kasetofona na računar korišćenjem naredbe PLOAD**

Da biste preneli ili smestili program, koji je upamćen na magnetofonskoj traci u memoriju računara, treba najpre da povežete kasetofon kao što je opisano u odeljku 1.2. Zatim postupite na sledeći način:

(1) .. Ubacite kasetu u kasetofon i pronađite početak programa.

(2) .. Uverite se da vam program ili podaci, koji se sada nalaze u memoriji, nisu više potrebni ili da su već zapamćeni. Ovo treba učiniti zbog toga što će korak (3) izbrisati celokupnu korisničku memoriju i svaki program i/ili podaci u memoriji biće izgubljeni.

(3) .. Pretpostavljamo da je računar uključen i da je na ekranu prikazana poruka „READY" nakon koje dolazi „:" . Sada unesite PLOAD i pritisnite prekidač PLAY. Kada se čuje zvuk visokog tona, pritisnite dirku RET da biste započeli snimanje programa. Kursor će se prebaciti na novi red. Sada možete čuti niz zvukova koji je opisan u odeljku 3.13.1.

(4) .. Kada se završi učitavanje programa, ponovo će se pojaviti poruka „READY" koju prati „: ".

3.13.4. **Učitavanje podataka sa kasete korišćenjem naredbe DLOAD**

Naredbom DLOAD (učitavanje podataka) učitavaju se sa trake prethodno upamćeni podaci sa DSAVE Postupak je sličan postupku opisanom u odeljku 3.13.3. osim što se umesto PLOAD koristi DLOAD! Podaci se automatski smeštaju na kraj postojećeg memorijskog prostora za program i to preko postojećih podataka.

3.13.5. **Dalje napomene u vezi sa DLOAD**

Tri stvari se moraju uzeti u obzir kada se radi o upotrebi naredbe DLOAD:

(1) .. Ako korisnički memorijski prostor sadrži i program i podatke, svaka promena programa obrisaće podatke. Stoga, pre nego što se započne bilo kakva promena, unesite DSAVE da bi sačuvali podatke, izmenite program i unesite DLOAD da biste uneli podatke.

(2) .. Sa izvršenjem DLOAD automatski se dimenzioniše svako polje u okviru upamćenih podataka.

(3) .. Nizovi se smeštaju na kraju oblasti za polja. Ako se nova polja dimenzionišu nakon što su nizovi generisani, nizovi će biti izbrisani porastom oblasti za polja. Zbog toga, dimenzionisanje polja treba da se obavi pre nego što se generiše neki niz.

3.13.6. **Dalje napomene o smeštanju ili učitavaju programa sa trake**

Da bi se sprečio gubitak vrednih programa i podataka, treba voditi računa o sledećim stvarima:

(1) .. Napravite više od jedne kopije (višestrukim korišćenjem naredbi PSAVE i DSAVE) programa koji predstavljaju višečasovni posao i smestite ih i na drugoj traci.

(2) .. Slušajte redosled zvukova koji pruža dobru indikaciju o uspešnom prenosu.

(3) .. Posle koraka jedan, ponovo učitajte upamćeni program sa trake u računar i izlistajte ga. Ako učitavanje ne može uspešno da se obavi originalni program treba da ostane nedirnut i pokušajte da ponovo smestite program na traku.

(4) .. Pažljivo rukujte sa trakama da biste izbegli njihovo oštećenje. Takođe, izbegavajte da stavljate trake u blizini jakih magnetnih polja. Čuvajte trake daleko od TV. Vidite, takođe Dodatak A.3 „Dvadeset važnih upustava za snimanje na kasetu".

3.14. **Funkcije kontrolne dirke CTL**

Ako se izvesne dirke pritisnu dok je pritisnuta kontrolna dirka CTL računar će preduzeti izvesne akcije.

CTL–C

Kao što je rečeno u odeljku 2.1. ako se pritisne dirka C dok je pritisnuta dirka CTL (operacija čija je skraćenica „CTL C"), pre pritiska na RET, red koji se prikazuje biće ignorisan i kursor će postaviti u krajnji levi položaj u sledećem redu. CTL C omogućava korisniku da da računaru informaciju o tome da ignoriše delimično otkucanu naredbu.

CTL–H

Pritiskom na dirku H za vreme dok se drži dirka CTL kursor se pomera za jedan znak unazad, a znak preko koga se prelazi se automatski briše. Na ovaj način moguća je korekcija pogrešno unetih naredbi i podataka.

CTL–A

Ukoliko se računar nalazi u fazi izvršenja EDIT naredbe u nekoj od tri opcije, pritiskom na dirku A dok se dirka CTL drži, ignoriše se izvršena opcija i može se izaći iz EDIT naredbe ili ponovo otpočeti neka od C, D ili I opcije.

CTL–D

Pritiskom na dirku D za vreme dok je pritisnuta dirka CTL briše se ekran.

CTL–S

U toku izvršenja EDIT funkcije, aktiviranjem CTL–S pojavljuje se nova varijanta reda sa ispravkama, a ponovnim startovanjem CTL–S završava se editovanje i upravljanje se vraća na READY.

3.15. Kontrola programa u realnom vremenu–uz upotrebu dirke KEY i komandne palice (JOYSTICK)

KEY

Ova funkcija daje decimalni ekvivalent koda pritisnute dirke na tastaturi. Ukoliko nijedna dirka nije pritisnuta daje se vrednost 0. Njena upotreba se može ilustrovati sledećim primerom:

```
10    IF KEY=65 THEN GOTO 30
20    GOTO 10
30    PRINT "DIRKA A PRITISNUTA"
40    END
```

U navedenom primeru programa funkcija KEY javlja se u redu 10. Kada program u toku izvršenja dođe do nje, prelazi se na skaniranje tastature. Skaniranje tastature se vrši sve dok se ne pritisne dirka čiji je decimalan kod 65. Kako znamo da dirka A daje decimalan kod 65, to znači da se čeka na pritisak dirke A. Kada se pritisne dirka A na ekranu računara će se ispisati poruka:

```
DIRKA A PRITISNUTA
READY
:
```

Ako se pritisne neka druga dirka, a ne dirka A, program će se vrteti u petlji.

Navedeni program ilustrujte tipičnu situaciju kada se upotrebljava funkcija KEY. Ona se veoma često koristi u „IF naredbama" koje su smeštene u petlji, tako da se tastatura skanira u intervalima, a izvesne akcije će se preduzeti ako se pritisne neka određena dirka. Ovo omogućava da se pritiskom na određenu dirku kontroliše tok programa.

(Napomena: Sledeće informacije mogu biti korisne za detaljnije razumevanje funkcije KEY. Kada se pritisne neka dirka, njen ASCII kod se šalje u registar ili u leč. Funkcije kao što se KEY ili INPUT, očitaće sadržaj ovog leča a zatim će ga izbrisati. Ako se ne pritisne ni jedna dirka, funkcija KEY dobija vrednost 0).

KOMANDNA PALICA (JOYSTICK)

Ugrađena komandna palica je uređaj koji kontroliše 4 dirke. Ona će uneti u računar jedan od 4 koda u skladu sa smerom u kome je palica pritisnuta ... Četiri koda (koji se neće prikazati) imaju ASCII decimalne kodove 94, 93, 91, 92, koji odgovaraju položaju komandne palice naviše, desno, naniže i levo. Ako se naredba „IF KEY= =94 THEN ... " pojavi negde u petlji, kad god se komandna palica prebaci u „gornji" položaj (koji ima ASCII decimalni kod 94), uslov u okviru IF naredbe se zadovoljava. Ovo omogućava komandnoj palici da kontroliše tok programa.

Unesite,

```
10      IF KEY=94 THEN GOTO 30
20    GOTO 10
30    PRINT "JOYSTICK JE AKTIVIRAN"
40    END
```

Program će prikazati poruku kada se komandna palica prebaci u „gornji" položaj.

4. OSNOVNA KONCEPCIJA ORGANIZACIJE RAČUNARA

4.1. Šta je računar?

Računar je uređaj koji je sposoban da prihvata, obrađuje i daje izlazne informacije. Informacije su u obliku električnih signala i predstavljaju alfanumeričke podatke (tj. tekst, reči i brojeve). Obrada uključuje matematičke i logičke operacije, sortiranje, pretraživanje i preuređivanje. Izlazna (obrađena) informacija ima tri glavne kategorije primene: (a) obrada podataka (uključujući poslovne i knjigovodstvene aplikacije, vođenje arhive, pretraživanje informacija, informacione sisteme za primenu u rukovođenju (da se olakša donošenje odluka), (b) naučna izračunavanja i projektovanje i (c) industrijsko upravljanje kod koga senzori ili pretvarači primaju ulazne informacije i pretvaraju ih u električne signale koji se, zatim, obrađuju na propisani način. Izlazni signali se tada koriste za aktiviranje raznovrsnih elektromehaničkih uređaja.

4.2.Glavni delovi računara

Probleme može da rešava računar ili čovek (operater) uz pomoć kalkulatora, olovke i papira. Preporučuje se da se uporede ove dve situacije i da se na taj način, uoče podsistemi računara. Upoređivanje je prikazano u tabeli 4.2.

Tabela 4.2.

Rešavanje problema od strane čoveka i računara – upoređivanje njihovih zahteva

Čovek	Računar
1. List papira – za pisanje specifikacije problema	Ulazni uređaj
2. Algoritam – uređeni redosled postupaka koji specificira postupak za rešenje problema	Program smešten u memoriji
3. Podaci za obradu	Podaci smešteni u meoriji
4. Kalkulator–koji obavlja osnovne operacije	Aritmetičko-logička jedinica
5. Čovek (operater)–koji upravlja izvršenjem algoritma	Upravljačka jedinica
6. Olovka i papir–za beleženje međurezultata	Memorija
7. Lista za beleženje krajnjih rezultata	Izlazni uređaj
8. Biblioteka referentnih knjiga	Masovna memorija

Iz tabele 4.2. se zaključuje da se računar sastoji od:

(a) Podsistema za obavljanje aritmetičko-logičkih operacija i drugih podsistema za upravljanje tokom izvršenja svakog postupka. Prvi se naziva aritmetičko-logička jedinica a drugi upravljačka jedinica. Ova dva podsistema se kod računara često spajaju u jednu centralnu jedinicu ili CPU. Centralna procesna jedinica se „povezuje“ sa čovekom koji ima kalkulator.

(b) Uređaja za pamćenje programa, ulaznih podataka i međurezultata. Ovi se kod računara pamte u glavnoj memoriji. Glavna memorija se „povezuje“ sa olovkom i papirom koje čovek (operater) koristi.

(c) Ulazno-izlaznih uređaja. Ulazno izlazni uređaji omogućavaju računaru da komunicira sa spoljašnjim svetom, da se preko njih unose programi za rešavanje nekog problema i prateći podaci (npr. sa tastature) kao i da se daju izlazni rezultati (npr. na TV ekranu).

(d) Uređaja za pamćenje velike količine informacija koji se ređe koriste, kao što je masovna memorija (npr. flopi disk ili traka). Oni su sporiji od glavne memorije ali mogu da upamte veliki broj informacija. Ovi uređaji se „povezuju“ sa bibliotekom referentnih knjiga koje čovek povremeno koristi.

Stoga su glavni podsistemi računara: centralna procesna jedinica, glavna memorija, ulazno-izlazni uređaji i masovne memorije.

4.3. Softver

Pod „softverom“ se podrazumevaju programi ili postupci korak po korak koji se daju računaru i koji mu saopštavaju kako da obavlja raznovrsne zadatke. Postoje tri glavne vrste softvera:

(a) Aplikacioni softver, za rešavanje specifičnih naučnih problema ili obradu podataka, npr. programi za vođenje opšteg knjigovodstva, kontrolu inventara i za projektovanje konstrukcija zgrada. Aplikacioni softver piše sam korisnik.

(b) Sistemski softver koji korisniku omogućava lagodniji rad sa računarom npr. programi koji mogu da obavljaju mnoštvo zadataka kao što su mogućnost da korisnik smesti program na kasetu a kasnije da ga ponovo unese u memoriju.

(c) Kompajleri i/ili interpretatori jezika, koji omogućavaju korisniku da svoje aplikacione programe napiše na jeziku bliskom njegovom prirodnom jeziku. Unutar računara programi i podaci se predstavljaju velikim brojem uređaja koji mogu da postoje u dva i samo dva „stanja" označena sa 1 i 0 (kao prekidač koji je ili otvoren ili zatvoren). Računar može direktno da interpretira ove oblike „jedinica-nula". Međutim, pisanje programa uz upotrebu samo jedinica i nula za korisnike je zaista težak ako ne i nemoguć zadatak, a programi koji bi iz toga proizašli ne bi bili razumljivi. Korisnik želi da postupak za rešenje problema izrazi jezikom koji je sličan njegovom prirodnom jeziku, kao što je BASIC i koji treba da se pretvori u oblik „jedinica-nula" (binarni) pre nego što računar može da ga izvrši. Ovakvo pretvaranje se ostvaruje pomoću programskih prevodioca koji se nazivaju „kompajleri" i/ili „interpretatori". Kompajler prevodi celokupan program koji je napisan na višem programskom jeziku u mašinski jezik koji se zatim izvršava. Interpretator prevodi i izvršava svaku naredbu u izvornom jeziku pre nego što pređe na narednu naredbu.

5 – VODIČ KROZ BASIC

5.1. Programi, naredbe, izrazi i funkcije

Program je skup naredbi koje računar može da izvrši a koji mu kaže kako da reši neki problem. On uključuje metodologiju u obliku procedure korak po korak koja se naziva **algoritam** i podatke nad kojima algoritam radi. Tako je „program = podaci + algoritam".

Naredba (kojoj prethodi broj reda) sastoji se od ključnih reči u BASIC-u (npr. END, NEXT), izraza i funkcije.

Funkcije su ključne reči u BASIC-u koje daje vrednosti, npr. LOG (X), SIN (X), itd. Pregled funkcija dat je odeljku 5.4. Vrednost u zgradi naziva se **argument** funkcije. Ključna reč definiše pravilo kojim se argument može transformisati u vrednost koja se naziva vrednost funkcije.

Izraz, što je skraćeni oblik za aritmetički izraz, može se sastojati od samo jednog broja, jedne promenljive, funkcije ili grupe operanata (broj, promenljiva i funkcija, itd.) povezanih međusobom aritmetičkim operatorima. Evo nekih primera:

8
5
A
3 * SIN(45)↑0,5
3 * (2 + A)/(B + C)

5.2. Brojevi

PECOM-32 obrađuje i cele brojeve i brojeve sa pokretnim zarezom. Pamte se kao 32. bitni označeni brojevi.

Brojevi sa pokretnim zarezom mogu da uzimaju vrednosti koje idu u opsegu od $-0{,}170141 \times 10^{39}$ do $+0{,}170141 \times 10^{39}$. Razlomljeni deo mantise tačan je do šest cifara.

Celi brojevi idu u opsegu od −2147483647 do 2147483647. Tačni su u celom opsegu.

Brojevi koji se unesu sa tastature pamte se u obliku u kome su uneti. Ako se unesu bez decimalnog zareza pamte se kao celi brojevi; ako se unesu sa decimalnom tačkom ili sa oznakom „E" pamte se kao brojevi sa pokretnim zarezom.

5.3. Operatori

Operatori se kod računara **PECOM-32** mogu podeliti u pet kategorija: aritmerički, za dodeljivanje, relacioni, logički i raznovrsni.

Aritmetički operatori

U sledećem pregledu dati su aritmetički operatori sa njihovim simbolima po prioritetu ili redosledu po kome se nad njima radi u matematičkom izrazu.

minus kao predznak	−	prvi prioritet
stepenovanje	↑	prvi prioritet
množenje	*	drugi prioritet
delenje	/	drugi prioritet
sabiranje	+	treći prioritet
oduzimanje	−	treći prioritet

Nad operatorima istog nivoa prioriteta radi se sleva u desno. Simbol „−" upotrebljava se i za obeležavanje negativnih brojeva. U daljem tekstu dati su neki primeri koji ilustruju prioritete.

4 * 3↑2 = 36 (3↑2 se prvo izvršava a onda 4 * 9)
4/2 * 3 = 6
4 + 2 * 3 − 5 = 5

Operator za dodeljivanje

Simbol za dodeljivanje jednak je znaku „=". Kada se ovaj simbol upotrebi za dodeljivanje, on ima značenje „uzima vrednost za" a ne „jednako". Evo nekih primera:

```
LET A = 5
LET B = B+A * 2
```

Ključna red LET je opciona i može da se izostavi. Drugi primer može da se interpretira kao: izračunajte B+A * 2 i upotrebite rezultat da biste zamenili B.

POKE (izraz1, izraz2)

Ova naredba, koja treba da se upotrebljava sa krajnjom pažnjom, koristi se da se neka memorijska lokacija napuni podacima. Lokacija u memoriji se definiše pomoću izraza1 a podaci se definišu pomoću izraza2. Na primer,

```
POKE (5000, 255)
```

smešta heksadecimalni ekvivalent za 255 (FF) u memorijsku lokaciju sa decimalnom adresom 5000. Uobičajeniji i sigurniji pristup je

```
POKE (& 1388, # FF)
```

što smešta heksadecimalan podatak FF u memorijsku lokaciju sa heksadecimalnom adresom 1388.

Relacioni operatori

Relacioni operatori su sledeći

Jednako	=
Nejednako	< >
Veće od	>
Manje od	<
Veće od ili jednako	> =
Manje od ili jednako	< =

Izrazi koji sadrže relacione operatore ocenjuju se kao istiniti i lažni a ne kao numerička vrednost. Evo nekih primera:

```
IF A = 3 + B↑ 2 THEN
IF A < > 4 THEN
IF A * 2 < = B/4 THEN
```

Logički operatori

Logički operatori su AND, OR, XOR i NOT. Treba napomenuti da logički operatori pretvaraju izraze u celobrojne vrednosti pre nego što počnu da obavljaju naznačenu operaciju i da izraz koji dolazi posle operatora NOT mora da bude u zagradi. Dalje, logički operatori imaju isti nivo prioriteta kao + i −. Evo nekih primera:

```
IF (A AND B) < > C THEN
FOR I=(A AND B) TO (A OR B) STEP A/B
PR (A XOR B)
GOTO NOT (A)
```

U prva tri primera zagrade su ubačene radi jasnoće. One se mogu izostaviti.

Raznovrsni operatori

Raznovrsni operatori su sledeći:

```
: odvaja naredbe u istom redu
; PRINT ograničavač
" "
'
```

" ograničavač niza
(grupni ograničavač
) " "
%...% 8-bitne binarne vrednosti
8-bitne heksadecimalne vrednosti
& 16-bitne heksadecimalne vrednosti

Evo nekih primera:

```
PRINT A,B: "DA": GOTO 10
LET A = (3+4)/2
LET B = #F8:C=& 01F3
IF A < > % 10011010 % THEN GOTO 100
```

Napomena 1: koristite znak ; u PRINT naredbi da biste prikazali sledeću stavku nakon poslednje bez razmaka između.

Napomena 2: Koristite znak , u PRINT naredbi da biste prikazali sledeću stavku nakon osam znakova za razmak.

Napomena 3: Upotrebite znak " da biste naznačili granice niza znakova.

5.4. **Matematičke funkcije**

Termin funkcija definisan je u odeljku 1. Argument matematičke funkcije može biti neki broj, promenljiva ili neki izraz. Funkcije se dele na ugrađene i izvedene. Izvedene funkcije su one koje se mogu dobiti iz nekog izraza koji uključuje ugrađene funkcije. Npr. SIN(X) i COS(X) su ugrađene funkcije, dok je TAN(X) izvedena funkcija, pošto se TAN(X) može izračunati kao SIN(X)/COS(X).

U početnom stanju računara **PECOM-32** (po uključenju) svi uglovi izraženi su u radijanima. Sledeći primeri podrazumevaju ovo stanje. Da bi se ovi pretvorili u stepene, unesite DEG i posle toga RET. Da biste ih ponovo vratili u radijane unesite RAD a zatim RET.

5.4.1. **Ugrađene funkcije**

Ugrađene funkcije su sledeće: ABS, ATN, COS, EXP, FNUM, INT, INUM, LOG, MOD, PI, RND, SGN, SIN i SQR.

ABS (izraz)

Ova funkcija daje apsolutnu vrednost izraza. Npr.

```
PR ABS (-10 * 2)
```

daće

```
20
```

ATN (izraz)

Ova funkcija, koja ima isto značenje kao i ARCTAN (izraz) daje kao svoju vrednost ugao (izažen u radijanima) čiji je tangens jednak vrednosti izraza. Ova funkcija je funkcija sa pokretnim zarezom.

Npr.

```
PR ATN (1)
```

daće

```
0,7854
```

zbog toga što je TAN (0,7854) = TAN PI/3 = TAN 45° = 1

COS (izraz)

Ova funkcija daje kosinus ugla određenog vrednošću izraza (izražen u radijanima). Ovo je funkcija sa pokretnim zarezom. Npr.

```
PR COS (PI/3)
```

daće

0.5

zbog toga što je cos 60° = 0.5

EXP (izraz)

Ova funkcija daje broj sa pokretnim zarezom 2.71828 (e) stepenovan vrednošću izraza. Npr.

PR EXP (6/3)

daće

7.3891

pošto je $e^{6/3} = e^2 = 2.71828^2 = 7.3891$

FNUM (izraz)

Ova funkcija zaokružuje izraz na najbliži ceo broj i pretvara ga u broj sa pokretnim zarezom. Npr.

PR FNUM (RND(6))

daće vrednost sa pokretnim zarezom.

INT (izraz)

Ova funkcija vraća celobrojni deo izraza sa pokretnim zarezom pri čemu se odbacuje razlomljeni deo. Rezultat je i dalje u pokretnom zarezu. Ova funkcija ne treba da se meša sa INUM (koja će kasnije biti objašnjena). Npr.

A = 7,9 : B = INT(A) : PR A,B

daće

7,9 7

INUM (izraz)

Ova funkcija pretvara izraz sa pokretnim zarezom u celobrojni, pri čemu se vrši zaokružavanje na najbliži ceo broj. Ona pruža mogućnosti za prebacivanje neke posebne funkcije u ceo broj.

Npr.

PR SQR (62,41)

imaće za rezultat

7,9

a

PR INUM (SQR (62,41))

imaće za rezultat

8

LOG (izraz)

Ova funkcija daje broj sa pokretnim zarezom čija je vrednost prirodni logaritam vrednosti izraza. Npr.

PR LOG (10)

daje

2.3026

a

PR LOG (EXP(2))

daje

2

MOD (izraz1, izraz2)

Ova funkcija vraća kao svoju vrednost sledeći ekvivalent

izraz1 – (izraz1/izraz2) * izraz2

gde se svaki izraz prvo pretvara u ceo broj pa je i vrednost funkcije ceo broj. Npr.

PR MOD (10,3)

imaće za rezultat

1

PI

Ova funkcija daje kao svoju vrednost 3.14159.

RND

Ova funkcija daje slučajan broj sa pokretnim zarezom koji je veći ili jednak nuli a manji ili jednak jedinici.

RND (izraz)

Ova funkcija daje slučajan ceo broj veći ili jednak nuli a manji ili jednak vrednosti izraza.

SGN (izraz)

Vrednost ove funkcije je + 1, 0 ili – 1 zavisno od znaka izraza. Ako je izraz pozitivan, funkcija daje + 1. Ako je izraz jednak nuli, funkcija daje kao svoju vrednost nula. Ako je izraz negativan, funkcija daje – 1.

PR SIN (PI/6)

daje

0.5

SIN (izraz)

Ova funkcija daje sinus ugla određenog vrednošću izraza (izražen u radijanima). Funkcija je sa pokretnim zarezom. Npr.

PR SIN (PI/6)

daje

0.5

SQR (izraz)

Ova funkcija izračunava kvadratni koren vrednosti izraza. Funkcija je sa pokretnim zarezom. Npr.

PR SQR (65)

daje

8.0623

5.4.2. Dalji primeri matematičkih funkcija

Sledeći primeri služe kao ilustracija snage matematičkih funkcija BASIC-a računara **PECOM-32**. Pošto argument funkcija može da bude proizvoljan izraz mogu se obrađivati veoma složeni matematički problemi. Npr.

```
PR SIN (45)
A = INT (10 * SQR(LOG(A * B)))
C = EXP (10+A)
PR ATN (A * SIN(A))
```

5.4.3. Izvedene funkcije

Sledeće funkcije nisu ugrađene funkcije biblioteke BASIC-a **PECOM-32** već se mogu izračunati iz ugrađenih funkcija navedenih u odeljku 5.4.1 uz upotrebu sledećih obrazaca.

```
TAN(X) = SIN(X)/COS(X)
ARC SIN(X) = ATN (X/SQR (1 – X * X))
ARC COS(X) = – ATN (X/SQR (1 – X * X)) + 1.5708
```

```
ARC SEC(X) = ATN (SQR (X * X-1)+(SGN (X)-1) * 1.5708
ARC CSC(X) = ANT (1/SQR (X * X -1)+(SGN (X)-1) * 1.5708
ARC COT(X) = -ATN (X)+1.5708
ARC SINH(X) = LOG (X+SQR (X * X+1))
ARC COSH(X) = LOG (X+SQR (X * X-1))
ARC TANH(X) = LOG ((1+X)/(1-X))/2
ARC SECH(X) = LOG ((SQR (1-X * X)+1)/X)
ARC CSCH(X) = LOG ((SGN (X) * SQR (X * X+1)+1)/X)
ARC COTH(X) = LOG ((X+1)/(X-1))/2
COT(X) = 1/TAN (X)
CSC(X) = 1/SIN (X)
SEC(X) = 1/COS (X)
COSH(X) =(EXP (X)+EXP (-X))/2
COTH(X) = EXP (-X)/(EXP (X)-EXP (-X)) * 2+1
CSCH(X) = 2/(EXP (X)-EXP (-X))
SECH(X) = 2/((EXP (X)+EXP (-X))
SINH(X) = (EXP (X)-EXP (-X))/2
TANH(X) = -EXP (-X)/(EXP (X)+EXP (-X)) * 2+1
```

5.5 **Nizovi i funkcije nizova**

Niz je redosled znakova (alfanumeričkih i specijalnih znakova kao što su +, −, /, *, $ itd.). U niz se može uključiti i razmak ali ne može navodnik zbog toga što su navodnici ograničavači niza, tj. oni se upotrebljavaju da obeleže početak i kraj niza. Za BASIC računara **PECOM-32** svaki niz može da sadrži najviše 127 znakova.

Nizovi se upotrebljavaju za predstavljanje nenumeričkih podataka kao što su labele, poruke, itd. Upravo onako kako mogućnost rada sa brojevima omogućava računaru da obavlja naučna izračunavanja, tako i rad sa nizovima omogućava računaru da obavlja obradu reči, vođenje arhive i pretraživanje informacije.
Funkcije za rad sa nizovima su: ASC, CHR$, FVAL, LEN i MID$.

ASC (izraz niza)

Ova funkcija daje kao svoju vrednost decimalni ekvivalent ASCII vrednosti prvog znaka u datom nizu i to u pokretnom zarezu. Ako je ovoj funkciji prethodila neka celobrojna funkcija njena vrednost će se takođe pretvoriti u ceo broj. Npr.

```
A$(5) = "ARITMETIKA"
B = ASC(A$(5))
```

Promenljiva B će sadržati kao svoju vrednost, 65, decimalnu ASCII vrednost slova A.

Drugi primer:

```
 5 A$ = "TEST"
10 FOR A=1 TO LEN (A$)
20 PR ASC (MID$(A$, A)
30 NEXT A
```

imaće za rezultat

```
84
69
83
84
```

CHR$ (izraz, izraz, ...)

Ova funkcija kao svoju vrednost daje znak u skladu sa izrazom. Vrednost izraza jednaka je ASCII kodu znaka koji funkcija CHR daje. Npr.

```
PR CHR$ (65)
```

imaće za rezultat

```
A
```

zbog toga što je ASCII decimalan kod za A jednak 65.

```
PR CHR$ (#42)
```

imaće za rezultat

```
B
```

zbog toga što je ASCII heksadecimalan kod za B jednak 42.

```
PR CHR$ (#41, #42, #43)
```

imaće za rezultat

```
ABC
```

Neke primere upotrebe CHR$ vidi u odeljku 3.12. U dodatku A.2 dati su ASCII kodovi za sve ugrađene znake.

FVAL (izraz niza)

Ova funkcija je specijalna za **PECOM-32**. Ona izračunava izraz niza kao aritmetički izraz i daje njegovu vrednost. Na taj način korisnik može da kreira matematičke funkcije. U daljem tekstu data su tri primera upotrebe funkcije FVAL.

```
10 A$ = "8+4"
20 PR FVAL (A$)
30 PR FVAL (A$+"/2")
40 PR FVAL ("SQR(3)")

:RUN        (poziv programa)
12          (izračunati i prikazati A$)
10          (    "          "      FVAL ("8+4/2"))
1.73205     (    "          "      SQR (3))
READY
:
```

Niz je ograničen na 48 znakova.

LEN (promenljiva niza)

Ova funkcija daje kao svoju vrednost broj znakova u specificiranom nizu. Vrednost se daje u pokretnom zarezu osim ako joj ne prethodi ceo broj ili funkcija, u kom slučaju se ona automatski pretvara u ceo broj. Na primer:

```
A$ = "ARITMETIKA"
PR LEN (A$)
```

daje

```
10 (dužina A$)
```

a

```
PR 3 * LEN (A$)
```

daje

```
30
```

Obratite pažnju na to da niz mora prethodno da se definiše.

MID$ (promenljiva niza, izraz1)
MID$ (promenljiva niza, izraz1, izraz2)

Ova funkcija za niz generiše podniz specificiranog niza. Izraz1 definiše koji znak sleva treba da započne podniz. Izraz2 definiše broj znakova koji treba da se upotrebe u podnizu. Ako se ne upotrebi izraz2, upotrebljavaju se svi preostali znaci. Npr.

```
A$ = "DA"
B$ = MID$ (A$, 2, 1)
```

B$ će tada sadržati slovo "A"

```
A$ (11) = "EKSPERIMENT"
PR MID$ (A$ (11), 8, 4)
```

imaće za rezultat

```
"MENT"
```

koje se prikazuje.

```
10 INPUT A$ (1)
20 IF MID$ (A$ (1), 1, 1) = "Y" GOTO 100
30 GOTO 10
100 END
```

U zadnjem primeru čeka se na unos niza sa tastature. Ukoliko niz koji se unese počinje slovom Y, izvršenje programa se nastavlja sa redom 100. Ukoliko to nije slučaj program se vrti u petlji.

Zajedničke funkcije kao što su LEFT$ i RIGHT$ drugih BASIC-a mogu se ostvariti pomoću funkcije na sledeći način:

```
MID$ (A$,1,N)
```

je isto kao i

```
LEFT$ (A$,N)
```

a

```
MID$ (A$, (LEN (A$)−N+1), LEN (A$))
```

je isto kao i

```
RIGHT$ (A$,N)
```

Povezivanje nizova

Povezivanje je proces sabiranja dva ili više niza kao što je npr. povezivanje nizova "ZDRAVO" i "ANA" u novi niz "ZDRAVO ANA". Kod povezivanja koristimo znak + i formiramo naredbu dodeljivanja koristeći nizove i/ili promenljive niza. Npr.

```
10 A$ = "KRATKO"
20 B$ = "SPOJ"
30 LET A$ (5) = "NIK"
40 LET B$ (1) = A$+B$+A$ (5)
50 PRINT B$ (1)
```

Rezultat reda broj 40 biće generisanje novog niza B$ (1) koji sadrži tri niza koji su postavljeni jedan za drugim. "KRATKOSPOJNIK" će se prikazati na ekranu posle izvršenja reda 50.

Jedino ograničenje kod povezivanja nizova je u tome da svaki niz koji nastaje sabiranjem dva ili više niza ne sme da premaši granicu od 127 znakova.

Unošenje niza i korišćenje nizova u IF naredbama

Razmotrimo donji primer:

```
10 INPUT B$
20 A$ (1) = "NASTAVAK"
30 A$ (2) = "FAZA1"
40 A$ (3) = "FAZA2"
50 IF B$ = A$ (1)+A$ (2) GOTO 100
60 IF B$ = A$ (1)+A$ (3) GOTO 200
70 GOTO 10
```

Dati program će čekati u redu broj 10 da se unese niz sa tastature. Taj niz će biti označen sa B$. Tada će se uporediti sa A$ (1)+A$ (2) i A$ (1)+A$ (3) pri čemu se izvršenje programa prenosi na odgovarajuću tačku zavisno od saglasnosti ili nesaglasnosti.

5.6. Polja

Skup podataka, pri čemu svi nose isti naziv grupe, ali gde je svaki označen još i brojem (ili skupom brojeva) naziva se polje. Npr., $a_1, a_2 \ldots a_n = A(1:n)$ je jednodimenziono polje (ili lista) dimenzije n. Svaki element u listi a_i označen je brojem i koji obeležava položaj tog elementa u listi u odnosu na prvi slement u listi.

$$A(1:m, 1:n) = \begin{vmatrix} a_{11}, & a_{12}, & a_{1n} \\ a_{21}, & a_{22}, & a_{2n} \\ . & & \\ . & & \\ . & & \\ . & & \\ a_{m1}, & a_{m2} & a_{mn} \end{vmatrix}$$

je dvodimenzionalno polje (ili tabela). Svaki element u tabeli ima vrednost vrste i i vrednost kolone j koje zajedno definišu položaj elementa u polju.

Računar **PECOM-32** ima 26 polja A(izraz) do Z(izraz) i 26 nizova ili polja nizova A$ do Z$ i A$(izraz) do Z$(izraz). Ova 26 polja mogu biti jednodimenzionalna i dvodimenzionalna. Maksimalna veličina polja je 255 u svakoj dimenziji. Minimalna veličina svake dimenzije je 1. Najveće polje koje može da postoji je G(255, 255) a najmanje je G(1). Najveći broj elemenata polja ograničen je veličinom RAM-a koji stoji na raspolaganju. Svako polje koje se koristi mora da se dimenzioniše upotrebom naredbe DIM, koja je objašnjena u odeljku 5.9.

Memorijski prostor za polja briše se na nekoliko načina: RUN, NEW i CLD. Ovaj postupak razmatra se kasnije prilikom objašnjenja svake od ovih naredbi. Takođe je vredno znati koliko svako polje koristi memorijskog prostora. Svaki element u polju ima dužinu od 4 bajta. Šta više, svako polje ima zaglavlje od 5 bajtova. Stoga, polje koje je dimenzionisano sa DIM A (10,10) sadrži 100 elemenata (10×10) što je 405 bajtova za polje označeno sa A. Slično tome, polje B, dimenzionisano kao DIM B(20) sadržaće 20 elemenata, 80 bajtova plus zaglavlje i daće ukupno 85 bajtova. Komanda EOD, kao što će se kasnije objasniti, govori o tome kako se popunjava oblast za podatke.

Računar **PECOM-32** pruža mogućnost za rad sa nizovima. On obezbeđuje 26 promenljivih za nazive nizova (A$ do Z$). Svaki niz može da sadrži do 127 alfanumeričkih znakova. Takođe su uključena i polja niza koja omogućavaju programeru da ih pomoću nekih numeričkih argumenata indeksira u tabeli. Stoga se neki niz može pozvati sa A$ (N) gde je N bilo koji izraz (ili broj) maksimalne vrednosti 255. Ako se upotrebi broj veći od 255, može doći do pogrešnog rezultata. Za niz koji nema argumenat pretpostavlja se da ima argument 0. Npr.:

A$ je ekvivalentno sa A$(0).

Takođe je važno napomenuti da nije neophodno dimenzionisanje svakog niza. Memorija se automatski dodeljuje svaki put kada se niz generiše. Takođe treba napomenuti da generisanje niza C$(10) ne znači da automatski postoje C$(1) do C$(9). Postoji samo C$(10) i pokušaj da se očita C$(1) imaće za rezultat poruku o grešci koja kaže da C$(1) nije generisan.

5.7. **Naredbe za upravljanje tokom programa**

Ove naredbe se nazivaju upravljačke naredbe. One obavljaju uslovno i bezuslovno grananje. Upravljačke naredbe su: END, EXIT, FOR, GOSUB, GOTO, IF, NEXT, RETURN i WAIT.

END

Ova naredba (kraj programa) služi za zaustavljanje ili završetak programa i vraća BASIC3 na režim direktnog izvršenja. Naredba END može da se koristi bilo gde u programu u BASIC-u3 i to proizvoljan broj puta. Ona se, ako se želi, može obrisati ako je poslednja naredba u programu.

EXIT izraz

Ova naredba je ustvari bezuslovno grananje na broj reda definisan izrazom. Namenjena je za prevremeni izlaz iz petlje FOR/NEXT ili potprograma. Treba da se obrati pažnja na jednu meru predostrožnosti. Ako imamo slučaj poziva potprograma iz potprograma ili ako imamo slučaj korišćenja FOR/NEXT petlje u petlji, naredba EXIT služi da prenese upravljanje na broj reda koji je za jedan nivo niži od povezivanja. Na primer, ako postoji četiri FOR/NEXT petlji i želi se prevremeno grananje van četvrte FOR/NEXT petlje BASIC3 očekuje da to bude u trećoj FOR/NEXT petlji pošto se EXIT izvrši. EXIT može da bude i u nekom redu druge FOR/NEXT petlje, itd. Ova naredba se koristi umesto standardne GOTO naredbe i namena joj je da vrati sve modifikovane ukazatelje steka koji su proizašli iz FOR/NEXT ili poziva potprograma.

Sledi primer:

```
10 FOR I = 1 TO 10
20 FOR J = 1 TO 5
30 A = B+C
40 IF A = 15 THEN EXIT 60
50 NEXT J
60 PR A
70 NEXT I
```

FOR promenljiva = izraz1 TO izraz2 STEP izraz3

Ova naredba dodeljuje promenljivoj vrednost izraza1. Program nastavlja sa izvršavanjem sve dok ne naiđe na naredbu NEXT. U tom momentu vrednost promenljive se uvećava za iznos koji definiše izraz3 i koji može biti ili pozitivan ili negativan. Izraz2 – promenljiva se onda procenjuje i upoređuje sa znakom koraka. Pretpostavlja se da 0 ima pozitivan znak. Ako ne dođe do poklapanja znaka, izvršenje se preuzima pri prvoj naredbi posle poslednje naredbe FOR i nastavlja se kroz petlju. U bilo kom momentu za vreme petlje korisnik može da modifikuje vrednost naziva promenljive nekim raspoloživim sredstvom (npr. naredbom LET). Treba, takođe, obratiti pažnju da tip naziva promenljive postavlja tip izraza1, izraza2 i izraza3. Ako se STEP obriše, pretpostavlja se da je veličina koraka 1.

Primeri petlje FOR/NEXT:

```
10 FOR A = 1 TO 3:FOR B = 1 TO 6 STEP 2
20 PR A,B
30 NEXT B:NEXT A
40 PR "IZVRŠENO"
```

Ovaj primer ima za rezultat:

```
1       1
1       3
1       5
2       1
2       3
2       5
3       1
3       3
3       5
IZVRŠENO
```

Primer negativnog koraka:

```
10 FOR A = 10 TO 6 STEP-1
20 PR A
30 NEXT
40 PR "IZVRŠENO"
```

Ovaj primer ima za rezultat:

```
10
 9
 8
 7
 6
IZVRŠENO
```

Treći primer je:

```
10 FOR A = 1 TO 10
20 IF A = 5 LET A = 9
30 PR A
40 NEXT A
50 PR "IZVRŠENO"
```

Ovaj primer ima za rezultat:

```
 1
 2
 3
 4
 9
10
IZVRŠENO
```

GOSUB (izraz)

Ova naredba (poziv potprograma) identična je naredbi GOTO sa izuzetkom što program pamti gde je došlo do GOSUB. Kada BASIC naiđe na naredbu RETURN, on će se vratiti na naredbu koja dolazi posle poslednje izvršene naredbe GOSUB. Na ovaj način potprogrami se mogu povezivati onoliko duboko koliko to memorija dopušta (magacin raste sa porastom povezanih naredbi GOSUB). Primer naredbe GOSUB:

```
   5 A = 2000
  10 GOSUB 1000 : GOSUB 1000
  20 GOSUB A
  30 END
1000 PR "IZVRŠENO --"
1010 RETURN
2000 PR "OKONČANO" : GOSUB 3000 : GOTO 1010
3000 PR "KRAJ" : RETURN
```

Rezultat ovog primera je sledeći:

```
IZVRŠENO —
IZVRŠENO —
OKONČANO
KRAJ
```

GOTO

Ova naredba je bezuslovno grananje. Ona prenosi izvršenje na broj reda koji je specificiran izrazom. Ako broj reda ne postoji, generiše se poruka o grešci. Primeri su:

```
10 GOTO 50
10 GOTO A+B
10 GOTO 100 * (A-B)
```

IF oznaka niza <> izraz niza THEN naredba

IF izraz (relacioni operator) izraz THEN naredba

Ova naredba je uslovna naredba ali nije uslovno grananje kao kod ostalih BASIC-a. Ona testira uslov, i ako je uslov zadovoljen, naredba i grupa naredbi (odvojenih dvema tačkama) se izvode. Ako uslov nije zadovoljen, tada se izvođenje nastavlja sa sledećim redom. Kada se upoređuju izrazi nizova, jedini prihvatljivi relacioni operatori su znak jednako i nejednako. U slučaju aritmetičkih izraza, prihvatljivi operatori su:

```
= jednako
<> nejednako
> veće od
< manje od
>= veće ili jednako
<= manje ili jednako
```

Tip prvog izraza definiše tip drugog izraza. Primer uslovne naredbe:

```
10 IF A=B THEN PR A:WAIT(200):CLS:GOTO 200
20 PR B
```

U ovom slučaju, ako je vrednost za A jednako vrednosti za B, tada će se vrednost za A prikazati, doći će do kašnjenja, ekran će biti obrisan i izvršenje će se nastaviti u redu 200. Ako nije jednaka vrednosti za B, tada se vrednost za B prikaže a izvršenje se nastavlja od reda 20 pa nadalje.

Još jedna stvar koju treba napomenuti je da je ključna reč THEN opciona i ne mora da se upotrebi. Uobičajena greška koja treba da se izbegne je:

```
IF A>N THEN 200
```

Sve s desna od THEN, ako se javi, treba da bude izvršna naredba. Broj 200 nije izvršna naredba. Tačna verzija bi bila:

```
IF A>B THEN GOTO 200 ili
IF A>B GOTO 200
```

Primeri prihvatljivih IF naredbi su:

```
10 IF A(2) * B(1)=C * SIN(A) PR A : GOTO 50
10 IF A$(2)="LIST" LET B=3 : GOTO 100
10 IF B$(A+B)=MID$(A$,2,4) PR "U REDU"
10 IF MID$(A$(5),2,4)="EBCD" PR "JEDNAKO" : GOTO 500
10 IF A$=B$+C$ THEN GOSUB 200 : CLS : PR "IZVRŠENO"
10 IF A=INT (S/5) IF B>10 IF C<5 GOTO 500
```

Obratite pažnju na višestruke uslove u poslednjem primeru. Ako neki od uslova nisu zadovoljeni, izvršenje se nastavlja u sledećem redu. Ako se zadovolje svi uslovi, izvršenje će se konačno preneti na red 500 pomoću naredbe GOTO. Sledi takođe da je naredba GOTO mogla da bude svaka izvršna naredba.

NEXT

NEXT prosta promenljiva

Ova naredba zatvara petlju FOR/NEXT kao što je opisano u naredbi FOR. Ako se izostavi naziv promenljive, naredba NEXT vraća upravljanje na poslednju naredbu FOR i proces se nastavlja. Ako je dat naziv promenljive, BASIC3 će izvršiti proveru da ustanovi da li odgovara nazivu promenljive upotrebljene u poslednjoj naredbi FOR. Ako ne odgovara, izdaje se poruka o grešci.

RETURN

Ova naredba (povratak iz potprograma) označava kraj potprograma. Izvršenje se nastavlja sa naredbom iza poslednje izvršene GOSUB naredbe.

WAIT (izraz)

Ova naredba omogućuje unošenje kašnjenja u izvršenje programa. Dužina kašnjenja je direktno proporcionalna vrednosti izraza. 1 = 16000 kloka CPU. Pri brzini kloka od 2 MHz, kašnjenje za neki izraz od 1 je približno 8 mili sekundi (tj. $16000 \times 1/(2 \times 10^6)$). Primer primene naredbe WAIT:

```
10 INPUT A$(1)
20 PR "PORUKA JE";A$(1)
30 WAIT (500) : CLS : GOTO 10
```

Ovaj potprogram dozvoljava da se poruka posmatra izvesno vreme (500×8ms) pre nego što se ekran obriše i izvršenje nastavi.

5.8. **Komande**

Komande su sistemske direktive i normalno se izvršavaju u neposrednom režimu. BASIC-3 raspolaže sledećim komandama: CLD, CLS, CPOS, EDIT, EOD, EOP, FORMAT, LIST, NEW, PROB, RENUMBER, RUN, RUN+, i TRACE.

CLD

Kada se ova komanda izvrši (brisanje podataka) ona briše sve nizove i polja i resetuje ukazatelje svih nizova i polja na njihova početna stanja.

CLS

Briše ekran i postavlja kursor u početni položaj.

CPOS (izraz1, izraz2)

Ova komanda (pozicioniranje kursora) usmerava kursor na položaj na ekranu koji se specificira sa dva izraza u zagradi. Prvi izraz specificira položaj kolone (0 do 39) a drugi položaj vrste (0 do 23) kursora. CPOS može da se upotrebi zajedno sa PRINT za prikazivanje poruka i zaglavlja počev od specificiranog položaja na ekranu ili sa CHRGEN za kreiranje grafike u boji. Npr.:

```
10 CPOS (11, 12)
20 PRINT "OVO JE TEST"
RUN
```

prikazaće poruku OVO JE TEST u sredini ekrana.

CPOS (izraz1, izraz2, izraz3)

Ova komanda vrši pozicioniranje određenog znaka na ekranu pri čemu je:

Izraz1–redni broj kolone na ekranu i kreće se od 0 do 39
Izraz2–redni broj vrste na ekranu i kreće se od 0 do 23
Izraz3–decimalni kod za znak koji se pozicionira (uzima vrednost od 0 do 127).

EDIT izraz

Ova komanda otvara broj reda koji definiše izraz i dozvoljava korisniku da modifikuje red na nivou znaka. Dalji podaci o EDIT komandi dati su u odeljku 3.1.

EOD

EOP

Komanda EOD (kraj podataka) prikazuje heksadecimalnu adresu kraja memorijskog prostora za podatke (polja i podaci). Komanda EOP (kraj programa) slična je komandi EOD u tome što, kada se izvrši (normalno u direktnom režimu), automatski prikazuje heksadecimalnu adresu kraja memorijskog prostora za program. Korisnik je može uneti u bilo kom momentu za vreme izvršenja programa da bi video gde je u memoriji lociran kraj programa. Evo jednog primera:

```
EOP : EOD
```

može da ima za rezultat

```
& 2D05
& 2D55
```

& pokazuje da je broj koji sledi heksadecimalan.

Ovaj primer pokazuje da je kraj programa 2D05 a kraj podataka 2D55. EOD važi samo nakon što se generišu podaci izvođenjem programa (i pošto se izvrši DIM).

FORMAT N

Ova komanda specificira veličinu polja (tj. broj mesta za prikazane numeričke podatke). Svaka naredba PRINT koja dolazi posle komande FORMAT, bilo da prikazuje vrednost promenljive ili da prikazuje broj, imaće specificiranu veličinu polja. Npr.:

```
10 A = 12345,6
20 FORMAT 7
30 PR A
40 PR 67890,1
RUN
```

daće

```
12345,6
67890,1
```

Ako se red 20 zameni sa

```
20 FORMAT 6
RUN
```

rezultat je

```
******
******
```

Ako se red 40 zameni sa

```
40 PR–67890,1
```

rezultat je

```
******
–*****
```

Da rezimiramo: ako je veličina polja neki pozitivan broj koji je veći od N, prikazaće se N zvezdica. Ako je broj negativan, prikazaće se znak minus iza koga dolazi (N – 1) zvezdica. Opseg za n je od 1 do 15.

FORMAT 0 znači da se ograničenje usled FORMAT-a isključuje. Npr.:

```
10 FORMAT 5
20 PR 123456
30 FORMAT 0
40 PR 123456
RUN
```

daće

```
*****
123456
```

LIST
LIST izraz
LIST izraz, izraz

Ova komanda (listanje programa) bez izraza, izlistaće ceo program u korisničkom prostoru. Ako je dat jedan izraz, prikazaće se samo jedan red, čiji broj odgovara vrednosti izraza. Slično tome, ako su data dva izraza i ako su odvojeni zarezima, listanje će započeti i završiti se sa redovima koji odgovaraju vrednostima izraza. Ako je, u bilo kom momentu, neki od izraza jednak broju reda koji ne postoji, tada će se prikazati red čiji je broj najbliži datom. (Vidi odeljak 3.3).

NEW

Ova komanda u potpunosti inicijalizira BASIC računara **PECOM-32**, korisnički prostor i sav prostor za podatke tj. ona briše program u BASIC-u računara **PECOM-32** (koji je generisao korisnik) i inicijalizira ukazatelje svih polja i nizova.

PROB

Ova komanda prenosi upravljanje BASIC-a3 na mašinski jezik.

Nakon unošenja PROB sa tastature i pritiska na dirku RET, na ekranu se pojavljuje znak >. On vas upozorava da se sada nalazite u režimu rada sa mašinskim jezikom. Npr.

```
: PROB          .. pritisnite RET
>               .. režim rada na mašinskom jeziku
```

Sada se nalazite u novom režimu rada i na raspolaganju vam stoji 6 novih komandi: D, I, P, R, W i B.

D

Ova komanda služi za prikazivanje sadržaja memorije. Iza znaka D unosi se heksadecimalna adresa memorijske lokacije sa kojom se počinje prikazivanje sadržaja memorije. Nakon toga se unosi znak za blanko i heksadecimalan broj koji definiše koliko će se memorijskih lokacija prikazati. Npr. komanda

```
> D220 10
```

daje

```
220 FFFFFFFFFFFFFFFF  FFFFFFFFFFFFFFFF
```

što znači da je 16 memorijskih lokacija počev od adrese 0220 prazno.

Kao što se iz primera zaključuje memorijska adresa ne mora da se zada sa četiri heksadecimalne cifre. Adresa može da počne sa prvom cifrom najveće težine različitom od nule.

I

Ova komanda služi za unošenje programa na mašinskom jeziku kao i za njegove korekcije prilikom testiranja. Takođe ona služi za izmenu sadržaja bilo koje memorijske lokacije. Tako npr.

```
> I300 F800AA
```

znači da se u memorijske lokacije sa heksadecimalnom adresom 0300, 0301 i 0302 upisuje sadržaj F800AA. Tri bajta na mašinskom jeziku F800AA upisuju konstantu 00 u niži bajt registra A. (proučite Priručnik za mikroprocesor 1802).

P

Kada napišete program na mašinskom jeziku i želite da ga pozovete na izvršenje to ćete da uradite P komandom. Pre nego što izvršite komandu P, prekontrolišite još jednom svoj program koristeći komandu D.

R

Kao što se naredbom u BASIC-u PLOAD čita program sa kasete, tako se u režimu rada sa mašinskim jezikom komandom R čita program sa kasete koji je prethodno upisan na nju komandom W.

W

Ovo je komanda za upis programa na mašinskom jeziku na kasetu.

B

Ovom komandom vrši se prelaz iz režima rada na mašinskom jeziku na BASIC3. Posle unošenja B

```
> B
```

na ekranu se prikazuje poruka

```
READY
:
```

RENUMBER
RENUMBER N

Komanda RENUMBER omogućava korisniku da ponovo definiše brojeve redova u okviru programa. Ako posle naredbe RENUMBER ne dolazi ni jedan argument, smatra se da je početni broj reda 10 kao i razmak, tj. redovi u programu imaju sledeće vrednosti 10, 20, 30 itd. Po izvršenju komande RENUMBER izdaje se poruka koja pokazuje broj "izračunato grananje". "Izračunato grananje" je naredba koja zahteva od računara da se grana ili skoči na drugu naredbu čiji broj reda treba da se odredi. Npr., razmotrimo sledeći program:

```
10 FOR A = 1 TO 5
15 N(A) = 0
```

```
20 NEXT
25 GOTO 10
RENUMBER 20
```
daće poruku

```
0 COMP BR
```

a LIST će dati

```
20 FOR A = 1 TO 5
40 N(A) = 0
60 NEXT
80 GOTO 20
```

Vidite da su brojevi redova preuređeni. Takođe obratite pažnju na to da je naredba "25 GOTO 10" postala "80 GOTO 20" zbog toga što je red sa brojem 10 sada postao 20. "GOTO 10" nije "izračunato grananje" (odatle poruka "0 COMP BR") pošto je računar u stanju da odredi red na koji treba da se grana ili skoči.

Razmotrimo još jedan primer, pri čemu se "GOTO 10" zamenjuje sa "GOTO 5 * 2", (ili "GOTO B").

```
10 FOR A = 1 TO 5
15 N(A) = 0
20 NEXT
25 GOTO 5 * 2
RENUMBER 20
```

daće poruku

```
1 COMP BR
```

a LIST će dati

```
20 FOR A = 1 TO 5
40 N(A) = 0
60 NEXT
80 GOTO 5 * 2
```

Poruka o upozorenju da postoji jedno "izračunato grananje" prikazuje ustvari taj red, u ovom slučaju red 80 će prouzrokovati grešku zbog preuređenja. "GOTO 5 * 2" i "GOTO B" predstavljaju primere "izračunatog grananja".

RUN

Ova komanda postavlja BASIC3 u režim izvršenja programa. BASIC3 počinje traženje najmanjeg broja reda i počinje izvršenje svakog reda po numeričkom redosledu. Međutim, pre nego što počne izvršenje, komanda RUN briše svu oblast polja i niza da bi načinila prostor za nove podatke.

RUN izraz

Ova komanda započinje izvršenje programa sa redom čiji je broj određen izrazom. Ona je slična komandi RUN koju ne prati izraz u tome što postavlja BASIC3 u režim izvršenja. Ako broj reda koji specificira izraz ne postoji, generiše se kod greške. Pored toga, zbog toga što ova komanda RUN ne briše oblast podataka, korisnik može da izvrši program sa prethodno definisanim podacima (nizovima ili poljima).

RUN +

Ova komanda pretražuje korisnički program i zamenjuje interpretativne grane sa granama sa apsolutnim adresama i onda započinje izvršenje. Ovaj korak u velikoj meri povećava brzinu. Posle početnog ciklusa RUN +, RUN će omogućiti izvršenje u ovom bržem režimu.

Program koji se pozove sa RUN + ne treba da se čuva u ovm obliku zbog dodeljivanja apsolutnih adresa.

TRACE izraz

Ako je prateći izraz bilo šta osim nule, trasiranje se uključuje tj. svaki red koji izvršava BASIC3 preneće se na ekran:

```
TR (broj reda)
```

Ako se komanda TRACE izvršava sa pratećim izrazom čija je vrednost nula, trasiranje će se isključiti. Ova komanda se može staviti bilo gde u programu u BASIC-u3.

5.9. Naredbe za komentar i definisanje

Naredbe za komentar i definisanje dopuštaju programeru da ubaci komentar u program, da konfiguriše memoriju, tj. da definiše tačnu adresu programa, da dodeli memoriju za podatke i slično. Naredbe ove grupe su sledeće: DEFINT, DEFUS, DEG, DIM, FIXED, RAD i REM.

DEFINT
DEFINT naziv promenljive

Kada se ova naredba (definisanje celobrojnih promenljivih) izvrši bez naziva promenljive ona postavlja sve promenljive (A–Z) i sva poja (A(izraz)–Z(izraz)) kao promenljive sa pokretnim zarezom. Ako je naziv promenljive uključen u naredbu DEFINT, sve promenljive i polja od A do date promenljive (uključujući naziv promenljive) biće postavljene kao celobrojne. Npr.

DEFINT D

definiše nazive promenljivih i polja A, B, C, D kao celobrojne. Naredba NEW će uvek prebaciti sve promenljive u oblik sa pokretnim zarezom. Zbog toga, ako u programu treba da se upotrebe celobrojne promenljive, linija DEFINT treba da se javi na početku programa.

Treba, takođe, obratiti pažnju na to da ako se DEFINT upotrebi bez naziva promenljive, ona mora da se završi sa RETURN (RET), tj. dve tačke ne mogu da se upotrebe za povezivanje novih naredbi u istom redu.

DEFUS izraz

Ova naredba (definisanje starta korisničke oblasti) dopušta da se početak programske oblasti pomeri dalje u memoriji. Ona omogućava programeru da kreira "rupu" u memoriji u kojoj može da generiše programe na mašinskom jeziku. Izraz definiše gde programska oblast treba da počne. Kod BASIC-a3 korisnička oblast počinje sa adresom 0200. Izraz mora da počne od broja koji je veći od 0200. BASIC3 će zaokružiti izraz naniže da bi se dobila samo uvećanja pomeranja u parnim stranicama. Ako se načini pokušaj da se definiše korisnička oblast na nekoj adresi ispod 0200, BASIC3 će sam sebe uništiti. Kada se naredba DEFUS izvrši jedini način da se programska oblast vrati na 0200 jeste pomoću druge DEFUS naredbe na 0200. Linija uništava korisnički program koji je u memoriji.

Interesantna karakteristika pomeranja početka oblasti korisničkog programa leži u naredbi PSAVE. Ako je neki program generisan u pomerenoj lokaciji i sa njom je doveo u vezi neke programe na mašinskom jeziku, naredba PSAVE će upamtiti sve od 0200 do kraja programske oblasti. Programi na mašinskom jeziku biće uključeni kao i dati program u BASIC-u3. Naredba PLOAD će učitati sve gornje programe, uključujući, programe na mašinskom jeziku, i ponovo će definisati početak korisničke oblasti.

Primer za DEFUS naredbu dat je u daljem tekstu.

```
DEFUS 2D00
DEFUS 2DCC        pomera početak korisničke oblasti na 2DCC
DEFUS 0200        pomera početak programske oblasti u njegov početni položaj
```

Kada se koristi „rupa" koju je načinila naredba DEFUS, treba obratiti pažnju na to da se memorijski prostor 0200–0210 koristi u programima za smeštanje i treba da se izbegava kada se pišu programi na mašinskom jeziku.

DEG

Ova naredba postavlja jedinicu mere za ugaone funkcije na stepene. Početno stanje BASIC-a3 je merenje u radijanima. Komplement ovog stanja je RAD.

DIM naziv promenljive (izraz), naziv promenljive (izraz, izraz)...

Ova naredba (dimenzionisanje polja) služi da rezerviše memoriju za polja brojeva. Ova polja mogu biti jednodimenzionalna ili dvodimenzionalna. Naziv promenljive (A–Z) je naziv jednog polja. Izraz definiše koliko je polje veliko u jednoj ili obe dimenzije.

Vidi prethodno razmatranje o poljima i njihovim dimenzionim ograničenjima. Tip polja (pokretni zarez ili celobrojan) zavisi od toga da li je njegov prateći naziv definisan kao celobrojan ili sa pokretnim zarezom. Na primer, ako su sve promenljive A–Z sa pokretnim zarezom, tada će sva polja biti sa pokretnim zarezom (nezavisno od režima izraza koji definiše njegove granice). Ako su promenljive A, B i C definisane kao celobrojne, tada će polja A, B i C (ako se upotrebe) biti definisana kao celobrojna. Višestruka polja mogu se dimenzionisati u istoj DIM naredbi sve dok je svako polje odvojeno zarezom.

Ako se oblast memorije prekorači dok se dimenzioniše polje, rezultat će biti poruka o grešci. Pokušaj da se upotrebi element nekog polja koje prethodno nije dimenzionisano, imaće za rezultat poruku o grešci. Programer mora da upotrebi CLD naredbu da bi obrisao svu oblast za podatke.

Dajemo primer DIM naredbe:

DIM A (10,10), B(20)

Gornji primer ostavlja prostor za 100 brojeva (10×10), pri čemu nazivi idu od A(1,1) do A(10,10) i 20 brojeva sa nazivima koji od B(1) do B(20).

Naredba za dimenzionisanje može da se upotrebi, ako se želi, u direktnom režimu izvršenja. Svako polje koje program generiše ostaje nedirnuto nakon završetka izvođenja programa. Naredbom PRINT u direktnom režimu može da se sazna sadržaj nekog polja. Polje ostaje netaknuto sve dok se ne izvrši CLD ili dok se ne

obavi neka izmena programa. Ako se izvrši RUN izraz, polje nije obrisano i ponovno dimenzionisanje daće poruku o grešci. Iz tog razloga programer treba da započne izvršenje posle DIM naredbe ako želi da upotrebi prethodno generisane podatke. Podaci polja, kao i podaci niza, nalaze s neposredno iza korisničke oblasti. Svaka promena prvobitnog programa izmeniće korisničku oblast i na taj način uništiti podatke. Treba obratiti pažnju na to da neko polje može ponovo da se dimenzioniše a da se ne unište ostala polja ili nizovi.

FIXED izraz

Kada se ova naredba izvrši ona formatira prikazivanje svih brojeva i onih sa pokretnim zarezom a i celobrojnih. Vrednost izraza definiše koliko će se cifara prikazati s desna od decimalnog zareza. Prateće nule se dopisuju da bi se kompletirao broj. Ako je potrebno zadnja cifra broja će biti zaokružena. Vrednost izraza mora da bude broj između nula i šest. Ako se unese broj veći od 6, BASIC3 se prebacuje u normalan režim za prikazivanje brojeva. Npr,

```
FIXED 2
PR 123      daće za rezultat      123.00
PR 123.567        "               123,57
PR 6E7            "               0.60E8
FIXED 1
PR 123.50         "               123.5
```

RAD

Ova naredba postavlja jedinicu mere za ugaone funkcije na radijane. Ovo stanje je prethodno postavljeno stanje BASIC-a3. Komplement ove naredbe je DEG.

REM

Kada se REM postavi na početak nekog reda BASIC-a3, ceo red se prikazuje ali i ignoriše za vreme izvršenja programa. Njena svrha je da omogući programeru da komentariše program.

5.10. Naredbe za podatke programa

Naredbe za podatke programa omogućavaju programeru da ubaci liste podataka u program i daju mehanizam za čitanje podataka za vreme izvršenja programa. Naredbe za podatke pograma su: DATA, READ i RESTORE.

DATA izraz, niz funkcija

Naredba DATA sadrži podatke koje će koristiti naredba READ. Svaki elemenat u naredbi DATA mora da bude odvojen zarezom. Svaki niz znakova mora biti pod navodnicama. Naredba DATA može da se javi bilo gde u programu. Prihvatljive naredbe data su:

```
   1 DATA 2 * A,A$(1), "ZDRAVO",B
1000 DATA 1,2,3,4,
5000 DATA SIN(45), SIN(60), SIN(90)
```

READ naziv promenljive, naziv niza

Naredba READ se koristi za čitanje podataka iz naredbe DATA i dodeljivanje ovih podataka nekoj posebnoj promenljivoj. Svaki put kada naredba READ zahteva još podataka, interni ukazatelj BASIC-a3 pomera se do sledećeg podatka u naredbi DATA. Kada je naredba DATA bez podataka, BASIC3 će nastaviti sa pretraživanjem zbog neke druge naredbe DATA. Kada ne nađe ni jednu, vraća se poruka o grešci koja govori o nedostatku podataka. Važno je voditi računa o nazivima promenljivih ili nazivima niza u naredbi READ i odgovarajućih podataka u naredbi DATA. Oni moraju biti saglasni po pitanju oblika; nizovi idu sa nizovima, itd. Primer prihvatljivog para naredbi DATA/READ:

```
 5 DIM A(4)
10 DATA 10,20,30,40
20 FOR A=1 TO 4
30 READ A(A)
40 NEXT
50 READ A$(1),B,C
60 PR A$(1); B * C
70 DATA "B*C JEDNAKO JE",20,30
```

Rezultat gornjeg primera je da će se polje A puniti na sledeći način:

A(1)=10 A(2)=20 A(3)=30 A(4)=40

Rezultat reda 60 biće:

B * C JEDNAKO JE 600

RESTORE

Ova naredba resetuje prethodno pomenuti ukazatelj BASIC-a3 na početak prve naredbe DATA u programu, dopuštajući na taj način višestruku upotrebu DATA u jednom programu.

5.11. U/I naredbe

U/I (ulazno/izlazne naredbe) obezbeđuju vezu između programa i korisnika programa. U/I naredbe su INPUT i PRINT ili PR.

INPUT naziv promenljive, naziv promenljive

INPUT naziv promenljive niza

Kada BASIC3 naiđe na neku ulaznu naredbu, on se zaustavlja i izdaje na ekranu znak pitanja. Tada čeka na odgovor programera. Posle naredbe INPUT može da se pojavi niz naziva promenljivih. Svaki od naziva promenljive mora biti odvojen zarezom. Kao odgovor na upitnik (?) programer može da odgovori nekim izrazom ili grupom izraza odvojenih zarezima. Ako naredba INPUT ima tri naziva promenljive za sobom a programer odgovori sa dva ili više izraza, BASIC3 će nastaviti izvršenje. Kada naiđe na sledeću naredbu INPUT, uzeće poslednji neupotrebljeni izraz ne koristeći sledeći upitnik. Nastaviće tako sve dok se ne upotrebe svi neiskorišćeni uneti podaci. Zatim se, u tom momentu, izdaje drugi upitnik (?). Podaci koji se uzimaju dodeljuju se datom nazivu promenljive onim redom kojim se unose. Tip naziva promenljive definiše koji je tip unetog izraza.

Naredba INPUT ne sme da sadrži istovremeno i nazive promenljive niza i normalne nazive promenljive. Šta više, svaka naredba INPUT koja sadrži naziv promenljive niza može da sadrži jedan i samo jedan naziv. Zbog ovog ograničenja, kao odgovor na naredbu INPUT A$, programer može da odgovori nizom znakova bez znakova pitanja. Pritisak na RET označava kraj niza. Prihvatljive naredbe INPUT su:

```
10 INPUT A,B,C(1)
10 INPUT A(1), A(B)
10 INPUT A$
10 INPUT B$ (B)
```

Sledeće INPUT naredbe su pogrešne:

```
10 INPUT AB (nema ograničavača između promenljivih
10 INPUT A$, B$ (zabranjeni su višestruki nazivi niza)
```

INPUT „niz" naziv promenljive, naziv promenljive ...

Ova INPUT naredba je identična sa gornjom osim što se direktno posle INPUT naredbe javlja navodnik koji dopušta da se poruka štampa pre prikazivanja znaka pitanja. Pre prvog navodnika ili posle drugog ne koristi se ograničavač.

PRINT ili PR

PR se koristi kao skraćenica za PRINT. U listingu se ovi PR zamenjuju sa PRINT. Svrha ove naredbe je da U/I programima da prethodno definisani izraz ili funkciju niza. Svaka od stavki koja treba da se prikaže mora da se odvoji prihvatljivim ograničavačem, ili zarezom ili tačkom i zarezom. Svaki od ograničavača služi kao specijalna funkcija. Kada se upotrebi zarez, sledeća stavka koja treba da se prikaže postavlja se nakon razmaka od osam kolona. Kada se upotrebi tačka i zarez, ne ubacuju se razmaci i sledeća stavka se prikaže direktno posle poslednje. Tačka i zarez i zarez služe za zabranu RETURN pri kraju naredbe PRINT. Stoga će, naredna PRINT naredba na koju se naiđe u programu početi onde gde se prethodna zaustavila. Ako se PRINT naredba upotrebi sama, tada se ostavlja jedan red proreda. Postoje dve funkcije koje se mogu upotrebiti samo u PRINT naredbi: TAB (izraz) i CHR$ (izraz). Opis ove dve funkcije dat je u sledećem odeljku. Primeri prihvatljivih PR naredbi su:

```
PRINT 5
PRINT A
PR A
PR A,B, 1234,C(5)
PR "VREDNOST ZA A=";A
PR A$(1)+A$(2)
PR A$(2), A,B
PRINT MID$ (A$,1,2)
PRINT TAB (A+B);10;TAB(30);E
PR A
PR
```

Obratite pažnju na to da kada se više od jednog aritmetičkog izraza javi u naredbi PRINT, svaki izraz može da bude drugačijeg tipa.

5.12. **Naredba za poziv potprograma na mašinskom jeziku**

Ova naredba dopušta programeru da primeni kôd mašinskog jezika u svom programu i na taj način iskoristi posebne karakteristike procesora ili poboljša brzinu u primenama u realnom vremenu. Naredba za poziv potprograma u mašinskom jeziku je CALL.

CALL (izraz 1)
CALL (izraz 1, izraz 2)
CALL (izraz 1, izraz 2, izraz 3)

Ova naredba obezbeđuje vezu između BASIC-a3 i programiranja na mašinskom jeziku. Ona služi kao poziv potprograma na mašinskom jeziku i prenosi izvršenje na taj potprogram. Adresa potprograma određuje se pomoću izraza 1. Program na mašinskom jeziku treba da bude napisan imajući na umu sledeća pravila:

1. Programski brojač nakon ulaska u potprogram je R3.
2. Povratak na BASIC3 vrši se pomoću naredbe D5(SEP R5).
3. Programi na mašinskom jeziku slobodno koriste R8, RA, RC, RD i RE. Ako treba da se upotrebe neki drugi registri, sadržaj tih registra treba prvo da se smesti u magacin i obnovi pre vraćanja na BASIC3.

NAPOMENA: Standardna tehnika pozivanja i vraćanja (SCRT) ne koristi se zbog toga što se pomoću nje uništava viši deo registra F(RF.1)

4. BASIC3 je ustanovio konvenciju pozivanja i vraćanja i nije potrebna dalja inicijalizacija od strane potprograma na mašinskom jeziku.
5. Magacin stoji na raspolaganju za upotrebu (na njega ukazuje R2) sve dok se ne vrati na početno stanje.

Svaki od izraza (izraz1, izraz2, izraz3) može se izraziti ili celobrojno ili sa pokretnim zarezom. BASIC3 će ih automatski pretvoriti u celobrojne. Vrednost za izraz 2, ako se upotrebi prenosi se na potprogram na mašinskom jeziku kao 16-tobitni binarni ceo broj u R8. Drugi deo podatka može se, takođe, preneti potprogramu na mašinskom jeziku u registru RA. Sadržaj registra RA će ustvari biti vrednost za izraz 3.

5.13 **Ulazno-izlazne funkcije**

U/I funkcije ove grupe su KEY, MEM, PEEK i TAB.

KEY

Ova funkcija daje decimalni ekvivalent koda pritisnute dirke na tastaturi. Ukoliko nijedna dirka nije pritisnuta daje vrednost 0.

MEM

MEM funkcija dopušta korisniku da utvrdi koliko je još ostalo memorijskog prostora. Kada se izvrši, MEM daje decimalni broj koji predstavlja broj preostalih memorijskih bajtova između kraja podataka (nizovi i polja) i kraja normalnog magacina. Stvarni broj koji se vraća smanjuje se za 256 da bi se omogućilo povećanje prostora za magacin za vreme izvršenja programa. Treba obratiti pažnju na to da ako MEM da negativan broj, to znači da je kraj podataka u okviru 256 bajtova i treba voditi računa da se program dalje ne povećava.

Nikakva indikacija za prerastanje magacina u program za vreme izvršenja programa ne postoji. Da bi se na ekranu prikazala vrednost preostalog memorijskog prostora, korisnik treba da unese

```
PR MEM
```

PEEK (izraz)

Ova funkcija daje decimalni ekvivalent sadržaja memorijske lokacije čija je adresa određena izrazom. Decimalni rezultat prikazuje se u pokretnom zarezu.

TAB (izraz)

Ova funkcija nije funkcija u istom smislu kao prethodne funkcije zbog toga što ne daje nikakvu vrednost. Ona se koristi i prepoznaje samo u naredbi PRINT i postavlja kursor u horizontalni položaj određen vredošću izraza. Novi položaj kursora određuje se u odnosu na kolonu 1 a ne u odnosu na prethodni položaj kursora. Prikazivanje se nastavlja od te kolone pa nadalje. Na primer:

```
PR TAB(10);1
```

prikazuje 1 u koloni 10

PR TAB(10);1;TAB(20);2
prikazuje 1 u koloni 10 a 2 u koloni 20.
PR TAB(A);"*"
prikazuje zvezdicu u koloni A.

5.14. **Funkcija za prelaz na mašinski jezik**

Funkcija ove grupe je USR

USR (izraz)
USR (izraz1, izraz2)
USR (izraz1, izraz2, izraz3)

Ova funkcija ima dejstvo kao naredba CALL koja je opisana u prethodnom odeljku, ali s tom razlikom da je USR funkcija koja će se upotrebiti kao deo nekog izraza. Kada se naiđe na USR, vrši se poziv potprograma na mašinskom jeziku čija je adresa definisana izrazom. Podaci se mogu preneti na potprogram na potpuno isti način kao naredbom CALL. Kod funkcije USR, kada se naiđe na D5 u programu na mašinskom jeziku, BASIC3 će dati 32-bitni ceo broj kao vrednost za funkciju USR. Ovaj 32-bitni broj obrazuje sadržaj registra R8 i RA. Registar R8 daje 16 bitova nižeg reda a RA daje 16 bitova višeg reda. Na primer:

PR 2 * USR (&3C1E, #2E)+A/D

Molimo vas obratite pažnju na to da USR i CALL koriste registre mikroprocesora 1802. Opis unutrašnje strukture mikroprocesora 1802 možete naći u nekom priručniku za programiranje mikroprocesora 1802.

6. PRIMERI PROGRAMIRANJA

U ovom poglavlju dat je skup programa. Osim praktičnosti samih programa, oni su dati kao primeri pomoću kojih se ilustruje primena izvesnih naredbi u BASIC-u. Pored listinga i prikazivanja na ekranu, svaki program biće prokomentarisan po pitanju upotrebljenih tehnika programiranja.

6.1. Prikazivanje primene naredbi COLOR, SCR i TONE

6.1.1. Listing programa

```
 10 REM        OVAJ PROGRAM JE ILUSTRATIVAN
 20 REM        I POKAZUJE KAKO SE KORIŠĆENJEM NAREDBI
 30 REM        PECOM-32 MOŽE MENJATI BOJA ZNAKOVA, EKRANA
 40 REM        I KAKO SE GENERIŠE TON
 70 REM
 75 CPOS(0,0):CLS
 80 SCR7:CPOS(7,1):PRINT"JA SAM PAMETAN RAČUNAR PECOM-32
 90 CPOS(4,3):PRINT"ŽELIM DA VAM POKAŽEM NEKOLIKO STVARI:"
110 CPOS(4,7):PRINT"1.KAKO DA MENJATE BOJE PRIKAZANIH ZNA-"
120 CPOS(4,8):PRINT" KOVA"
140 CPOS(4,10):PRINT"2.KAKO DA MENJATE BOJU EKRANA"
160 CPOS(4,13):PRINT"3.KAKO DA GENERIŠETE TON".
170 CPOS(2,15):PRINT"ŠTA BISTE OD GORE NAVEDENOG ŽELELI DA SAZNATE?"
173 CPOS(2,17):INPUT"ODGOVORI SA 1,2 ili 3. ODGOVORI SA 0 ZA KRAJ" K
175 PRINT
177 IF K=0 THEN GOTO 710
180 GOTO K * 200
200 SCR1:CPOS(0,0):CLS
204 CPOS(8,3):PRINT"DA BISTE MENJALI BOJU ZNAKOVA"
205 CPOS(11,4):PRINT"KORISTITE NAREDBU COLOR (X,Y,Z),"
210 CPOS(11,5):PRINT"GDE X I Y ODREĐUJU DECIMALAN"
215 CPOS(11,6):PRINT"KOD ZNAKA OD KOGA SE POČINJE"
220 CPOS(8,8):PRINT"BOJENJE I DECIMALAN KOD ZNAKA"
225 CPOS(9,9):PRINT"SA KOJIM SE BOJENJE ZAVRŠAVA."
230 CPOS(8,11):PRINT"Z PREDSTAVLJA KOD ZA BOJU I MOŽE"
235 CPOS(9,12):PRINT"DA UZIMA VREDNOSTI OD 0 DO 7."
240 CPOS(8,14):PRINT"UNESITE"
245 CPOS(8,16):PRINT" COLOR (65,87,N)"
247 CPOS(8,18):PRINT"(GDE N MOŽE DA BUDE OD 1 DO 6)."
250 CPOS(8,20):PRINT"POSLE ZNAKA ? PRITISNITE RET"
260 INPUT A$
265 B$=MID$(A$,13,1)
270 C=FVAL(B$)
275 COLOR(65, 87, C)
280 PRINT"PRITISNITE BILO KOJU DIRKU"
285 PRINT"ZA POVRATAK NA OSNOVNU BOJU"
290 INPUT A$
295 COLOR(65,87,7):CPOS(0,0):CLS:GOTO80
400 SCR 7:CPOS(0,0):CLS
404 CPOS(4,6):PRINT"ZA PROMENU BOJE EKRANA KORISTI SE"
405 CPOS(4,8):PRINT"NAREDBA OBLIKA SCRM."
415 CPOS(4,13):PRINT"NA PRIMER, UNESITE"
420 CPOS(4,16):PRINT"    SCRM"
422 CPOS(4,19):PRINT"(GDE JE M CEO BROJ IZMEĐU 1 I 6)"
425 CPOS(4,21):PRINT"NAKON POJAVE ZNAKA?PRITISNITE RET"
430 INPUT A$
435 B$=MID$(A$,4,1)
440 C=FVAL(B$)
445 SCRC
450 PRINT"PRITISNITE BILO KOJU DIRKU A ZATIM RET"
455 PRINT" ZA POVRATAK NA POČETNU BOJU EKRANA"
460 INPUT A$
470 SCR1
500 CPOS(0,0):CLS:GOTO 80
600 SCR2:CPOS(0,0):CLS
604 CPOS(4,3):PRINT"MUZIČKI TONOVI GENERIŠU SE"
605 CPOS(4,5):PRINT"POMOĆU NAREDBE TONE(F,R,A)"
610 CPOS(4,8):PRINT" F=0 DO 127 ZA UČESTANOST"
```

```
615 CPOS(4,10):PRINT"R=0 DO 7 ZA OKTAVU"
620 CPOS(4,12):PRINT"Z=0 DO 15 ZA AMPLITUDU"
625 CPOS(4,15):PRINT"PROBAJTE, NPR., DA UNESETE"
630 CPOS(4,18):PRINT"    TONE(F,R,A)"
635 CPOS(4,21):PRINT"POSLE ZNAKA ?. ZATIM, PRITISNITE RET"
640 INPUT A$
645 F$=MID$(A$,6,1)
650 R$=MID$(A$,8,1)
655 A$=MID$(A$,10,1)
660 F=FVAL(F$)
665 R=FVAL(R$)
670 A=FVAL(A$)
675 TONE(F,R,A)
680 PRINT"PRITISNITE BILO KOJU DIRKU A ZATIM RET"
685 PRINT"DA BISTE UKINULI GENERISAN TON"
690 INPUT A$
700 TONE(0,0,0)
705 CPOS(0,0):CLS:GOTO 80
710 SCR1:END
```

6.1.2. Komentarisanje programa

(1)... Redovi od 10 do 50	Obratite pažnju na korišćenje REM za opis cilja programa
(2)... Red 75	Briše ekran za novo prikazivanje
(3)... Redovi od 80 do 160	Ilustracija korišćenja CPOS za pozicioniranje poruke na ekranu
(4)... Red 170	INPUT naredba daje uputstvo korisniku kako da „razgovara" sa računarom. U ovom slučaju od njega se zahteva da izabere opciju unošenjem 1,2,3 ili 0.
(5)... Red 260	Računar prihvata niz A$ oblika COLOR (65,87,N) koji korisnik unese. Pretpostavimo da korisnik unese COLOR (65,87,6).
(6)... Redovi 265 do 276	Računaru treba naložiti da izvrši naredbu COLOR (65,87,N) kako zahteva korisnik. Ova tri reda to zaista i čine korišćenjem funkcija MID$ i FVAL. U redu 265 izdvaja se znak 13 iz niza COLOR (65,87,6). Tako je B$ jednako ASCII kodu znaka 6. U redu 270, FVAL izračunava znak 6 i tu vrednost dodeljuje znaku C. U ovom slučaju znak C=6, i COLOR(65,87,C) vrši promenu boje prema zahtevu. Ista tehnika korišćena je u redovima 435 do 445, i redovima 645 do 675.

6.2. **Korišćenje PECOM-a 32 za rešavanje matematičkih izraza**

6.2.1. Listing programa

```
 10 REM      OVAJ PROGRAM POKAZUJE
 20 REM      KAKO PECOM-32 MOŽE DA
 30 REM      SE KORISTI KAO KALKULATOR
 40 REM      ZA IZRAČUNAVANJE SLOŽENIH
 50 REM      MATEMATIČKIH IZRAZA
 60 CPOS(0,0)
 70 CPOS(5,4): PRINT"ŽELIM DA VAM PRIKAŽEM"
 80 CPOS(6,5): PRINT"SVOJE MATEMATIČKE SPOSOBNOSTI."
 90 CPOS(5,7):PRINT"POČNIMO SA SABIRANJEM."
100 CPOS(5,8):PRINT"OPERATOR JE+."
130 CPOS(5,10):PRINT"MOLIMO VAS UNESITE"
140 CPOS(5,12):PRINT"    3+4"
150 CPOS(5,14):PRINT"POSLE ZNAKA ?."
160 CPOS(5,17):PRINT"ZATIM,PRITISNITE RET"
165 CPOS(5,19):PRINT"ČEKAJTE NA ODGOVOR"
167 CPOS(6,20):PRINT"    U SLEDEĆEM REDU."
170 INPUT A$
180 PRINT FVAL(A$)
182 INPUT"PRITISNI RET DIRKU ZA NASTAVAK" K$
184 CPOS(0,0):CLS
190 CPOS(4,8):PRINT:"SADA, POKUŠAJMO NEŠTO"
```

```
200 CPOS(6,9):PRINT"KOMPLIKOVANIJE."
210 CPOS(4,11):PRINT"POSLE ZNAKA ? UNESITE"
220 CPOS(4,13):PRINT"    12+14+61" .
221 CPOS(4,15):PRINT"ZATIM, PRITISNITE RET ZA ODGOVOR"
224 PRINT
226 PRINT
230 INPUT A$
240 PRINT FVAL (A$)
242 INPUT"PRITISNITE BILO KOJU DIRKU"
245 INPUT"  I TASTER RET ZA NASTAVAK" A$
250 CPOS(0,0)
260 CPOS(4,8):PRINT"OPERATOR JE –."
270 CPOS(4,11):PRINT"UNESITE"
280 CPOS(4,13):PRINT"    9–5"
290 CPOS(4,15):PRINT"POSLE ZNAKA?.PRITISNITE RET"
300 PRINT A$
310 PRINT FVAL(A$)
315 INPUT"PRITISNITE BILO KOJU DIRKU"
317 INPUT"  I RET DIRKU ZA PRODUŽETAK" K$
320 CPOS(0,0):CLS:CPOS(8,7):PRINT"SADA,POKUŠAJTE SLEDEĆE"
330 CPOS(8,9):PRINT"UNESITE POSLE ZNAKA?"
340 CPOS(8,12):PRINT"    5+6–7"
345 CPOS(8,15):PRINT" I PRITISNITE RET ZA ODGOVOR."
350 INPUT A$
360 PRINT FVAL(A$)
365 INPUT"PRITISNITE BILO KOJU DIRKU"
367 INPUT" I RET ZA NASTAVAK"K$
370 CPOS(0,0):CLS:CPOS(5,5):PRINT"SADA, POKUŠAJMO MNOŽENJE"
380 CPOS(5,7):PRINT"OPERATOR JE *"
390 CPOS(5,9):PRINT"UNESITE"
400 CPOS(5,12):PRINT"9 * 7"
410 CPOS(5,15):PRINT"POSLE ZNAKA ?. UNESITE RET"
420 INPUT A$
430 PRINT FVAL(A$)
435 INPUT"PRITISNETE BILO KOJU DIRKU I RET"K$
440 CPOS(0,0):CLS:CPOS(5,5):PRINT"SADA POKUŠAJMO DELJENJE."
450 CPOS(5,7):PRINT"OPERATOR JE /"
460 CPOS(5,9):PRINT"UNESITE"
465 CPOS(5,12):PRINT"   63/70"
470 CPOS(5,15):PRINT"POSLE ZNAKA ?. PRITISNITE RET."
480 CPOS(5,17):PRINT"   (63/7 ZNAČI 63 PODELJENO SA 7)"
490 INPUT A$
500 PRINT FVAL(A$)
520 CPOS(0,20):INPUT"PRITISNITE BILO KOJU DIRKU I RET"K$
522 CPOS(0,0):CLS:CPOS(5,2):PRINT"POKUŠAJTE NEŠTO KOMPLIKOVANIJE"
530 CPOS(5,4):PRINT"UNESITE IZRAZ"
540 CPOS(5,7):PRINT"  (5+6) * (9–7)"
550 CPOS(5,10):POSLE ZNAKA ?. PRITISNITE RET"
560 INPUT A$
570 PRINT FVAL (A$)
575 INPUT"PRITISNITE BILO KOJU DIRKU I RET" K$
580 CPOS(0,0):CLS:CPOS(4,8):PRINT"SADA, POKUŠAJTE SLEDEĆE"
590 CPOS(4,11):PRINT"  (7+4) * (19–14)/(8–3)"
600 CPOS(4,13):PRINT" POSLE ZNAKA ?. PRITISNITE RET"
610 INPUT A$
620 PRINT FVAL(A$)
625 INPUT"PRITISNITE BILO KOJU DIRKU I RET" K$
630 CPOS(0,0):CLS:CPOS(5,5):PRINT"SADA, POKUŠAJTE SAMI."
640 CPOS(5,7):PRINT"UNESITE NEKI SVOJ IZRAZ"
650 CPOS(5,10):PRINT"POSLE ZNAKA ?"
660 CPOS(5,12):PRINT" I PRITISNITE RET"
670 INPUT A$
680 PRINT FVAL(A$)
690 INPUT"DRUGI IZRAZ, ODGOVORITE SA DA/NE" Q$
692 IF Q$="DA" THEN GOTO 630
694 IFQ$="NE" THEN GOTO 720
```

```
720 CPOS(0,0):CLS:CPOS(5,5):PRINT"POŠTO STE PREŠLI"
730 CPOS(5,7):PRINT"OVAJ PRIRUČNIK, MOŽETE"
735 CPOS(5,9):PRINT"DA KORISTITE PECOM"
740 CPOS(5,12):PRINT"ZA IZRAČUNAVANJE"
750 CPOS(5,14):PRINT"BILO KOG IZRAZA PO SOPSTVENOM"
760 CPOS(5,16):PRINT"IZBORU.UNESITE SAMO PR ILI PRINT."
770 CPOS(5,19):PRINT"ZATIM UNESITE IZRAZ I PRITISNITE RET."
780 CPOS(5,21):PRINT"NPR. PR 6 * 5/9"
785 CPOS(1,23):INPUT"PRITISNI BILO KOJI TASTER I RET" K$
790 CPOS(0,0):CLS:CPOS(5,5):PRINT"PECOM MOŽE DA IZVRŠAVA:"
800 CPOS(5,7):PRINT"TRIGONOMETRIJSKE,"
810 CPOS(5,9):PRINT"LOGARITAMSKE,"
820 CPOS(5,11):PRINT"EKSPONENCIJALNE"
830 CPOS(5,13):PRINT"I MNOGE DRUGE FUNKCIJE."
840 CPOS(5,15):PRINT"MOLIMO VAS DA PRILIKOM"
850 CPOS(5,17):PRINT"UPOTREBE PECOM-a KORISTITE"
860 CPOS(5,21):PRINT"OVAJ PRIRUČNIK."
870 CPOS(5,23):PRINT"DOVIĐENJA. LEPO SE ZABAVLJAJTE."
880 END
```

6.2.2. Komentarisanje programa

U redu 170, od korisnika se zahteva da unese aritmetički izraz 3+4. To znači da je niz A$ jednak 3+4. Naredba PRINT FVAL (A$) u redu 180 izračunava aritmetički izraz 3+4. Na ovaj način mogu da se izračunavaju mnogo kompleksniji aritmetički izrazi. Ista tehnika se koristi u redovima 230, 240, 300 i 310. Naredba INPUT u redu 690 omogućava korisniku da izabere odgovor DA ili NE. Različite akcije se preduzimaju u redovima 692 i 694.

A.1– Poruke za greške i njihovo značenje

00 – poruka o prekidu
01 – sintaksna greška u ASC ili LEN funkciji
02 – Polje van opsega ili nedimenzionisano polje
03 – DIM greška; sintaksna greška ili prekoračenje memorije
04 – DEFINT ima pogrešan završetak reda
05 – nepostojanje zagrada u funkciji sa argumentom
06 – argument van opsega
07 – nedozvoljeno izračunavanje zbog različitih režima
08 – greška zbog delenja nulom ili log negativnog broja
09 – funkcija koja se ne može izvršiti
10 – EXIT naredba se koristi u GOSUB ili FOR/NEXT
11 – prekoračenje steka u FOR/NEXT, ili nemogućnost da se FOR izvrši direktno
12 – sintaksna greška u FOR
13 – prekoračenje steka u GOSUB
14 – neprihvatljiv znak u heksadecimalnom polju
15 – suviše veliki broj u pokretnom zarezu za prebacivanje u ceo broj
16 – nedozvoljen operator u uslovnoj naredbi
17 – INPUT ili FVAL ne mogu se izvršiti u direktnom režimu
18 – u READ naredbi mora da postoji naziv promenljive ili niza
19 – sintaksna greška u READ ili INPUT naredbi
20 – sintaksna greška u LEN funkciji
21 – sintaksna greška u LEN naredbi
22 – nepostojanje navodnika
23 – sintaksna greška u LIST
24 – nepostojeća reč u BASIC 3 biblioteci
25 – sintaksna greška u MID$ funkciji
26 – neprihvatljiv naziv promenljive u NEXT naredbi
27 – očekuje se broj ili slovo
28 – nedostaje aritmetička zagrada
29 – pogrešan broj argumenata u POKE naredbi
30 – nedozvoljeni zadanji znak u PRINT naredbi
31 – sintaksna greška u DATA naredbi
32 – nema više podataka
33 – nepostojeći niz u INPUT naredbi
34 – nema znaka = u LET naredbi za nizove
35 – nema zagrada u polju niza pri INPUT
36 – suviše argumenata u USR ili CALL
37 – sintaksna greška u CHR$ funkciji
38 – neprihvatljiv znak u binarnom polju (1 ili 0)
39 – prekoračenje linijskog bafera
42 – nedozvoljeni završetak reda ili nepostojeća naredba
43 – prekoračenje steka u pokušaju poziva programa za prekid
44 – unet broj ima suviše cifara (10)
45 – neprihvatljiv znak u polju brojeva
46 – nepostojeći broj reda
47 – neprihvatljiv operator u IF naredbi za nizove
48 – prekoračenje memorije pri pokušaju unošenja niza
49 – pogrešan broj argumenata u MOD funkciji
50 – suviše dug program
51 – argument van opsega
52 – pogrešan broj argumenata
53 – pogrešan broj argumenata
54 – nedefinisana promenljiva niza
63 – nedostatak memorije za RUNUMBER tabelu
64 – nepronađeni broj za RENUMBER
65 – nedefinisana promenljiva
66 – pokušaj generisanja niza većeg od 127 znakova
68 – broj argumenata u SCR mora da bude 1
72 – broj argumenata u TONE mora biti 3

A.2 – ASCII TABELA ZA UGRAĐENE ZNAKE I SIMBOLE

Decimalni kod	Heksa kod	Znak/ simbol	Pritisnuta dirka
0	00		NUL
1	01	C_A	CTLA
2	02	▭	CTLB
3	03	C_C	CTLC
4	04		
5	05	■	CTLE
6	06	–	CTLF
7	07	I	CTLG
8	08		CTLH
9	09	↘	CTLI
10	0A		LF
11	0B	⋰	CTLK
12	0C	I	CTLL
13	0D		CR
14	0E	–	CTLN
15	0F	→	CTLO
16	10	⌜	CTLP
17	11	¬	CTLQ
18	12	⟋	CTLR
19	13	C_S	CTLS
20	14	⌞	CTLT
21	15	●	CTLU
22	16	I	CTLV
23	17	⟍	CTLW
24	18	+	CTLX
25	19	←	CTLY
26	1A	⌟	CTLZ
27	1B	E_S	CTL ↓
28	1C	–	CTL←
29	1D	⤧	CTL→
30	1E	⁘	CTL ↑
31	1F		
32	20		BLANKO
33	21		SHIFT + 1
34	22		SHIFT + 2
35	23	#	SHIFT + 3
36	24	$	SHIFT + 4
37	25	%	SHIFT + 5
38	26	&	SHIFT + 6
39	27	'	SHIFT + 7
40	28	(	SHIFT + 8
41	29	)	SHIFT + 9
42	2A	*	SHIFT + :
43	2B	+	SHIFT + ;
44	2C	/	
45	2D	–	SHIFT + =
46	2E	.	
47	2F	/	
48	30	0	
49	31	1	
50	32	2	
51	33	3	
52	34	4	
53	35	5	
54	36	6	
55	37	7	
56	38	8	
57	39	9	
58	3A	:	
59	3B	;	
60	3C	<	SHIFT + ,
61	3D	=	
62	3E	>	SHIFT + .
63	3F	?	SHIFT + /
64	40	ç	
65	41	A	
66	42	B	
67	43	C	
68	44	D	
69	45	E	
70	46	F	
71	47	G	
72	48	H	
73	49	I	
74	4A	J	
75	4B	K	
76	4C	L	
77	4D	M	
78	4E	N	
79	4F	O	
80	50	P	
81	51	Q	
82	52	R	
83	53	S	
84	54	T	
85	55	U	
86	56	V	
87	57	W	
88	58	X	
89	59	Y	
90	5A	Z	
91	5B	↓	
92	5C	←	
93	5D	→	
94	5E	↑	
95	5F	–	SHIFT + ↑
96	60		
97	61	a	CAPS + A
98	62	b	CAPS + B
99	63	c	CAPS + C
100	64	d	CAPS + D
101	65	e	CAPS + E
102	66	f	CAPS + F
103	67	g	CAPS + G
104	68	h	CAPS + H
105	69	i	
106	6A	j	
107	6B	k	
108	6C	l	
109	6D	m	
110	6E	n	
111	6F	o	
112	70	p	
113	71	q	
114	72	r	
115	73	s	
116	74	t	
117	75	u	
118	76	v	
119	77	w	
120	78	x	
121	79	y	
122	7A	z	
123	7B	π	SHIFT + ↓
124	7C	✚	SHIFT + ←
125	7D	Σ	SHIFT + →
126	7E	–	SHIFT + ↑
127	7F		

A.3 UPUTSTVO ZA PRAVILNO KORIŠĆENJE KASETE I KASETOFONA

1. Koristite kvalitetne trake
2. Koristite najkraće trake
3. Glave i valjci kasetofona treba da budu čisti
4. Glave kasetofona treba da budu poravnjane pri zameni trake
5. Izbegavajte upisivanje na početak trake
6. Proverite da li je kaseta ispravno postavljena u kasetofon
7. Ako imate problema sa trakom kasete pokušajte sa drugom
8. Pokušajte da pronađete najbolju vrednost za učitavanje programa sa kasete
9. Prljav mehanizam kasetofona može da uništi traku
10. Proverite da li je kasetofon dobro povezan sa računarom
11. Premotajte traku pre nego što je izvadite iz kasetofona
12. Čuvajte kasete u originalnim kutijama
13. Ne izlažite trake toploti i dejstvu magnetnih polja
14. Pre upisivanja premotajte traku
15. Ima kasetofona koji ne prihvataju izvesne kasete itd. Ako to ustanovite, pokušajte sa drugim
16. Pre snimanja proverite da li je utikač za MIC na kasetofonu dobro povezan
17. Ako treba da snimate bez utikača za EAR, pogledajte još jednom priručnik za kasetofon.

ABC
ASC
ATN

CALL
CHR$
CHRGEN
CLD
CLS
COLOR
COS
CPOS

DATA
DEFINT
DEFUS
DEG
DIM
DLOAD
DSAVE

EDIT
END
EOD
EOP
EXIT
EXP

FIXED
FNUM
FOR
FORMAT
FVAL

GOSUB
GOTO

IF
INPUT
INT
INUM

KEY

LEN
LET
LIST
LOG

MEM
MID$
MOD

NEW
NEXT

PEEK
PI
PLOAD
POKE
PRINT (PR)
PROB
PSAVE

RAD
READ
REM
RENUMBER
RESTORE
RETURN
RND
RUN
RUN+

SGN
SIN
SQR
SCR

TAB
TONE
TRACE

USR

WAIT

A.4.1 – Skup naredbi BASIC 3 programskog jezika

CALL	– Poziv potprograma na mašinskom jeziku
CLD	– Brisanje podataka
CLS	– Brisanje ekrana
COLOR	– Bojenje ekrana
CHRGEN	– Generisanje znaka
CPOS	– Pozicioniranje kursora
DATA	– Programski podaci
DEFINT	– Definisanje promenljive kao celi broj
DEFUS	– Pomeranje početka programa
DEG	– Postavljanje mere ugla na stepen
DIM	– Dimenzionisanje polja
DLOAD	– Punjenje podataka sa trake
DSAVE	– Smeštanje podataka na traku
EDIT	– Editovanje reda
END	– Ograničavač programa
EOD	– Kraj podatka
EOP	– Kraj programa
EXIT	– Izlaz iz petlje
FIXED	– Format za štampanje brojeva
FOR	– FOR petlja
FORMAT	– Postavljanje polja numeričkog podatka
GOSUB	– Poziv potprograma
GOTO	– Bezuslovni skok
IF	– Uslovna naredba
INPUT	– Ulaz sa tastature
LET	– Naredba dodeljivanja
LIST	– Listanje programa
NEW	– Brisanje programa
NEXT	– Ograničavač FOR petlje
PLOAD	– Punjenje programa sa trake
POKE	– Direktno adresiranje memorije
PRINT (PR)	– Prikazivanje podataka na ekranu
PROB	– Prelazak iz BASIC-a 3 na mašinski jezik
PSAVE	– Smeštanje programa na traku
RAD	– Postavljanje mere ugla na redijane
READ	– Unošenje programskih podataka
REM	– Komentar
RENUMBER	– Ponovno zadavanje brojeva za redove u programu
RESTORE	– Resetovanje ukazatelja za podatke u programu
RETURN	– Povratak iz potprograma
RUN	– Startovanje programa
RUN +	– Startovanje programa
SCR	– Bojenje ekrana
TONE	– Generisanje tona
TRACE	– Prikazivanje brojeva redova programa onim redom kojim se izvršavaju
WAIT	– Kašnjenje

Dodatne editorske funkcije

CTL –S	Završetak editovanja
CTL –A	Završetak editovanja bez izmene
CTL –C	Brisanje zadnje linije
CTL –D	Brisanje celoga ekrana
CTL –H	Brisanje zadnjeg znaka (back–space)
I	Umetanje teksta (ili naredbi) na poziciju ukazatelja kursora
C	Izmena znakova na poziciji ukazatelja kursora
D	Brisanje znakova na poziciji ukazatelja kursora

A.4.2 – Skup funkcija BASIC 3 programskog jezika

ABS	– Apsolutna vrednost
ASC	– ASCII vrednost
ATN	– Arcustangens
CHR$	– ASCII izraz
COS	– Cosinus
EXP	– e^x
FNUM	– Prebacivanje broja u režim pokretnog zareza
FVAL	– Izračunavanje niza kao izraza
INT	– Prebacivanje broja iz pokretnog zareza u ceo broj
INUM	– Prebacivanje broja iz pokretnog zareza u ceo broj
LEN	– Dužina niza
LOG	– lnx
MEM	– Izračunavanje slobodnog RAM-a
MID$	– Generisanje niza
MOD	– Modulo
PEEK	– Direktno adresiranje memorije
PI	– 3.14159
RND	– Slučajan broj
SGN	– Vrednost za znak
SIN	– Sinus
SQR	– Kvadratni koren
TAB	– Tabulacija
USR	– Funkcija mašinskog jezika
KEY	– Generisanje decimalnog ekvivalenta koda pritisnute dirke

A.5 OPERATORI BASIC3 PROGRAMSKOG JEZIKA

Simbol	Značenje
↑	stepenovanje (4↑3=64)[4]
*	množenje
/	delenje
+	sabiranje, povezivanje niza
−	oduzimanje
=	jednako, uzima vrednost
<>	nejednako
>	veće od
<	manje od
>=	veće ili jednako
<=	manje ili jednako
AND	Bulov operator
OR	Bulov operator
XOR	Bulov operator
NOT( )	Bulov operator
:	odvaja naredbe u istom redu
;	ograničavač za PRINT
,	ograničavač za PRINT
(	grupni ograničavač
)	grupni ograničavač
% ... %	8-bitne binarne vrednosti (% 10010110 %)[1]
#	8-bitne heksadecimalne vrednosti[2]
&	16-bitne heksadecimalne vrednosti[3]

[1] Mora biti 8 binarnih cifara; tj. binarni ekvivalent za 2 je % 00000010 %

[2] Mora biti 2 heksadecimalne cifre; tj. heksadecimalni broj A je #0A.

[3] Mora biti 4 heksadecimalne cifre; tj. heksadecimalni broj A je &000A.

[4] Prihvatljivo je stepenovanje negativnog broja celim brojem na sledeći način −2↑3=−8.

Spisak naredbi je dat u tabelama I i II.

R(W): Registar adresiran sadržajem registra W gde je W=N, X ili P.
R(W).0: bajt manje težine registra R(W)
R(W).1: bajt veće težine registra R(W)
N0: bajt najmanje težine registra N.

Označavanje operacije
M(R(N))→D;R(N)+1
Znači: bajt iz memorijske lokacije adresiran sadržajem registra R(N) se prenosi u akumulator D i sadržaj registra R(N) se uvećava za 1.

TABLICA I – SPISAK NAREDBI
(prema tipu operacije)

NAREDBA	MNEMONIČKI KOD	KOD OPERACIJE	OPIS OPERACIJE
Naredbe za rad sa registrima			
INCREMENT REG N	INC	1N	R(N)+1→R(N)
DECREMENT REG N	DEC	2N	R(N)−1→R(N)
INCREMENT REG X	IRX	60	R(X)+1→R(X)
GET LOW REG N	GLO	8N	R(N).0→D
PUT LOW REG N	PLO	AN	D→R(N).0
GET HIGH REG N	GHI	9N	R(N).1→D
PUT HIGH REG N	PHI	BN	D→R(N).1
Naredbe za rad sa memorijom			
LOAD VIA N	LDN	0N	M(R(N))→D; FOR N NOT 0
LOAD ADVANCE	LDA	4N	M(R(N))→D; R(N)+1→R(N)
LOAD VIA X	LDX	F0	M(R(X))→D
LOAD VIA X AND ADVANCE	LDXA	72	M(R(X))→D; R(X)+1→R(X)
LOAD IMMEDIATE	LDI	F8	M(R(P))→D; R(P)+1→R(P)
STORE VIA N	STR	5N	D→M(R(N))
STORE VIA X AND DECREMENT	STXD	73	D→M(R(X)); R(X)−1→R(X)
Naredbe za logičke operacije			
OR	OR	F1	M(R(X)) OR D→D
OR IMMEDIATE	ORI	F9	M(R(P)) OR D→D; R(P)+1→R(P)
EXCLUSIVE OR	XOR	F3	M(R(X)) XOR D→D
EXCLUSIVE OR IMMEDIATE	XRI	FB	M(R(P)) XOR D→D; R(P)+→R(P)
AND	AND	F2	M(R(X)) AND D→D
AND IMMEDIATE	ANI	FA	M(R(P))AND D→D;R(P)+1→R(P)
SHIFT RIGHT	SHR	F6	SHIFT D RIGHT, LSB(D)→DF, 0→MSB(D)
SHIFT RIGHT WITH CARRY	SHRC	76	SHIFT D RIGHT, LSB(D)→DF, DF→MSB(D)
RING SHIFT RIGHT	RSHR		
SHIFT LEFT	SHL	FE	SHIFT D LEFT, MSB(D)→DF, 0→LSB(D)
SHIFT LEFT WITH CARRY	SHLC	7E	SHIFT D LEFT, MSB(D)→DF, DF→LSB(D)
RING SHIFT LEFT	RSHL		

TABLICA I – SPISAK NAREDBI

NAREDBA	MNEMONIČKI KOD	KOD OPERACIJE	OPIS OPERACIJE
Naredbe za aritmetičke operacije			
ADD	ADD	F4	M(R(X)) + D → DF, D
ADD IMMEDIATE	ADI	FC	M(R(P)) + D → DF, D; R(P) + 1 → R(P)
ADD WITH CARRY	ADC	74	M(R(X)) + D + DF → DF, D
ADD WITH CARRY, IMMEDIATE	ADCI	7C	M(R(P)) + D + DF → DF,D R(P) + 1 → R(P)
SUBTRACT D	SD	F5	M(R(X)) – D → DF,D
SUBTRACT D IMMEDIATE	SDI	FD	M(R(P)) – D → DF,D; R(P) + 1 → R(P)
SUBTRACT D WITH BORROW	SDB	75	M(R(X)) – D – (NOT DF) → DF,D
SUBTRACT D WITH BORROW, IMMEDIATE	SDBI	7D	M(R(P)) – D – (NOT DF) → DF,D; R(P) + 1 → R(P)
SUBTRACT MEMORY	SM	F7	D – M(R(X)) → DF,D
SUBTRACT MEMORY IMMEDIATE	SMI	FF	D – M(R(P)) → DF,D; R(P) + 1 → R(P)
SUBTRACT MEMORY WITH BORROW	SMB	77	D – M(R(X)) – (NOT DF) → DF,D
SUBTRACT MEMORY WITH BORROW, IMMEDIATE	SMBI	7F	D – M(R(P)) – (NOT DF) → DF,D R(P) + 1 → R(P)
Naredbe grananja u kratkom formatu			
SHORT BRANCH	BR	30	M(R(P)) → R(P).0
NO SHORT BRANCH (SEE SKP)	NBR	38	R(P) + 1 → R(P)
SHORT BRANCH IF D = 0	BZ	32	IF D = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF D NOT 0	BNZ	3A	IF D NOT 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF DF = 1	BDF		
SHORT BRANCH IF POS OR ZERO	BPZ	33	IF DF = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EQUAL OR GREATER	BGE		
SHORT BRANCH IF DF = 0	BNF	3B	IF DF = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF MINUS	BM		
SHORT BRANCH IF LESS	BL		
SHORT BRANCH IF Q = 1	BQ	31	IF Q = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF Q = 0	BNQ	39	IF Q = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF1 = 1 (1 = VSS)	B1	34	IF EF1 = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF1 = 0 (0 = VCC)	BN1	3C	IF EF1 = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF2 = 1 (1 = VSS)	B2	35	IF EF2 = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF2 = 0 (0 = VCC)	BN2	3D	IF EF2 = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF3 = 1 (1 = VSS)	B3	36	IF EF3 = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF3 = 0 (0 = VCC)	BN3	3E	IF EF3 = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF4 = 1 (1 = VSS)	B4	37	IF EF4 = 1, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)
SHORT BRANCH IF EF4 = 0 (0 = VCC)	BN4	3F	IF EF4 = 0, M(R(P)) → R(P).0 ELSE R(P) + 1 → R(P)

TABLICA I – SPISAK NAREDBI
(prema tipu operacije)

NAREDBA	MNEMONIČKI KOD	KOD OPERACIJE	OPIS OPERACIJE
Naredbe grananja u dugom formatu			
LONG BRANCH	LBR	CO	M(R(P))→R(P).1 M(R(P)+1)→R(P).0
NO LONG BRANCH (SEE LSKP)	NLBR	C8	R(P)+2→R(P)
LONG BRANCH IF D=0	LBZ	C2	IF D=0,M(R(P)→R(P).1 M(R(P)+1)→R(P).0 ELSE R(P)+2→R(P)
LONG BRANCH IF D NOT 0	LBNZ	CA	IF D NOT 0, M(R(P))→ R(P).1 M(R(P)+1)→ R(P).0 ELSE R(P)+2→R(P)
LONG BRANCH IF DF=1	LBDF	C3	IF DF=1, M(R(P))→R(P).1 M(R(P)+1)→ R(P).0 ELSE R(P)+2→R(P)
LONG BRANCH IF DF=0	LBNF	CB	IF DF=0, M(R(P))→R(P).1 M(R(P)+1)→ R(P).0 ELSE R(P)+2→R(P)
LONG BRANCH IF Q=1	LBQ	C1	IF Q=1, M(R(P))→R(P).1 M(R(P)+1)→R(P):0 ELSE R(P)+2→R(P)
LONG BRANCH IF Q=0	LBNQ	C9	IF Q=0, M(R(P)→R(P).1 M(R(P)+1)→ R(P).0 ELSE R(P)+2→R(P)
Naredbe preskoka			
SHORT SKIP (SEE NBR)	SKP	38	R(P)+1→R(P)
LONG SKIP (SEE NLBR)	LSKP	C8	R(P)+2→R(P)
LONG SKIP IF D=0	LSZ	CE	IF D=0, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF D NOT 0	LSNZ	C6	IF D NOT 0, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF DF=1	LSDF	CF	IF DF=1, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF DF=0	LSNF	C7	IF DF=0, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF Q=1	LSQ	CD	IF Q=1, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF Q=0	LSNQ	C5	IF Q=0, R(P)+2→R(P) ELSE CONTINUE
LONG SKIP IF IE=1	LSIE	CC	IF IE=1, R(P)+2→R(P) ELSE CONTINUE
Upravljačke naredbe			
IDLE	IDL	00	WAIT FOR DMA OR INTERRUPT; M(R(0)→BUS
NO OPERATION	NOP	C4	CONTINUE
SET P	SEP	DN	N→P
SET X	SEX	EN	N→X
SET Q	SEQ	7B	1→Q
RESET Q	REQ	7A	0→Q
SAVE	SAV	78	T→M(R(X))
PUSH X,P TO STACK	MARK	79	(X,P)→T; (X,P)→M(R2)) THEN P→X; R(2)−1→R(2)
RETURN	RET	70	M(R(X))→(X,P); R(X)+1→(RX) 1→IE
DISABLE	DIS	71	M(R(X))→(X,P); R(X)+1→R(X) 0→IE

TABLICA I – SPISAK NAREDBI

NAREDBA	MNEMONIČKI KOD	KOD OPERACIJE	OPIS OPERACIJE
Naredbe za ulaz/izlaz			
OUTPUT 1	OUT 1	61	M(R(X))→BUS;R(X)+1→R(X);N=1
OUTPUT 2	OUT 2	62	M(R(X))→BUS;R(X)+1→R(X);N=2
OUTPUT 3	OUT 3	63	M(R(X))→BUS;R(X)+1→R(X);N=3
OUTPUT 4	OUT 4	64	M(R(X))→BUS;R(X)+1→R(X);N=4
OUTPUT 5	OUT 5	65	M(R(X))→BUS;R(X)+1→R(X);N=5
OUTPUT 6	OUT 6	66	M(R(X))→BUS;R(X)+1→R(X);N=6
OUTPUT 7	OUT 7	67	M(R(X))→BUS;R(X)+1→R(X);N=7
INPUT 1	INP 1	69	BUS→M(R(X))→D;N=9
INPUT 2	INP 2	6A	BUS→M(R(X))→D;N=A
INPUT 3	INP 3	6B	BUS→M(R(X))→D;N=B
INPUT 4	INP 4	6C	BUS→M(R(X))→D;N=C
INPUT 5	INP 5	6D	BUS→M(R(X))→D;N=D
INPUT 6	INP 6	6E	BUS→M(R(X))→D;N=E
INPUT 7	INP 7	6F	BUS→M(R(X))→D;N=F

ELEKTRONSKA INDUSTRIJA — NIŠ
Ei-RO „Ei RAČUNARI"
OOUR Fabrika računskjh mašina
Bul. Veljka Vlahovića 80-82
18000 Niš
Direktor (018) 325-461
Direktor marketinga (018) 55-583
Plasman (018) 54-779, 51-568
Softver (018) 52-782, 52-876
Školski centar (018) 54-090
Servis (018) 54-867
Tlx. 16283 YU Ei FRM

POSLOVNA JEDINICA BEOGRAD
Rudo 2 — 11000 Beograd
Direktor (011) 488-260, 483-265

POSLOVNA JEDINICA TITOGRAD
Ul. Brace Bracanovića 58
81000 Titograd
Tel. (081) 34-739, 34-612

ASIMPEX OOUR MEHANOGRAFSKI ZAVOD 21000 NOVI SAD