OSNOVNE KARAKTERISTIKE PROGRAMSKOG JEZIKA BASIC

BASIC je skraćenica od izraza u engleskom jeziku **Beginner´s All-purpose Symbolic Instruction Code**, što bi u našem, malo slobodnijem prevodu značilo: programski jezik opšte namjene za početnike. To je programski jezik koji se može jednostavno naučiti, a takođe ga je jednostavno koristiti u rješavanju raznih problema uz pomoć računara.

BASIC se pojavio još 1963.godine na koledžu Darmut, autori su profesori sa ovog koledža. Mada se od tada BASIC dosta mijenjao i dograđivao, ipak su osnovne karakteristike zadržane.

Instrukcije u BASIC-u su vrlo slične rečenicama engleskog jezika. Ukoliko se rješavaju matematički problemi, onda se matematički izrazi zapisuju vrlo slično standardnoj notaciji u algebri i aritmetici. Postoji još jedna veoma važna karakteristika ovog jezika. BASIC je tzv. konverzacioni jezik u kome se lahko ostvaruje stalna komunikacija između korisnika i računara. Zahvaljujući ovoj komunikaciji, korisnik jednostavno i brzo unosi program, ispravlja greške u programu, a takođe i vrši određene dopune.

Program se zadaje u obliku niza instrukcija. Ove instrukcije se moraju pisati prema tačno utvrđenim pravilima. Skup svih pravila čini sintaksu programskog jezika, u ovom slučaju BASIC-a.

Instrukcije u BASIC-u se pišu u redovima. Maksimalan broj znakova u jednom redu je 72. u svakom redu mogu da budu jedna ili više instrukcija. Veoma je važno da se instrukcija koja počne u jednom redu u tom redu i završi. To nije isto kao kod prirodnog jezika, gdje jedna rečenica može da bude raspoređena u više redova.

Da bi se programi, ali i podaci mogli napisati u BASIC-u, definisan je skup znakova koji obuhvata:

a) Slova engleskog alfabeta: A,B,C,....X,Y,Z. Ponekad su uključena i mala slova, ali uglavnom za podatke, dok se u programima isključivo primjenjuju velika slova.
b) Sve decimalne cifre: 0,1,2,....7,8,9.
c) Specijalni znaci: +,-,*,=,<,>,.... Skup ovih znakova nije ograničen i zavisi od vrste računara, kao i od savremenosti prevodioca za BASIC. Sve češće se u ovu grupu uključuju i specijalni grafički znaci koji omogućavaju crtanje uz pomoć računara.
d) Specijalni znak koji predstavlja jedno prazno mjesto u tekstu programa ili podatka. To je analogno praznom mjestu koje se pojavljuje između riječi u pisanom prirodnom jeziku, jer i na tastaturi pisaće mašine postoji posebna tipka za ovaj ˝znak˝. Obično se on naziva blanko-znak ili prazan znak.
e) Specijalni simboli koje definiše korisnik.

Svi ovi znaci, u različitim kombinacijama, omogućavaju zapisivanje programa i podatka u programskom jeziku BASIC.

Da bi se sagledale osnovne karakteristike jednog programskog jezika i mogućnosti njegove primjene u rješavanju raznih problema, potrebno je objasniti:

a) kako se zapisuju i koje su vrste podataka u datom programskom jeziku.
b) Kako se zapisuju i koje su vrste izraza koje podržava programski jezik. Najpoznatiji su aritmetički izrazi, ali oni ne moraju da budu i jedini.
c) Koje su osnovne instrukcije u okviru jezika za unošenje podataka, njihovu obradu i prikazivanje rezultata.
d) Koje su složene programske instrukcije datog programskog jezika.
e) Neke specifičnosti programskog jezika.

PODACI

Osnovna namjena svakog računara je manipulacija raznim podacima. Ti podaci mogu da budu plate radnika u fabrici, ocjene učenika u školi, cijene u prodavnicama, ali takođe i imena, adrese, mjerenja u raznim eksperimentima itd. Kada se razmišlja o računaru, većinom se misli na numeričke podatke i njihovu obradu. Međutim, računar je prilagođen i za rad sa nenumeričkim podacima kao što su razni tekstovi. Tipičan primjer su savremeni računari koji imaju mogućnost prevođenja tekstova iz jednog prirodnog jezika u drugi (npr. Iz engleskog u japanski). Ovi tekstovi za date računare predstavljaju ulazne podatke, odnosno izlazne rezultate.

Različiti programski jezici prihvataju i razne podatke. Podaci u BASIC-u mogu da budu:

1) Numerički, i to samo decimalni (jer nema npr. Tipova kao što su cjelobrojni, binarni i drugi koje srećemo u drugim programskim jezicima). Ovo znatno pojednostavljuje učenje programskog jezika BASIC.
2) Alfa-numerički ili tekstualni podaci u obliku niza znakova koji su dozvoljeni u BASIC-u.

Decimalni brojevi u BASIC-u predstavljaju se na dva načina.

Prvi je pisanje brojeva u obliku elementarnih numeričkih konstanti, i to napisanih kao niz cifara sa ili bez decimalne tačke. Broju može da predhodi i predznak -, dok se za pozitivne brojeve predznak + podrazumijeva. Primjeri ovako napisanih brojeva su:

125.432 1089 0.234123 -12.65

Treba zapaziti slijedeće činjenice. Decimalna tačka u BASIC-u predtavlja standardni znak decimalnog zareza u uobičajenoj matematskoj notaciji. Maksimalan broj cifara koje se mogu predstaviti u ovom načinu zapisivanja je ograničen na 6 najznačajnijih cifri.

Drugi način zapisivanja decimalnih brojeva je u eksponencijalnoj ili E-notaciji. Opšti oblik pisanja broja u ovoj notaciji je:

aEn

gdje je: a- elementarna numerička konstanta

n- cijeli broj,

E- oznaka da se radi o eksponencijalnoj predstavi broja, a značenje ovog izraza je: $a \times 10^n$.

Ovaj način predstavljanja decimalnih brojeva je naročito pogodan za veoma velike i veoma male vrijednosti decimalnog broja. Tako npr.broj:

-2.36E8 je pogodan i skraćen oblik pisanja broja: -236000000, a s druge strane, broj zapisan u E-notaciji:

1.23E-10 predstavlja izrazito malu vrijednost: 0.000000000123. slijedeći su primjeri također zapisani u E-notaciji: 1.15E-5; -3.1E-12; 2E10

pored numeričkih podataka, računar može da prihvati i tekstualne podatke. Oni su u BASIC-u definisani kao znakovne konstante, jer se u suštini i sastoje iz niza različitih znakova koje dozvoljava ovaj programski jezik. Da bi se razlikovale od drugih elemenata programskog jezika BASIC, znakovne konstante se pišu pod znacima navoda. Primjeri zapisivanja ovih podataka su:

˝TEHNIČKI ODGOJ I INFORMATIKA˝

″NASTAVA POČINJE U 7:30 SATI″
″_____″
″2000″

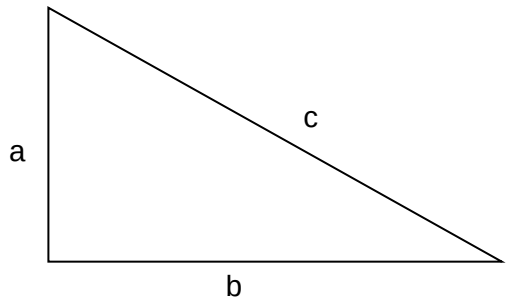
Potrebno je posebnu pažnju obratiti na posljednji primjer. I pored toga što se konstanta sastoji samo od znakova koji su cifre, ipak je to znakovna konstanta, jer je pod navodnicima. Prema tome, ona se ne može posmatrati kao običan broj kojim se mogu vršiti standardne aritmetičke operacije, nego kao običan tekst.

PROMJENLJIVE

Promjenljiva je veoma važan koncept u računarskim programima. Određena je sa svoje dvije veličine, a to su:

- ime i
- vrijednost

Ovaj koncept je već poznat iz matematike. Na primjer, neka se strane pravouglog trougla označe sa a,b i c kao na slici 1.



slika 1. pravougli tougao

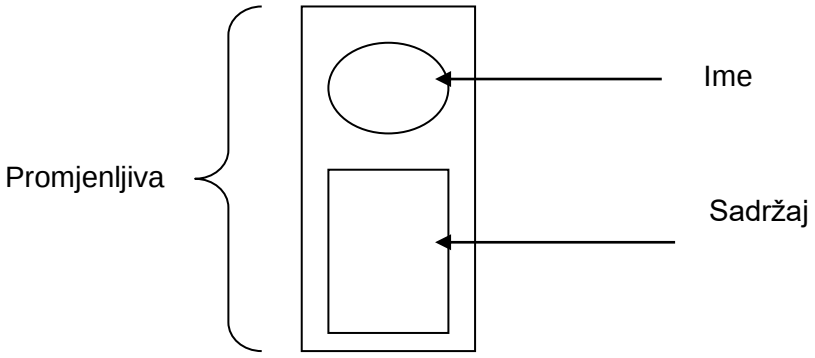
Pitagorina teorema definiše vezu među dužinama ovih strana: $c^2=a^2+b^2$ ova relacija koristi, u suštini, promijenljive a,b i c da bi definisala opštu formulu, koja se može primijeniti u svakom specifičnom slučaju. Na primjer, ukoliko je strana a jednaka 3, strana b jednaka 4, pomoću ove formule izračunaćemo i treću stranu pravouglog trougla, odnosno njegovu hipotenuzu c:

$$c^2 = 3^2 + 4^2$$
$$c^2 = 25$$
$$c = 5.$$

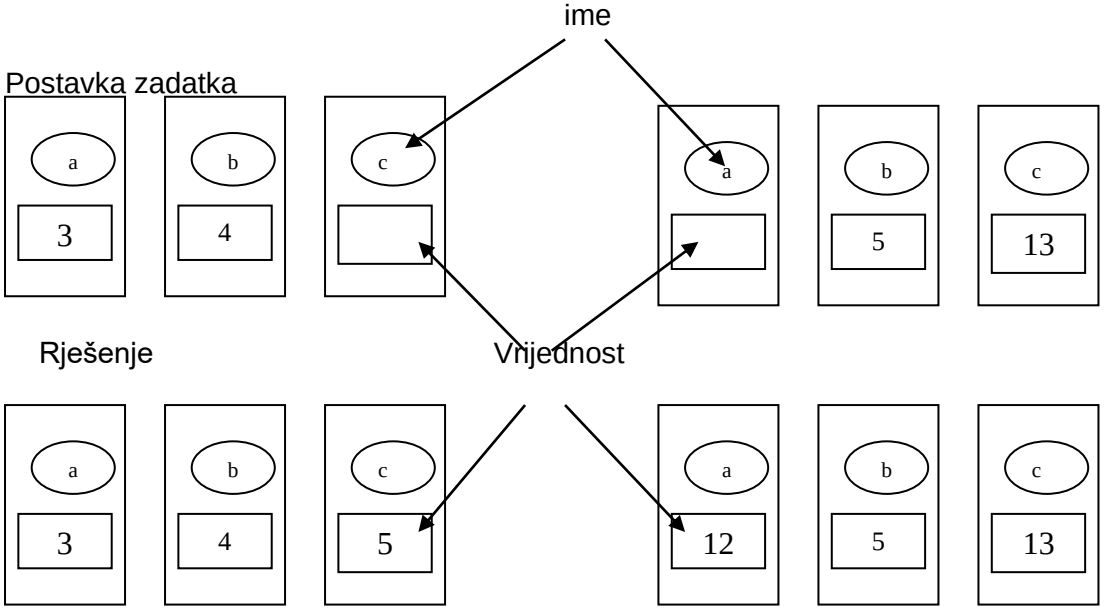
U računarskom programu, na isti način, primjena promijenljivih omogućuje specifikaciju opštih formula za sračunavanje. Kao i u matematskom primjeru, promijenljive u programu imaju svoje ime, a takođe pri njihovom korištenju i određenu vrijednost. U opštem slučaju one mogu da posjeduju mnoge vrijednosti. Međutim bitno je to da promijenljiva u jednom trenutku, kada se koristi, može da ima samo jednu vrijednost.

Sada se pojam promijenljive može povezati i s njenom interpretacijom na računaru. Poznato je da se svi podaci drže u memoriji. Pri tome, osnovni elementi memorije, bilo da su to bajti ili riječi, imaju svoju adresu i sadržaj. Ako se jedna promijenljiva pridruži određenom elementu memorije, onda ime promjenljive odgovara adresi, dok vrijednost promjenljive odgovara sadržaju tog memorijskog elementa. Pri tome je ime nepromjenljivo, jer se radi o tačno određenom

memorijskom elementu, dok se vrijednpost može mijenjati. Šematski prikazano, promjenljiva se može posmatrati kao jedan blok s dvije komponente, kao što je to prikazano na slici 2.



Slika 2. Šematski prikaz promjenljive



Slika 3. Ponašanje promijenljivih u toku izvršenja zadatka

U ovom svjetlu može da se sagleda i predhodni primjer pravouglog trougla. Ako bi računar rješavao postavljeni zadatak, onda bi su u memoriji trebala rezervisati 3 mjesta čija su imena a,b, i c. Za prvu primjenu pitagorine teoreme izgled memorijskih elemenata prije i nakon sračunavanja bio bi kao na slici 3.

Promijenljive su, kao i konstante osnovni elementi svakog programskog jezika, pa, prema tome, i BASIC-a. Vidjeli smo da u BASIC-u konstante mogu da budu numeričke i znakovne. Isti je slučaj i s promjenljivim, tj., zavisno od vrijednosti koje imaju, mogu da budu:

- numeričke promjenljive i
- znakovne promjenljive.

Vrijednost numeričke promjenljive je podatak u brojnom obliku. Ime numeričke promjenljive je niz znakova, ali zapisan na tačno određeni način. Ovo ime može da bude ili samo jedno slovo ili slovo s jednom cifrom. Novije verzije BASIC-a dozvoljavaju da ime promjenljive sadrži i više znakova. Prema tome, primjeri imena numeričkih promjenljivih su:

B, C8, W1, B1

Dok A.B, 1SUMA, 9F, C45,0 nisu važeća imena za numeričke promjenljive u BASIC-u.

Vrijednost znakovne promjenljive je podatak u obliku teksta, odnosno niza znakova. Ime znakovne promjenljive se definiše u BASIC-u kao i kod numeričke promjenljive, samo što se na kraju dodaje znak $. Imena znakovnih promjenljivih u BASIC-u su:

A1$, B$, X3$, WO$ itd. Dok AB$$, 1$, AB.23$,$ nisu važeća imena znakovne promjenljive u BASIC-u.

Ukoliko je jedna promjenljiva definisana kao numerička, onda njen sadržaj ne može biti znakovni, a važi i obratno. Primjeri pravilno i nepravilno definisanih promjenljivih s njihovim sadržajem:

Pravilno:

Ime	vrijednost
A	126
C3	4.47
M$	″IME″
W1$	″O.Š. KOVAČI″

Nepravilno:

Ime	vrijednost
M23	128
M2	″POSAO″
W12$	″124″
W2$	124

IZRAZI

Pojam izraza poznat je već iz matematike.
Na primjer, (2+17,5)*(23-48,6) je aritmetički izraz koji nakon sračunavanja ima određenu brojnu vrijednost. Međutim, izraz može da bude i oblika: (2X+3Y)/4Z, gdje u izrazu, pored brojnih vrijednosti povezanih aritmetičkim operatorima i promjenljive X,Y,Z.

Konstante i promjenljive u BASIC-u se takođe mogu povezati u izrazu koji predstavljaju osnovu bilo kog sračunavamja u računaru. Kao prvo, to su aritmetički izrazi, koji se u BASIC-u vrlo slično zapisuju kao i u matematici. Oni se predstavljaju nizom brojnih konstanti i brojnih promjenljivih povezanih aritmetičkim operatorima. Na tabeli definisani su osnovni skup aritmetičkih operatora i način njihovog označavanja u BASIC-u.

OPERATOR	ZNAČENJE
+	Sabiranje (i pozitivan predznak)
-	Oduzimanje (i negativan predznak)
*	Množenje
/	Dijeljenje
^	Stepenovanje

Da bi se naučilo zapisivanje izraza u BASIC-u, može se poći od standardnih matematičkih izraza i vršiti prevođenje u odgovarajuće izraze programskog jezika BASIC. Slijedeći primjeri pokazuju ovo pretvaranje:

Aritmetički izraz u matematici	Aritmetički izraz u BASIC-u
A+B	A+B
3x+2y	3*X+2*Y
-7+2,45	-7+2.45

(x+2)y	(X+2)*Y
$\frac{2+5}{3-a}$	(2+5)/(3-A)
$(a+4{,}23)^2$	(A+4.23)^2
$\frac{25}{48{,}5}$	25/48.5
$\frac{\frac{a}{b}+c}{d}\cdot c$	((A/B+C)/D)*E

Već i na osnovu ovih primjera se vidi da se i u aritmetičkim izrazima u BASIC-u mogu upotrebljavati i zagrade. Njihova namjena je ista kao i kod standardnog matematskog označavanja. Služe za definisanje prioriteta operacija u aritmetičkom izrazu. Pošto su operatori stepenovanja prioritetniji od operatora množenja i dijeljenja, a ovi od sabiranja i oduzimanja, zagradama se rješava problem obavljanja operacije nižeg prioriteta prije onih sa višim prioritetom. Na primjer, ako imamo izraz X+YZ, on će u BASIC-u biti zapisan kao X+Y*Z.

Međutim, ako je prvo potrebno vršiti sabiranja promjenljivih X i Y, a tek onda množenje zbira s promjenljivim Z, tada će ovaj aritmetički izraz glasiti (X+Y)Z, odnosno u BASIC-u (X+Y)*Z.

Ukoliko je u aritmetičkom izrazu potrebno zapisati više zagrada kao, na primjer, $\left\{x+\left[y-(a-3)^2\right]^3\right\}(a+b)$, tada se u Basic-u sve ove zagrade označavaju na isti način: (X+(Y-(A-3)^2)^3)*(A-B).

Prilikom pisanja ovako složenih izraza potrebno je uvijek prekontrolisati da li je broj otvorenih jednak broju zatvorenih zagrada.

Druga vrsta izraza su znakovni izrazi. Oni vezuju znakovne konstante i promjenljive, a osnovni operator za povezivanje je & i označava spajanje dva niza znakova u jedan. Tipični primjeri znakovnih izraza su:

```
"HASAN"&"HASANOVIĆ"
A$&"SARAJEVO"
A$&B$.
```

Kada se u znakovnim izrazima pojave znakovne varijable, to ne znači da se njihovo ime (A$,B$)povezuje, nego da se povezuju njihove znakovne vrijednosti koje se vode pod tim imenom. Potrebno je odmah naglasiti da je vrijednost znakovnog izraza opet jedna znakovna konstanta. Prema tome, znakovni izrazi se ne smiju povezati s aritmetičkim izrazima u nekakve složenije.

Konačno treća vrsta izraza u Basic-u su relacioni izrazi, koji se mogu posmatrati analogno logičkim sudovima u matematici. Oni omogućavaju ispitivanje odnosa dva aritmetička ili dva znakovna izraza. Vrijednost relacionog izraza je binarna veličina, zapravo, kao i kod svakog logičkog suda ona može da bude jedna od vrijednosti istinitosti: 1(tačno) ili 0(netačno). Da bi se mogli ispitati odnosi pojedinih izraza, uvode se relacioni operatori dati na slijedećoj tabeli.

OPERATOR	OZNAKA U BASIC-u	ZNAČENJE
=	=	Jednako
<	<	Manje

≤	<=	Manje ili jednako
>	>	Veće
≥	>=	Veće ili jednako
≠	<>	Različito

Sada se mogu definisati neki konkretni primjeri relacionih izraza odgovarajućom vrijednošću izraza.

RELACIONI IZRAZ	VRIJEDNOST
2=5	Netačan (0)
2<>5	Tačan (1)
3+5.8<127.9	Tačan (1)
2*6<=4*3	Tačan (1)
2*6<4*3	Netačan (0)
1.56>2.5	Netačan (0)
″SARAJEVO″=″SARAJEVO″	Tačan (1)
″SARAJEVO″=″ SARAJEVO ″	Netačan (0)
″TUZLA″<>″ZENICA″	Tačan (1)

Ovi relacioni izrazi ne odnose se samo na brojne i znakovne konstante nego i na promijenljive. Tako važe i relacioni izrazi:

```
X^2<Y
(2.34+A)*B<>A+5
A$=″HASAN″
```

Međutim, ne može se direktno odrediti njihova vrijednost, odnosno ustanoviti da li su tačni ili ne sve dok se ne znaju vrijednosti promjenljivih koje se nalaze u ovim izrazima.

Na primjer, ukoliko je u prvom primjeru vrijednost promjenljive X jednaka 3, a Y jednaka 7, tek onda se, uvrštavajući ove vrijednosti u lijevu i desnu stranu izraza, može odrediti i stvarna vrijednost relacionog izraza. U ovom slučaju ona je netačna, jer 3^2 nije manje od 7.

Treba zapamtiti da se u relacionom izrazu kao što je npr.

```
X=(2+3)*4
```

Ne dodjeljuje promjenljivoj X vrijednost sa desne strane (u ovom slučaju to je 20), nego se ta vrijednost samo poredi s trenutnom vrijednošću promjenljive X. Ukoliko je i ona jednaka 20, onda je izraz tačan, a ukoliko nije, vrijednost izraza je netačna.

UGRAĐENA FUNKCIJA

Svaki programski jezik može da ima svoje ugrađene funkcije, mada se neke standardne pojavljuju u većini programskih jezika. Tipičan primjer je funkcija

nalaženja kvadratnog korijena iz nekog broja. U matematici je bilo to zapisano kao $\sqrt{25}$, a kao rezultat ove funkcije (korjenovanje) nad brojem 25 se dobija vrijednost, u ovom slučaju 5.

BASIC dozvoljava niz ugrađenih funkcija, a tipičan primjer je upravo funkcija korjenovanja. Sve ove funkcije definisane su kao niz tačno određenih znakova, a izvršavaju se ne samo nad konstantama nego i nad izrazima definisanim u prethodnom poglavlju.

Na primjer, funkcija korjenovanja označena je u Basic-u sa SQR, pa, prema tome, predhodni primjer se u Basic-u zapisuje kao SQR(25). Funkcija SQR za argument može da ima, pored konstanti i promjenljivih, i složene izraze:

`SQR(X^3)` odgovara $\sqrt{X^3}$

`SQR((A+B)^2/3)` odgovara $\sqrt{\frac{(A+B)^2}{3}}$

Pri tome, ova funkcija se odnosi samo na aritmetičke izraze, tako da je nepravilno pisati: `SQR(HASAN);` `SQR(A$).`

Postoji niz funkcija ovog tipa koje se primjenjuju nad aritmetičkim izrazima. Tipične funkcije s njihovim značenjem date su u sljedećoj tabeli.

FUNKCIJA	ZNAČENJE	PRIMJER
SQR(brojni izraz)	Izračunavanje kvadratnog korijena brojnog izraza	`SQR(X*Y)`
EXP(brojni izraz)	Izračunavanje eksponencijalne vrijednosti brojnog izraza	`EXP(X+3)`
LN(brojni izraz)	Izračunavanje prirodnog logaritma brojnog izraza	`LN(1/(X+1))`
ABS(brojni izraz)	Izračunavanje apsolutne vrijednosti brojnog izraza	`ABS(X-Y)`
SGN(brojni izraz)	Određivanje predznaka brojnog izraza	`SGN(X/Y)`
INT(brojni izraz)	Određivanje cjelokupne vrijednosti brojnog izraza	`INT(X^2)`
SIN(brojni izraz)	Izračunavanje trigonometrijske funkcije sinus brojnog izraza	`SIN(SQR(X))`
COS(brojni izraz)	Izračunavanje trigonometrijske funkcije cosinus brojnog izraza	`COS(1/X)`
TAN(brojni izraz)	Izračunavanje trigonometrijske funkcije tangens brojnog izraza	`TAN(LN(1+X))`

Kao što postoji klasa funkcija koje se izvršavaju nad aritmetičkim izrazima, tako u BASIC-u postoje ugrađene funkcije za znakovne izraze. Skup ovih funkcija može da bude različit na pojedinim računarima i za razne prevodioce za BASIC. Primjer ovakve funkcije je ugrađena funkcija koja računa dužinu niza znakova. Prema tome, njena konačna vrijednost je brojna, mada operiše nad nizovima,

konstantama i promjenljivim. Funkcija je označena sa **LEN,** a primjeri njene primjene su:

```
LEN(˝HASAN˝),
LEN(˝ HASAN ˝),
LEN(A$&b$).
```

Pri tome, vrijednosti ove funkcije su: u prvom slučaju 5, jer ima pet znakova u konstanti ´HASAN´, u drugom 7, jer se ubrajaju i prazni znaci, dok u trećem primjeru ova vrijednost zavisi od vrijednosti znakovnih promjenljivih A$ i B$. Na primjer, ako je vrijednost A$ jednaka ´17 000´, a promjenljive B$ jednaka ´SARAJEVO´, tada je vrijednost ove funkcije jednaka 14. pored toga što se funkcije izvršavaju nad izrazima, isto tako i one same mogu da budu uključene u složenije aritmetičke, odnosno znakovne izraze. Primjeri su sljedeći:

```
A+B*SQR(B^2+A^2)
SQR(A+3)+SQR(A+5))/10
A+5*LEN(˝SARAJEVO˝)
LEN(X$)+LEN(Y$).
```

Očito je da dobro treba paziti da izrazi u funkcijama budu odgovarajući, ali i da funkcije u složenijim izrazima budu odgovarajuće. Ako je rezultat funkcije broj, onda ona može da bude samo u aritmetičkim izrazima. Ako je rezultat niz, tada se funkcija može primijeniti samo u znakovnim izrazima.

ELEMENTARNI ISKAZI U BASIC-u (NAREDBE)

Da bi se definisale osnovne instrukcije programskog jezika BASIC, potrebno se za trenutak vratiti na dijagram toka. On je bio rezultat prve faze u rješavanju problema i sastojao se od niza povezanih blokova. Da bi prelazak s dijagrama toka na programski jezik bio što jednostavniji, potrebno je uspostaviti tačan odnos između pojedinih blokova dijagrama toka i instrukcije (naredbe) u programskom jeziku. U tom slučaju pisanje programa bi bilo jednostavno prevođenje svakog bloka iz dijagrama toka u jednu ili više tačno određenih instrukcija.

Polazeći od ove predpostavke, mogu se zapaziti tri osnovna tipa blokova u dijagramu toka. To su blokovi ulaza, obrada i izlaza, što je i logično, jer se suština primjene računara svodi na to da se na osnovu nekih ulaznih podataka vrši zahtijevana obrada i prezentiraju rezultati obrade na izlazu. Svaki od ovih blokova ima u većini programskih jezika, pa tako i u BASIC-u, odgovarajuće instrukcije. Ove instrukcije nazivaju se još i iskazi. Tako za blok ulaza postoje odgovarajući iskazi ulaza u BASIC-u, za obradu postoje iskazi pridruživanja, a izlazni blok se preslikava na izlazne iskaze.

Iskazi ulaza

Za blokove ulaza u dijagramu toka, koji naznačavaju unošenje podataka u računar, odgovarajući BASIC-iskaz ima oblik:

INPUT lista promjenljivih,

Gdje se pod listom promjenljivih podrazumijeva više promjenljivih odvojenih zarezima. INPUT je obavezna riječ (na engleskom znači ulaz) kojom se definiše šta treba raditi s nabrojanim promjenljivim. Primjeri dobro napisanih iskaza ovog tipa u BASIC-u su sljedeći:

```
INPUT A
INPUT A1
INPUT ˝UNESI NEKI BROJ:˝;A
INPUT ˝UNESI IME I PREZIME˝;A$
INPUT X,Y,Z
INPUT A6$,B2$.
```

Kada računar u toku izvršenja programa naiđe na iskaz INPUT, tada se na ekranu monitora pojavi upinik (?) i računar očekuje da korisnik unese podatke. Promjenljive definisane u iskazu govore o tome koliko podataka treba unijeti u računar i kojeg su tipa (brojni ili znakovni). Tako, u prvom i drugom primjeru treba unijeti samo jedan brojni podatak, u trećem primjeru na ekranu će pisati:

UNESI NEKI BROJ:?

U četvrtom primjeru na ekranu će pisati:

UNESI IME I PREZIME ?

U petom primjeru računar očekuje da se unesu tri brojna podatka, a u šestom primjeru dva znakovna podatka. Ovako uneseni podaci, odnosno brojne i znakovne konstante pohranjuju se u memoriji računara, i to kao sadržaj promjenljivih definisanih u iskazu.

U prvom i drugom slučaju, ukoliko je unijeta vrijednost 5, to znači da će, u daljem programu, promjenljiva sa imenom A odnosno A1 imati vrijednost 5. ova vrijednost se kasnije može pozvati da se obrađuje, a poziva se imenom promjenljive kojoj je pridružena kao sadržaj (tj. Ako napišemo PRINT A odnosno PRINT A1, na ekranu će se ispisati broj 5).

Ukoliko se u jednom ulaznom iskazu zahtijeva više podataka, onda se oni unose jedan iza drugog, odnosno redaju se sa brojevima naredbi na početku svakog reda u programu naprimjer:

```
10 INPUT ˝UNESI IME I PREZIME˝;A$
20 INPUT ˝UNESI RAZRED˝;A
30 INPUT ˝UNESI KOJI SI BROJ U DNEVNIKU˝;B   itd.
```

Iskazi pridruživanja

Ovo je osnovni iskaz u BASIC-u pomoću koga se vrše sva sračunavanja. Osnovni oblik ovog iskaza je:

Promjenljiva=iskaz

A značenje je sljedeće:

Kada računar pri izvršenju programa dođe do ovog iskaza, on prvo sračunava vrijednost izraza na desnoj strani i tu vrijednost postavlja kao novi sadržaj promjenljive definisane na lijevoj strani.

Neka je na primjer definisan iskaz:

```
X=2*2-3/6+1.5
```

Tada će računar prvo sračunati vrijednost izraza na desnoj strani: to je u ovom slučaju 5. nova vrijednost 5 pridružuje se promjenljivoj X kao njen novi sadržaj (bez obzira kakav je sadržaj imala prije).

U izrazu sa desne strane mogu da se pojave promjenljive pa i funkcije. Pri sračunavanju, računar uvijek radi s vrijednostima promjenljivih, a ne s njihovim imenima. Tako npr. Ako je dat iskaz:

```
A1=B1+SQR(9),
```

Računar će prvo uzeti vrijednost promjenljive B1. neka je ona 10. zatim će biti izračunata vrijednost funkcije korjenovanja. U ovom primjeru je to 3. konačno, biće sabrane vrijednosti 10 i 3 i nova vrijednost 13 biće pridružena promjenljivoj A1 na lijevoj strani iskaza.

Ovakva interpretacija iskaza pridruživanja omogućuje da se shvate i izrazi koji su u standardnom matematskom obliku bili nelogični. Npr. Iskaz pridruživanja:

```
X1=X1+5
```

Ne predstavlja jednačinu čije je rješenje nelogično, 0=5. ovaj iskaz govori o tome da je potrebno naći vrijednost promjenljive X1. neka je ona 7. toj vrijednosti potrebno je dodati 5, a rezultat 12 pridružiti kao novu vrijednost istoj promjenljivoj X1. stara vrijednost 7 u tom slučaju se gubi, jer više nigdje nije zapisana u računaru.

Do sada su se iskazi pridruživanja odnosili samo na brojne promjenljive i brojne izraze. Međutim, isti iskazi mogu povezati znakovne promjenljive sa znakovnim izrazima. Npr.iskaz:

```
A$=″SARA″& ″JEVO″
```

Uzima prvu znakovnu konstantu sa desne strane, a to je ″SARA″; nju povezuje s drugom znakovnom konstantom ″JEVO″ i time dobije novu znakovnu konstantu ″SARAJEVO″, koju pridružuje kao novu vrijednost znakovnoj promjenljivoj A$.

Iskaz pridruživanja može da počne u BASIC-u posebnom riječi LET (engleska riječ sa značenjem –neka je) kako bi se ovaj iskaz razlikovao od drugih. Međutim većina prevodilaca za BASIC ne zahtijevaju ovu posebnu riječ i prilikom programiranja korisnici je ne upisuju, jer, to je samo dodatno, nepotrebno kucanje na računaru.

Na kraju je navedeno nekoliko korektno napisanih iskaza pridruživanja u BASIC-u:

```
X5=10.27
A$=″UNESITE PODATKE″
B1=SQR(X^2+Y^2)
C9=LEN(A$)+LEN(B$)
F8=-BSQR(B^2-4+A*C)
A$=″IME″& ″PREZIME:″&B$
```

Za vašu vježbu treba analizirati šta se dešava pri izvršenju svakog od ovih iskaza.

Iskaz izlaza

Zahvaljujući ovom iskazu, računar predaje korisniku sve potrebne informacije i rezultate do kojih je došao.

Za blokove izlaza u dijagramu toka, koji naznačavaju prikazivanje raznih tekstova i rezultata korisniku, odgovarajući BASIC-iskaz ima oblik:

PRINT lista promjenljivih, konstanti i izraza.

Za razliku od ulaznog iskaza, u ovom slučaju elementi liste mogu da budu kako promjenljive tako i konstante, pa čak i izrazi. Naravno, i u ovom slučaju oni su odvojeni zarezima ili tačka-zarez znacima. Riječ PRINT je obavezna i ona naznačava da se radi o izlaznom iskazu (na engleskom jeziku znači štampaj).

Ovaj iskaz preuzima vrijednost promjenljivih datim u listi, odnosno direktno uzima konstante iz liste i štampa ih na izlaznoj jedinici računara (ekranu monitora i štampaču). Ukoliko se radi o iskazima, oni se prvo sračunaju i njihova vrijednost se štampa.

Na primjer, ukoliko je vrijednost promjenljive A jednaka 5, a promjenljive B jednaka 8, tada iskaz:

```
PRINT ″A=″,A,=B=″,B
```

Treba da štampa dvije znakovne konstante i dvije vrijednosti brojnih promjenljivih. U ovom slučaju izraz će biti:

```
A=    5     B=    8.
```

Zarezi u iskazu naznačavaju da između elemenata koji se štampaju treba da bude određeni prazan prostor. U ovom slučaju se svaki element štampa u posebnoj zoni od po 14 znakova. Maksimalni broj zona u jednom redu je 5. očito je da je izlaz u prethodnom primjeru jako nepregledan; zato se uvodi i drugi način štampanja. Kada su elementi liste odvojeni tačka-zarezom, oni se ne štampaju u posebnim zonama, nego se jednostavno štampanje jednog elementa nastavlja na predhodni. Zbog toga bi se prethodni izlazni iskaz trebao dati u obliku:

`PRINT ″A=″L;A,″B=″;B` što bi se odštampalo kao: `A=5     B=8`

i očito bi bilo daleko preglednije.
Korektno napisani izlazni iskazi su i:

```
PRINT A$;B$
PRINT A,B,A+B,A*B
PRINT A;″JE VEĆE OD″;B
PRINT A,2*A,3*A
PRINT A,,B
PRINT A1,A2,A3,A4,A1+A2,A3*A4
```

Dva uzastopna zareza u prednjem primjeru ukazuju na to da će se prilikom štampanja jedna zona preskočiti, odnosno ostati prazna. U zadnjem primjeru definisano je 6 elemenata. Pošto se svi ne mogu štampati u jednom redu, prvih 5 se štampa u istom redu, dok se preostali šesti elemenat A3*A4 štampa u sljedećem redu, i to otpočetka. Za vježbu analizirati kako će se izvršiti i preostali primjeri izlaznih iskaza**.**

PRVI BASIC-PROGRAM

Već sa ove tri vrste osnovnih iskaza u BASIC-u mogu da se pišu programi, odnosno da se rješavaju jednostavniji problemi uz pomoć računara. Potrebno je dati još samo neke preporuke pri pisanju.

Važno pravilo pri pisanju programa je da on bude što razumljiviji i da u samom programu postoje objašnjenja o pojedinim njegovim dijelovima. Jedna od sredstava za sva pojašnjavanja je REM iskaz (od engleske riječi remark-opažanje). Ovaj iskaz omogućuje postavljanje komentara u obliku običnog teksta na bilo kom mjestu u programu. Opšti oblik REM-iskaza je:

Bn REM tekst,

Gdje je **Bn** broj reda naredbe (od 1 do 999999)
A konkretan primjer je:

```
1 REM PROGRAM ZA RAČUNANJE POVRŠINE TROUGLA
```

drugi važan iskaz označava završetak programa napisanog u BASIC-u. To je END iskaz (od engleske riječi END-kraj). Ovaj iskaz nema nikakvih dodataka kao što su promjenljive, konstante, izrazi, nego se uglavnom pojavljuje samostalno na kraju svakog programa.

Pošto se BASIC program sastoji od više iskaza raznog tipa, potrebno je utvrditi redoslijed kojim će se oni izvršavati. U tu svrhu u BASIC-u su uvedeni tzv. Brojevi linija. Svaki iskaz u programu kji se piše u jednom redu ima na samom početku reda jedan cijeli broj. Taj broj zapisuje korisnik pri pisanju iskaza i on određuje redoslijed izvršenja pojedinih iskaza. Pravilo glasi: *iskazi će se izvršavati od onih s najnižim do onih s najvišim brojem linije.* To se najbolje može vidjeti na sljedećem primjeru:

```
10 REM POČETAK
30 PRINT X
20 INPUT X
```

i pored toga što je iskaz PRINT napisan prije iskaza INPUT, ipak će se PRINT izvršiti nakon INPUT-a, jer ima veći broj linije (30), dok je kod ulaza INPUT taj broj manji (20). U slučaju da se u računar u istom programu unese više linija s istim brojem, zapamtiće se samo ona koja je unesena posljednja. Ovdje se govori o broju linija, a ne o broju iskaza, jer u jednoj liniji može da se zapiše i više BASIC-iskaza. Tada se oni odvajaju znakom dvotačka (:).

Prilikom pisanja programa preporuka je da se linijama ne pridruže uzastopni brojevi, nego treba da postoji veća razlika između brojeva susjednih linija. To su u predhodnom primjeru brojevi 10, 20, 30 (znači korakom 10). Na taj način moguće je naknadno ubacivati dodatne linije između postojećih i vršiti izmjene i poboljšanja programa.

Krenimo sada na prvi BASIC-program. Neka je potrebno za datu stranu kvadrata izračunati njegov obim i površinu. Mada je problem dosta jednostavan, ipak se u njemu mogu sagledati sve potrebne etape u rješavanju zadatka uz pomoć računara.

Kao prvo, potrebno je dobro definirati problem, a u okviru toga sagledati ulaze, izlaze i uslove. U ovom primjeru to su:

ULAZI: a – strana kvadrata,

IZLAZI: p- površina kvadrata i O – obim kvadrata,

USLOVI: $p = a^2, \quad O = 4a$.

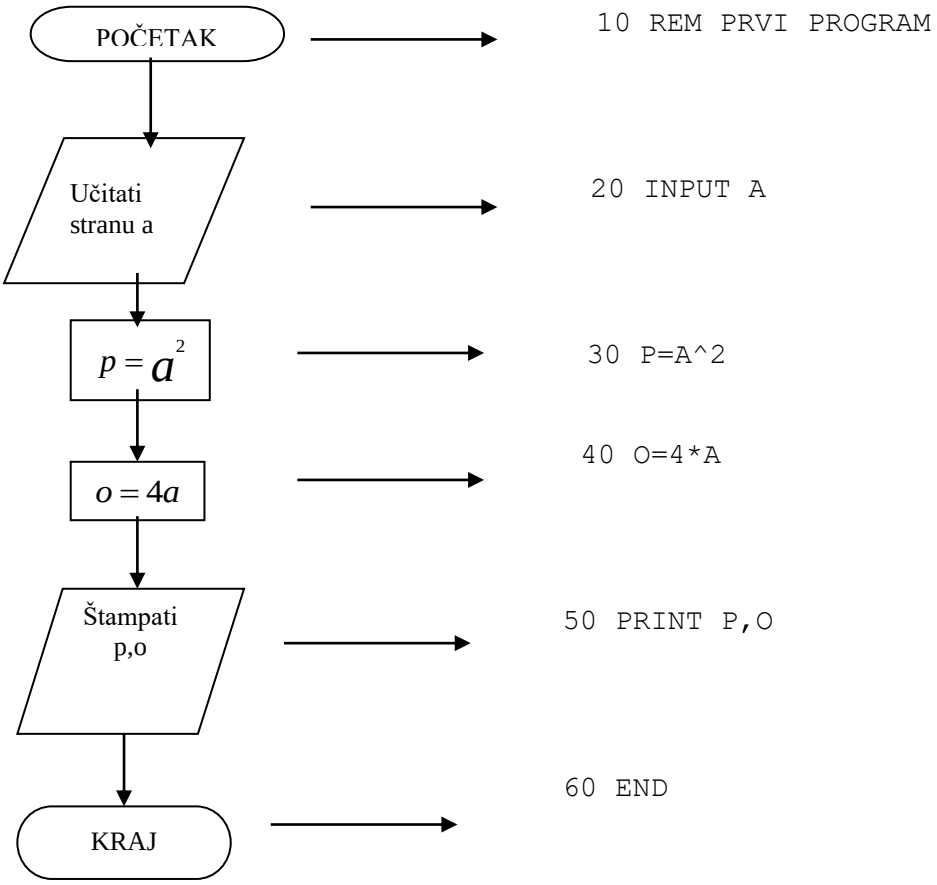
Sljedeća faza je razrada dijagrama toka. Pošto je ovo vrlo jednostavan problem, moguće je već u prvom koraku postići dovoljnu detaljnost. Taj dijagram toka dat je na slici 4a. na osnovu ovog dijagrama toka može se pristupiti programiranju u BASIC-u. Za svaki blok u dijagramu toka zapisuju se odgovarajuće BASIC-instrukcije. Konačan program dobijen ovim putem dat je na slici 4b.

Sada se može prekontrolisati dali program dobro radi. Potrebno ga je prvo pokrenuti. To se vrši naredbom RUN (od engleske riječi run-trči). Unošenje vrijednosti za stranicu ***a*** kvadrata i dobijanje rezultata, površine i obima, imat će sljedeći oblik:

```
RUN
?5
25        20
```

mora se priznati da u ovom slučaju rezultati koje daje računar nisu baš pregledni. Gledajući detaljnije, može se vidjeti da je definisana vrijednost strane a jednaka 5, izračunata površina je 25, a obim 20.

Da bi ovaj dijalog sa računarom bio pregledniji, bilo bi potrebno dodati još i određena tekstualna objašnjenja pri unošenju podataka i pri štampanju rezultata.



a) Dijagram toka b) Program u BASIC-u

slika 4a-b. Problem nalaženja površine i obima kvadrata

program sa dodatnim korekcijama ima slijedeći oblik:

```
10 REM PRVI PROGRAM
20 INPUT ˝UNESITE STRANU KVADRATA˝;A
30 P=A^2
40 O=4*A
50 PRINT ˝POVRŠINA JE=˝;P,˝OBIM JE=˝;O
60 END
```

u ovom slučaju dijalog bi bio mnogo pregledniji i pored toga što se rješava isti problem. Imao bi slijedeći oblik:

```
RUN
UNESITE STRANU KVADRATA?7
POVRŠINA JE=49        OBIM JE=28
```

Potrebno je napomenuti da za predhodne korekcije programa nije neophodno ponovo pisati cijeli program, nego je naknadno ubačen tekst iza iskaza INPUT i izmijenjena linija 50. računar automatski formira novu, ispravljenu verziju programa.

Da bi se program sačuvao i kasnije se ponovo koristio i proširivo, potrebno ga je snimiti.pohraniti na hard disk ili disketu. Naredba za pohranjivanje programa je SAVE, a naziv programa se zadaje maksimalno od 8 znakova unutar zkaka navodnika, npr. SAVE ˝PROG1˝ i pritisnuti tipku ENTER. Kasnije taj isti program kojeg smo snimili pod tim nazivom u BASIC-u prozivamo naredbom LOAD, u ovom slučaju to bi bilo: LOAD ˝PROG1˝ i tipka ENTER. Sa neredbom LIST na ekranu vidimo sadržaj programa.