

Z Á K L A D N Á Š K O L A M A R H A Ň  
UČEBNA ZÁKLADOV VÝPOČTOVEJ TECHNIKY

Z Á K L A D Y V Ý P O Č T O V E J  
T E C H N I K Y

Miroslav Vojtek

## P R E D S L O V

Milý mladý priateľ(-ka)!

Dostáva sa ti do ruky experimentálny učebný text pre predmet základy výpočtovej techniky na základnej škole. Je to nový predmet, ktorý sa na základnej škole doposiaľ nevyučoval. Avšak s rozvojom techniky a potrebou stále kvalifikovanejších pracovníkov je potrebné začať sa zaoberať výpočtovou technikou už na základnej škole. Tieto texty majú byť prvou pomôckou žiakom pri vnikaní do tajov výpočtovej techniky. Ich hlavnou úlohou je však oboznámiť všetkých žiakov so základmi mikropočítačov a ich ovládania, vzbudiť záujem o výpočtovú techniku a hľadať talenty. Pretože mikropočítačová technika je stále drahá, pri jej využívaní veľkým počtom žiakov je potrebné dodržiavať určité zásady. Tie sú súčasťou prvej kapitoly. O tom čo je vlastne mikropočítač a ako pracuje sa dočítaš v druhej kapitole. Tretia kapitola je venovaná ovládaniu mikropočítača v programovacom jazyku BASIC. Obsahuje výklad najzákladnejších slov tohto jazyka, potrebných na zostavenie jednoduchých programov - kresieb alebo výpočtových úloh napr. z fyziky. Vo štvrtej časti je načrtnutá problematika tvorby programov na mikropočítačoch. Postihuje návrh úlohy, jej rozbor a tvorbu vývojového diagramu. Poslednou učebnou časťou textov sú príklady programov a úlohy pre tvorbu jednoduchých riešení v jazyku BASIC.

Učebné texty majú ešte 5 príloh. V prvej z nich je vysvetlený význam hlásení, ktoré ti počítač pri práci oznamuje. Aby si lepšie porozumel(-a) významu jednotlivých slov programovacieho jazyka, nauč sa aj ich slovenský význam podľa prílohy 2. V prílohe 3 je podrobne popísaná klávesnica mikropočítača MATO a taktiež jej odlišnosti od klávesnice mikropočítača PMD-85. Dobrým zvládnutím tejto časti si ušetriš mnoho času a tvoja práca bude zaujímavejšia. V ostatných prílohách sú zoznamy programovacích slov jazyka BASIC, ktoré sú vysvetľovné v týchto textoch.

Tieto texty vznikli po skúsenostiach s vyučovaním nepovinného predmetu Výpočtová technika (v 6.roč.) na Základnej škole v Marhani v lete 1990 a boli doplnené o poznatky z vyučovania povinného predmetu Základy výpočtovej techniky v 6.ročníku a nepovinného predmetu Výpočtová technika v 7. a 8. ročníku na základnej škole v Marhani v školskom roku 1990/91.

a u t o r

## OBSAH:

Predslov .....	5
Obsah .....	7
1. ZÁSADY PRÁCE V UČEBNÍ ZVT .....	9
2. AKO PRACUJE MIKROPOČÍTAČ .....	9
Hlavné časti mikropočítača .....	10
Prídavné zariadenia mikropočítačov .....	10
3. PROGRAMOVACÍ JAZYK BASIC .....	11
Forma zobrazovania čísel .....	12
Aritmetické premenné .....	12
Operátory .....	12
Zápis matematických výrazov .....	13
PRIAMY REŽIM .....	13
Práca v priamom režime .....	14
PROGRAMOVÝ REŽIM .....	14
Základné príkazy a povel pre programový režim .....	15
Príkaz REM .....	15
Príkaz LET .....	15
Príkaz PRINT .....	15
Príkaz AT .....	16
Príkaz END .....	16
Príkaz RUN .....	16
Povel LIST .....	16
Povel LLIST .....	17
Povel AUTO .....	17
Povel NEW .....	17
Príkaz BEEP .....	17
Príkaz PAUSE .....	18
Príkaz STOP .....	18
Povel CONT .....	18
Príkaz GCLEAR .....	18
Zápis a oprava programov .....	18
GRAFICKÉ PŘÍKAZY .....	19
Príkaz SCALE .....	20
Príkaz MOVE .....	20
Príkaz PLOT .....	20
Príkaz AXES .....	21
Príkaz LABEL .....	21
Príkaz FILL .....	22
Príkazy cyklu .....	22
Príkazy pre údaje .....	23
Príkaz INPUT .....	23
Príkaz DATA .....	24
Príkaz READ .....	24
Príkaz RESTORE .....	25
Príkaz CLEAR .....	25
Polia .....	25
Príkaz DIM .....	26
Príkazy na riadenie programu .....	26
Príkaz GOTO .....	26
Príkaz ON GOTO .....	27
Príkaz IF THEN .....	27
Podprogramy .....	27
Príkaz GOSUB .....	28
Príkaz RETURN .....	28
Príkaz ON GOSUB .....	29
Niektoré funkcie a pomocné príkazy .....	29
Funkcia ABS(X) .....	29
Funkcia SQR(X) .....	29
Funkcia INT(X) .....	29

Funkcia RND(X) .....	30
Funkcia TAB(X) .....	30
Príkaz DISP .....	30
Príkaz ? .....	30
Funkcia INKEY .....	31
Funkcia LEN .....	31
Goniometrické funkcie .....	31
Funkcia SIN(X) .....	32
Funkcia COS(X) .....	32
Funkcia TAN(X) .....	32
Funkcia ATN(X) .....	32
PRÁCA S KAZETOPÁSKOVOU PAMÄŤOU (MAGNETOFONOM) .....	33
Povel SAVE .....	33
Povel CHECK .....	33
Povel LOAD .....	34
Povel MGLD .....	34
PRÁCA S TLAČIARŇOU .....	34
Príkaz CONTROL .....	35
Príkaz OUTPUT .....	35
Príkaz LIST# .....	35
4. TVORBA A LADENIA PROGRAMOV .....	36
Vývojové diagramy .....	36
Ladenie programov .....	38
5. PRÍKLADY A ÚLOHY .....	38
OBRAZOVÁ PRÍLOHA: 1 - Značky vývojových diagramov .....	41
OBRAZOVÁ PRÍLOHA: 2 - Vývojový diagram A .....	42
OBRAZOVÁ PRÍLOHA: 3 - Vývojový diagram B .....	43
PRÍLOHA: 1 - Chybové hlásenia .....	44
PRÍLOHA: 2 - Anglicko - Slovenský slovníček .....	46
PRÍLOHA: 3a - Klavesnica mikropočítača Maťo .....	47
PRÍLOHA: 3b - Odlišnosti klávesnic PMD - 85 a MAŤO .....	49
PRÍLOHA: 4 - Zoznam slov v Basicu .....	50
Niekoľko slov na záver .....	51
Literatura .....	52
Poznámky .....	53



## 1. ZÁSADY PRÁCE V UČEBNÍ ZÁKLADOV VÝPOČTOVÉJ TECHNIKY

Vybavenie učebne Základov výpočtovej techniky (ZVT) má zvyčajne vysokú hodnotu. Prístroje sú citlivé na nešetrnú manipuláciu. Preto pri snahe o maximálne využitie zariadenia je potrebné dodržiavať predovšetkým tieto zásady:

1. Do UZVT vstupuj zasadne iba v prezúvkach a s umytými rukami.
2. Po vstupe do UZVT zaujmi svoje alebo pridelené miesto a vykonaj zápis do záznamu o používaní techniky.
3. Techniku zapínaj iba na pokyn alebo len so súhlasom vyučujúceho.
4. Žiakom v UZVT nie je dovolené svojvoľné zapínanie, vypínanie alebo prenášanie techniky.
5. Pri zapínaní-vypínaní techniky dodržiavaj poradie:
  - a) pri zapínaní - 1.monitor (televízor)  
2.mikropočítač
  - b) pri vypínaní - 1.mikropočítač  
2.monitor (televízor)
6. Pre prácu so záznamom údajov s kazetopáskovou pamäťou a tlačiarňou si vyžiadať súhlas vyučujúceho.
7. Najcitlivejšou a zároveň najporuchovejšou časťou počítača je klávesnica. Preto klávesy stláčaj opatrne, zľahka, netlač silno, iba do zaznenia zvukového návestia.
8. Pri výklade učiva alebo pri poznámkach vyučujúceho pozorne sleduj a pri samostatnej práci nevyrušuj susedov.
9. Keď nevieš ako postupovať ďalej, zdvihni ruku a čakaj. Tým dáš pokyn vyučujúcemu, že ho žiadaš o pomoc.
10. Po zistení akejkoľvek poruchy, napríklad poškodený počítač, vytrhnutý kábel a pod., túto hneď ohlas vyučujúcemu!

## 2. AKO PRACUJE MIKROPOČÍTAČ ?

Ak chceme odpovedať na túto otázku, musíme si najprv položiť otázku inú.

### Čo je to mikropočítač ?

Je to pomerne zložité elektronické zariadenie, ktoré dokáže veľmi rýchlo počítať, kresliť, ale aj riadiť nejaké stroje (napr. šijacie, koovoobrábacie). Pracuje oveľa rýchlejšie a presnejšie ako človek a pritom nepozná únavu. Kým my napríklad spočítame dve štorciferné čísla, mikropočítač takýchto príkladov za ten istý čas vyrieši niekoľko sto alebo tisíc.

### Z čoho je zložený mikropočítač ?

Tak ako dom je zložený z tehál, tak mikropočítač je zložený z elektrotechnických súčiastok, ktoré nazývame integrované obvody. Tak ako tehly na dome sú pospájané maltou, tak aj integrované obvody v mikropočítači sú pospájané tenkou medenou fóliou, ktorú nazývame plošný spoj. Dom, aby sme v ňom mohli pohodlne bývať, je zložený z viacerých miestností. Tak isto aj počítač má svoje časti, ktoré sú nutné pre jeho činnosť.

Jednu časť integrovaných obvodov nazývame pevná alebo stála pamäť. V tejto pamäti má počítač uložený svoj pracovný poriadok, to čo má vlastne robiť keď ho zapneme. Aby sme sa s ním mohli dohovoriť, musí nám počítač rozumieť. Tak ako ľudia v rôznych krajinách majú svoje dorozumievacie jazyky, napríklad anglický, ruský, slovenský, nemecký jazyk, aj počítače majú svoje jazyky.

Tieto jazyky sa skladajú zo slov, ktoré nazývame príkazy alebo povel. Týmto potom počítač rozumie tak, že ich mu napíšeme cez klávesnicu - tá vyzera podobne ako na písacom stroji. Tak ako človek má v hlave uložený svoj naučený dorozumievací jazyk, tak aj počítač má v jednej časti integrovaných obvodov uložený svoj jazyk (presnejšie jeho prekladáč). Tento jazyk nazývame programovací jazyk.

V pevnej pamäti je uložený pracovný poriadok mikropočítača, ktorý nazývame program - MONITOR a tiež dorozumievací - programovací jazyk. U väčšiny mikropočítačov je to programovací jazyk BASIC.

Počítač po zapnutí sám nič robiť nebude, iba oznámi, že je pripravený pracovať. Musíme mu najprv nejakú úlohu zadať tak, že mu ju napíšeme vo forme niekoľkých povelov alebo príkazov, ktoré potom vykoná. Tieto povelky môže vykonávať postupne tak ako mu ich píšeme, alebo vykoná príkazy až vtedy, keď sme mu zadali všetky príkazy, ktoré sme mu chceli zadať. V prvom prípade hovoríme, že počítač pracuje v priamom režime (kalukulačkovom), v druhom prípade v programovom režime.

Príkazy a povelky sa v počítači ukladajú do ďalšej skupiny integrovaných obvodov, ktoré nazývame voľná pamäť.

Všetky matematické úkony, ktoré mikropočítač dokáže vykonať sa vykonávajú v jedinom integrovanom obvode, ktorý nazývame mikroprocesor. Ten pomocou elektrického prúdu dokáže spočítavať, odpočítavať, deliť, násobiť porovnávať a mnoho ďalších činností. Okrem toho má mikropočítač ešte riadiacu skupinu integrovaných obvodov. Tie rozhodujú o tom kde, kedy a ktoré príkazy bude plniť a kde sa bude ukladať výsledok.

Poslednou hlavnou časťou mikropočítača sú vstupno-výstupné obvody. Pomocou nich sa upravujú výsledky úloh do takej podoby, aby sme ich mohli prečítať na papieri tlačiarne alebo obrazovke monitora. Taktiež napomáhajú mikropočítaču pri zadávaní úloh. Naše slová - príkazy a povelky - menia na elektrický signál, ktorý počítač už vie spracovať.

#### HLAVNÉ ČASTI MIKROPOČÍTAČA:

1. Pamäť - pevná a voľná
2. Mikroprocesor
3. Riadiace obvody
4. Vstupno-výstupné obvody.

#### Ako teda pracuje mikropočítač ?

Nie je to veru jednoduchá otázka. Na pochopenie vnútornej činnosti mikropočítača sú potrebné obsiahle vedomosti z fyziky, matematiky, kybernetiky a hlavne elektrotechniky a teórie obvodov. Žiakom základnej školy stačí vedieť z akých častí sa mikropočítač skladá, čo tieto časti vykonávajú, a že ich činnosť je možná za pomoci elektrického prúdu. Pre užívateľov mikropočítačov je potrebné vedieť hlavne to, ako mikropočítač obsluhovať, ako s ním komunikovať a aké zariadenia sú ešte potrebné na jeho lepšie využitie.

#### PRÍDAVNÉ ZARIADENIA MIKROPOČÍTAČOV

Na kontrolu činnosti mikropočítača a pri zadávaní úloh pripájame k mikropočítaču obrazovkový monitor, alebo televízny prijímač. Ak chceme používať väčšie programy, pomocou klávesnice by zadávanie trvalo dlho, až niekoľko hodín. Preto programy ukladáme na pásku kazetového magnetofónu, ktorý v tejto funkcii nazývame kazetopásková pamäť. Ak potrebujeme výsledky úloh uchovať dlhší čas, potrebujeme si ich nejako zapísať. Aby sme ich nemuseli prepisovať z obrazovky na papier ručne, používame na to tlačiareň - ďalšie prídavné zariadenie. Tlačiareň umožňuje aj vypísať na papier celý program, ktorý sme počítaču zadali a je uložený v jeho pamäti. Časti a zariadenia, ktoré pripájame k počítaču zvonka nazývame - periférie. Okrem týchto, ktoré boli spomenuté majú počítače aj ďalšie, na-

pr. joystick - ktorým ovládame beh programu pomocou páky s tlačidlom. Myš - umožňuje kresliť rôzne obrázky a jednoduchšie ovládať programy. Svetelné pero - umožňuje kresliť pomocou počítača priamo na obrazovku monitora a nakreslené obrázky ukladať do pamäti. Ešte zaujímavejšou perifériou je scanner, ktorý umožňuje kopírovať rôzne listiny, fotografie pod. Modernejšie a väčšie mikropočítače používajú na ukladanie programov a údajov disketové jednotky a pevné disky. Práca s disketami je oveľa pohodlnejšia, rýchlejšia a spoľahlivejšia ako s magnetofónovými kazetami. S rozvojom vedy a techniky a masovejším využívaním počítačov narastá aj počet a kvalita periférnych jednotiek.

Pracovisko s mikropočítačom obsahuje tieto hlavné súčasti:

1. Mikropočítač
2. Monitor alebo televízny prijímač
3. Kazetopásková pamäť (alebo disketová jednotka)
4. Tlačiareň.

Pre prácu s počítačom sú potrebné dva druhy prostriedkov. Prvým druhom sú technické prostriedky (hardware), druhým druhom sú programové prostriedky (software). K technickým prostriedkom patria samotné časti mikropočítača a rôzne periférie. K programovým prostriedkom patria súbory príkazov, teda programy pre vykonávanie rôznych činností, nahrané na kazetu alebo disketu.

Programy sa vytvárajú v rôznych programovacích jazykoch. O tom ako zhotoviť jednoduchý program v jazyku BASIC sa dozvieme v ďalšej časti týchto textov.

### 3. PROGRAMOVACÍ JAZYK BASIC

Programovací jazyk BASIC vznikol v polovici šesťdesiatych rokov. Jeho tvorcovia si dali za úlohu navrhnuť univerzálny programovací jazyk a súčasne poskytnúť aj "neodborníkovi", ktorý nie je špecialistom - programátorom, prístupný programovací nástroj na vyjadrenie riešení svojich úloh počítačom. Názov jazyka bol pôvodne skratkou BASIC: Beginners Allpurpose Symbolic Instruction Code (všetranne použiteľný programovací jazyk pre začiatočníkov).

Basic sa často hodnotí ako jeden z najjednoduchších programovacích jazykov, ktorý sa dá veľmi ľahko naučiť. Pre svoju jednoduchosť je veľmi rozšírený a pracujú s ním skoro všetky počítače a mikropočítače. Jeho grafickú podobu, ktorá umožňuje aj kresliť rôzne obrázky majú naše mikropočítače, napr. Didaktik Alfa, PMD-85 a MAŤO.

Každý jazyk má svoju abecedu. "Abecedu" jazyka BASIC tvoria tieto znaky:

písmená - A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y  
 čísllice - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9  
 symboly - + - \* / ^ = ( ) < > \ \_ ' ! @ # \$ % & ? : " ; , .

Kombináciou týchto symbolov vytvárame slová, ktoré nazývame priказы a povely. Pri ich vytváraní platí určitá "gramatika", ktorú nazývame syntax jazyka.

Basic umožňuje dvojaký pracovný režim:

1. priamy (kalkulačkový)
2. programový

V priamom režime počítač po zadaní povelu a stlačení potvrdzovacej klávesy EOL, povel ihneď vykoná. V programovom režime pracuje tak, že najprv musíme napísať príkazy, ktoré má vykonávať. Poradie, v akom ich má vykonávať, určíme číslom pred každým príkazom. Takto vlastne vytvoríme program v BASIC-u. Potom napíšeme povel, ktorý spôsobí spustenie programu. Jednotlivé príkazy sa vykonávajú v poradí podľa rastúcich čísel riadkov, pokiaľ niektorý príkaz v programe neurčí iné poradie. Slová v basicu, používané v priamom režime nazývame povelmi a slová používané v programovom režime, pred ktoré píšeme čísla, nazývame príkazmi. Niektoré slová sú zároveň povelmi aj príkazmi.

#### FORMA ZOBRAZOVANIA ČÍSEL

Všetky čísla, ktorých absolútna hodnota je v rozsahu 0,01 až 999999 sú zobrazované zvyčajne. Všetky ostatné (väčšie i menšie) sa zobrazujú v exponenciálnom tvare:

S M.MMMM E S XX

kde S predstavuje znamienko + alebo -, pričom znamienko + sa nezobrazuje,

M je mantisa čísla,

E je skratka pre exponent a

X je exponent daného čísla

NAPRIKLAD:      - číslo 2458000                      sa zobrazí ako    2.458 E 06  
                   - číslo -49300000                      - 4.93 E 07  
                   - číslo 0.000325                      3.25 E -04  
                   - číslo -0.009328                      - 9.28 E -03

#### ARITMETICKÉ PREMENNÉ.

Aritmetické premenné v matematických výrazoch, napr. X, Y, je možné označovať až dvoma znakmi. Prvý znak musí byť písmeno a druhý písmeno alebo číslica.

Například: X, Z, Y, A, A3, X1, CD, XA, .....

Aritmetické premenné používame na uchovanie hodnôt matematického výrazu.

#### OPERÁTORY

V jazyku basic sú niektoré symboly pre matematické operácie iné ako v matematike. Uvedieme si tabuľky, kde bude symbol s jeho významom a spôsobom zápisu výrazu.

OPERÁTORY ARITMETICKÉHO VÝRAZU		
symbol	výraz	význam
+	A+B	sčítanie
-	A-B	odčítanie
*	A*B	násobenie
/	A/B	delenie
^	A^B	umocnenie A na B
znamienko +	+A	kladné číslo A
znamienko -	-A	záporné číslo A

OPERÁTORY V LOGICKÝCH VÝRAZOCH		
symbol	výraz	význam
OR	A OR B	logický súčet
AND	A AND B	logický súčin
NOT	NOT A	negácia A

#### OPERÁTORY V RELAČNÝCH VÝRAZOCH

symbol	výraz	význam
=	A=B	A je rovné B
<	A<B	A je menšie ako B
>	A>B	A je väčšie ako B
<=	A<=B	A je rovné alebo menšie ako B
>=	A>=B	A je rovné alebo väčšie ako B
0 <>	A<>B	A sa nerovná B

#### ZÁPIS MATEMATICKÝCH VÝRAZOV

Zápis matematických výrazov v basicu je trochu odlišný ako v matematike. Líši sa práve v niektorých symboloch. Pri používaní viacerých operácií v jednom výraze využívame zátvorky. Pre správny zápis výrazu musíme vedieť v akom poradí mikropočítač spracováva zadané výrazy.

Hierarchia je nasledovná:

1. exponenciálne výrazy (mocniny)
2. znamienka plus alebo minus
3. násobenie alebo delenie
4. sčítanie alebo odčítanie
5. relačné operátory
6. logické operátory

Ak majú dva operátory rovnakú úroveň spracovania, postupuje mikropočítač smerom zľava doprava. Napríklad výraz:  $20-A+B \cdot C^D$  bude mikropočítačom spracovaný v tomto poradí:

1. vypočíta  $20-A$
2. vypočíta  $C^D$
3. vynásobí B výsledkom získaným v kroku 2
4. výsledok získaný v kroku 1 pripočíta k výsledku, ktorý získa v kroku 3.

#### PRÍKLADY ZÁPISU VÝRAZOV V BASIC-u

matematický zápis:	zápis v BASIC-u:
$3x-2$	$3 \cdot X-2$
$12.12(25-4y)$	$12.12 \cdot (25-4 \cdot Y)$
$15:(43-8.5)$	$15/(43-8 \cdot 5)$
3,14	3.14
2548576,23	2.54857 E 06

V basicu u desatinných čísel nepoznáme desatinnú čiarku, ale používame desatinnú bodku, podobne ako na kalkulačke.

#### P R I A M Y R E Ž I M

Nazýva sa aj kalkulačkový. Spočíva v tom, že po zadaní povelu a po stlačení klavesu EOL sa povel ihneď vykoná. Povel zadávame cez klávesnicu. Pri písaní sa zobrazuje do riadku v dolnej časti obrazovky. Tento riadok nazývame dialógový riadok. Do tohto riadku oznamuje počítač tiež chyby, ktorých sa môžeme dopustiť pri zadávaní povelu (pozri prílohu 1). Výsledok spracovania povelu sa väčšinou zobrazuje v pracovnej časti obrazovky.

## PRÁCA V PRIAMOM REŽIME

Po zapnutí mikropočítača vypíšeme do dialógového riadku slovo BASIC a stlačíme klávesu EOL. Pri vypisovaní slova sa objavuje v dolnej časti dialógového riadka vodorovná čiarka. Nad ňu sa napíše ďalší znak, podľa toho, akú klávesu stlačíme. Túto čiarku nazývame kurzor.

Po vypísaní BASIC a stlačení EOL sa v pracovnej časti obrazovky zobrazí nápis: BASIC G/V2.0 a v dialógovom riadku sa vypíše: OK. Teraz už môžeme zadávať povel v jazyku BASIC.

Napišeme: `PRINT 2+3` a stlačíme EOL.

Na obrazovke sa v jej pracovnej časti zobrazí:

```
PRINT 2+3
5
```

`PRINT` je povelom pre vykonanie výrazu a jeho výpis na obrazovku. Keďže za ním je aritmetický výraz, vypíše sa jeho hodnota. Ak chceme deliť dve čísla, napíšeme:

`PRINT 30/5` a stlačíme EOL.

Na obrazovke sa zobrazí:

```
PRINT 30/5
6
```

Ak chceme zobraziť text, napíšeme ho za povel `PRINT` do úvodzoviek napr.: `PRINT "VELKA PRESTAVKA"` Po stlačení EOL sa zobrazí:

```
PRINT "VELKA PRESTAVKA"
VELKA PRESTAVKA
```

Pri výpočte viacerých výrazov napíšeme výrazy za sebou za povel `PRINT`: `PRINT 2+3, 3*8/4, 25-4*2`

Výrazy oddelíme čiarkou alebo bodkočiarkou. Ak ich oddelíme čiarkou, zobrazia sa na obrazovke v jednom riadku iba štyri výrazy. Ďalšie výrazy z toho istého povelu `PRINT` sa budú zobrazovať do ďalšieho riadku. Ak výrazy v povele `PRINT` oddelíme bodkočiarkou, budú sa zobrazovať za sebou tak, že bude medzi nimi dvojznaková medzera.

Poznámka: Ak sa zmýliš pri zápise výrazu, opraváš ho tak, že stláčaním kláves so šípkami posunieš kurzor tam, kde si urobil chybu a nesprávnu časť prepíšeš. Na úpravu textu slúžia špeciálne funkcie kláves. (Pozri v prílohe 3 - popis klávesnice).

## PROGRAMOVÝ REŽIM

Ak chceme vykonávať zložité výpočty alebo kresliť obrázky, využívame programový režim mikropočítača. Jednotlivé kroky riešenia programu zoradujeme za sebou určitým spôsobom. Takémuto usporiadaniu jednotlivých príkazov a výrazov podľa čísel hovoríme program. Túto činnosť - usporiadovanie príkazov do nejakej postupnosti podľa určitého premysleného postupu nazývame - programovanie.

Program môže mať niekoľko desiatok až stoviek riadkov. Každý riadok musí začínať číslom. Číslovanie riadkov sa robí zvyčajne po 10, teda prvý riadok programu má číslo 10, druhý 20, tretí 30, atď. Robí sa to preto, aby v prípade potreby sa dal program doplniť o ďalšie riadky, ktoré budú mať čísla napr.: 11, 15, 25, 26, ..., ak by sme na niečo zabudli.

O zásadách pri tvorbe programov pojednáva štvrtá kapitola. Najprv sa naučíme jednoduché príkazy a povel, aby sme mohli napísať a opraviť program v jazyku, ktorý sa učíme - v Basicu.

## ZÁKLADNÉ PRÍKAZY A POVELY PRE PROGRAMOVÝ REŽIM

### Príkaz: REM

Tvar: REM poznámka

kde poznámka je text, ktorý obsahuje ľubovoľné znaky v basicu. Používa sa na vkladanie komentárov a vysvetľujúcich poznámok do programu. Pri vykonávaní programu tento príkaz nemá význam, slúži iba pre orientáciu programátora pri prezeraní programu. Príkazom REM by mal začínať každý program. Do poznámky sa píše názov programu, meno programátora, prípadne dátum a miesto vzniku programu. Pri rozsiahlejších programoch sa v príkaze REM pomenujú jeho jednotlivé časti.

Príklad: 10 REM Program pre výpočet kvadra  
20 REM VOMSOFT, júl 1990, ZS MARHAN

### Príkaz: LET

Tvar: LET p=v

kde p je meno premennej a v je výraz, ktorého hodnota sa má priradiť premennej p. Príkaz LET je nepovinný, môžeme ho aj vynechať.

Príklad: 120 LET X1=(Y/Z)-100  
na riadku 120 sa premennej X1 priradí hodnota aritmetického výrazu (Y/Z)-100. Rovnaký účinok bude mať aj tento tvar:  
120 X=(Y/Z)-100

Ak chceme premennej priradiť nie hodnotu výrazu ale text, potom za názov premennej napíšeme znak \$ a text napíšeme do úvodzoviek. Premennú so znakom \$ nazývame reťazcová premenná a text v úvodzovkách reťazec znakov.

Príklad: 100 X\$="MIKROPOCITAC MATO"

### Príkaz: PRINT

Tvar: PRINT zoznam údajov

kde zoznam údajov je súbor údajov, ktoré chceme vypísať na pracovnú časť obrazovky. Jednotlivé údaje oddeľujeme od seba čiarkou alebo bodkočiarkou. Riadok PRINTu je rozdelený na štyri štrnásťznakové zóny. Čiarka medzi výrazmi predpisuje prechod na nasledujúcu zónu. V riadku sa teda zobrazia iba štyri údaje zo zoznamu. Ďalšie sa zobrazia v nasledujúcom riadku. V jednom riadku sa príkazom PRINT zobrazí iba 48 znakov. Ak je oddeľovačom znakov nie čiarka ale bodkočiarka, ide o zhustené zobrazovanie vedľa seba. Medzi výrazmi je potom dvojznaková medzera.

Príklad: 10 PRINT 2, 15, 4\*8, 93/4  
20 PRINT 2; 15; 4\*8; 140; 3.23

Ak sa zaplní celý pracovný priestor obrazovky, ďalšie výrazy sa zobrazujú na miesto posledného riadka a predchádzajúce riadky sa posunú o jeden vyššie. Ak napíšeme iba PRINT bez zoznamu údajov, vynechá sa pri zobrazovaní jeden riadok.



### Príkaz: AT

Tvar: PRINT AT r,s

kde r je číslo riadku (môže byť 0 až 26)  
s je číslo stĺpca (môže byť 0 až 47)

Príkaz určuje miesto výpisu na obrazovke v príkaze PRINT. Používame ho v spojení s príkazom PRINT. Pomocou príkazu PRINT môžeme vypísať na obrazovke v riadku 48 znakov a takýchto riadkov môže byť 27. Naraz teda môžeme zobrazíť v pracovnej časti obrazovky 1296 znakov. Číslo 0,0 na mieste r,s znamenajú ľavý horný roh obrazovky. Riadky v príkaze AT čísloujeme zhora nahor, stĺpce zľava doprava. Číslami v príkaze AT určujeme začiatok výpisu výrazu(jeho prvého znaku).

Príklad: 30 PRINT AT 26,0;"CIRKUS"

Po spracovaní tohto programového riadka sa v ľavom dolnom rohu pracovnej časti obrazovky vypíše slovo CIRKUS.

### Príkaz: END

Tvar: END

Označuje koniec programu. Keď sa vykoná, program ukončí svoju činnosť. Nie je to povinný príkaz. Ak v programe chýba, vykonávanie programu sa ukončí po spracovaní programového riadka s najvyšším číslom.

Príklad: 200 END

### Povel: RUN

Tvar: RUN cr

kde cr znamená číslo riadka kde začína program. Je to povel na spustenie programu uloženého v pamäti počítača. Ak neuvedieme číslo riadku, spustí sa program od riadku s najnižším číslom v pamäti.

### Povel: LIST

Tvar: LIST cr

kde cr je číslo programového riadka. Je to povel na výpis programu v BASICu uloženého v pamäti počítača, na obrazovku. Jeho odoslanie spôsobí to, že sa program začne postupne zobrazovať na obrazovke, až sa celý zobrazí. Ak je dlhší ako pracovná časť obrazovky, "roluje" sa zdola nahor. Ak chceme výpis programu prerušiť, aby sme holi niektorú časť dôkladnejšie prezrieť, pridržíme klávesu SHIFT. Po jej uvoľnení pokračuje výpis programu ďalej. Ak chceme výpis prerušiť a nechceme v ňom pokračovať ďalej, stlačíme klávesu STOP. Výpis programu môžeme obnoviť od ľubovoľného riadku povelom: LIST cr, kde cr je číslo riadku, od ktorého chceme program vypísať.



#### Povel: LLIST

Tvar: LLIST cr

Je to povel na opravu riadku v programe, cr je číslo riadku, ktorý chceme opraviť. Po jeho odoslaní sa programový riadok s číslom cr vypíše do dialógového riadku obrazovky. Pod prvým znakom dialógového riadku sa objaví kurzor. Ak chceme opraviť niektorý znak, nastavíme najprv kurzor pomocou kláves so šípkami pod znak, ktorý chceme opraviť a tento znak prepíšeme. Po vykonaní všetkých opráv v riadku odošleme opravený riadok do pamäte stlačením klávesy EOL. Tento spôsob opravy používame napr. v prípade "preklepov", pri vkladaní nového riadka, pri vložení chybného príkazu a pod.

#### Povel: AUTO

Tvar: AUTO

Je povelom na automatické číslovanie riadkov. Po jeho odoslaní sa nastaví číslo prvého riadku na 10 a čísla riadkov sa budú po stlačení EOL zväčšovať s krokom 10. Číslo druhého riadku bude 20, tretieho 30, atď. Ak ho použijeme pri písaní programu, napr. po oprave riadku povelom LLIST, vypíše sa automaticky číslo ďalšieho programového riadku, ktoré bude o 10 väčšie ako predchádzajúce.

Tvar: AUTO m,n

kde *m* a *n* sú celé kladné čísla zabezpečí, že sa automatické číslovanie vkladanych riadkov nastaví od čísla *m* s krokom *n*.

Príklad: AUTO 200,5 (stlačenie EOL)

Po vložení tohto povelu bude mať prvý vkladany riadok číslo 200, druhý číslo 205, tretí 210, atď.

Poznámka: povel zrušíme tak, že nastavíme kurzor na začiatok riadku a napíšeme povel LIST, ktorý vypíše vložený program.

#### Povel: NEW

Tvar: NEW

Je povelom na vymazanie obsahu pamäte počítača. Používame ho pred písaním nového programu, lebo spôsobuje vymazanie starého programu z pamäte mikropočítača.

#### Príkaz: BEEP

Tvar: BEEP

Používame ho na spustenie zvukového signálu (tzv. pípnutia). Môžeme ho umiestniť kdekoľvek v programe, keď chceme upozorniť na nejakú dôležitú skutočnosť, napr. nesprávne vložený údaj, ukončenie výpočtu, zadávanie údajov a pod.

Príklad: 120 BEEP

### Príkaz: PAUSE

Tvar: PAUSE n

kde n je číselná premenná, konštanta alebo aritmetický výraz. Používame ho na pozastavenie vykonávania programu na vopred stanovený čas. Dĺžku pozastavenia určuje hodnota n, ktorá môže byť z intervalu od 0 do 255. To zodpovedá času pozastavenia od 0 sekúnd do 25,5 sekundy. V prípade, že parameter n v príkaze PAUSE chýba, preruší sa spracovanie programu na 25,5 sekundy. Po uplynutí nastavenej doby program pokračuje automaticky na ďalšom riadku.

Príklad: 40 PAUSE 50

Tento programový riadok vyvolá pozastavenie vykonávania programu na 5 sekúnd. Číslo 1 v príkaze PAUSE znamená čas 1 sekundu.

### Príkaz: STOP

Tvar: STOP

používa sa na prerušenie vykonávania programu. Po jeho spracovaní sa v dialógovom riadku obrazovky vypíše správa:

+ Stop at line cr +

kde cr je číslo riadka s príkazom STOP, ktorý spôsobil prerušenie. Príkaz používame hlavne vtedy, keď potrebujeme v dlhších programoch hľadať logickú alebo inú chybu.

Príklad: 40 STOP

### Povel: CONT

Tvar: CONT

Používame ho pri prerušení programu príkazom STOP. Po odoslaní povelu CONT program pokračuje ďalej od miesta, kde bol prerušený.

Poznámka: Každý povel používame tak, že napíšeme do dialógového riadka jeho tvar a stlačíme kláves EOL.

### Príkaz: GCLEAR

Tvar: GCLEAR

Je to príkaz na vymazanie obrazovky. Pritom program uložený v pamäti počítača sa uchováva. Používa sa v priamom aj programovom režime. Mal by sa používať na začiatku programu, hneď po úvodných poznámkach, aby sa vymazali zvyšky starého programu na obrazovke.

### **ZÁPIS A OPRAVA PROGRAMOV**

Po naučení niekoľko dôležitých príkazov a povelov skúsime zapísať jednoduchý program. Napíšeme program na delenie dvoch čísel, ktoré budú zadané v programe.

```
10 REM DELENIE DVOCH CISEL
20 GCLEAR
30 A=25
40 B=4
```

```

50 C=A/B
60 PRINT C
70 END

```

Číslovanie riadkov si zabezpečíme povelom AUTO. Po jeho odoslani stlačením klávasy EOL sa v dialógovom riadku zobrazí číslo 10, za ním jedna medzera a kurzor. Môžeme teda písať prvý riadok programu. Keď ho napíšeme celý, stlačíme EOL. Týmto sa riadok zapíše do pamäti počítača a jeho obsah sa tiež zobrazí v pracovnej časti obrazovky. Ak by sme sa pri písaní pomylili, vrátime kurzor späť pomocou kláves so šípkami a chybný údaj prepíšeme. Keď zistíme, že máme chybu v riadku, ktorý sme už odoslali do pamäte, použijeme povel LLIST, za ktorým napíšeme číslo riadku s chybou a stlačíme EOL. Tento sa zobrazí v dialógovom riadku, opravíme ho a odošleme stlačením klávesy EOL. Keď sme už napísali celý program, nastavíme kurzor na začiatok riadku a napíšeme povel LIST. Po stlačení EOL sa vypíše program na obrazovku tak, ako je uložený v pamäti. Tým sa presvedčíme, že program je naozaj uložený v pamäti mikropočítača. Program spustíme povelom RUN a stlačením EOL.

Ako pracuje popisovaný program ?

V riadku 10 je poznámka - názov programu, ten si počítač nevšima. Po spracovaní riadku 20 sa vymaže obrazovka. V riadku 30 priradí premennej s menom A číslo 25 a v riadku 40 priradí premennej s menom B číslo 4. Potom v riadku 50 priradí premennej C výsledok delenia A/B a v riadku 60 vypíše na obrazovku hodnotu premennej C. Po spracovaní riadku 70 ukončí svoju činnosť a v dialógovom riadku vypíše správu OK. Po vykonaní programu je v pracovnej časti obrazovky iba jedno číslo - výsledok delenia.

Aby bolo jasné odkiaľ sme dostali číslo vypísané na obrazovke, upravujeme v programe výpis výsledkov na obrazovku takto:

```

10 REM DELENIE DVOCH CISEL
20 GCLEAR
30 A=25
40 B=4
50 C=A/B
60 PRINT "A=25"; B=4"
70 PRINT "A/B= "C
80 END

```

#### GRAFICKÉ PRÍKAZY

Jazyk BASIC G má pre kreslenie obrázkov rozdelenú pracovnú časť obrazovky na 256x243 bodov. Pre kreslenie používame tieto príkazy:

SCALE	nastavenie mierky zobrazenia - rozdelenie obrazovky
MOVE	nastavenie bodu pre začiatok kreslenia
PLOT	vykreslenie bodu alebo úsečky
AXES	kreslenie súradnicových osí
LABEL	vykresľovanie a zväčšovanie znakov
FILL	kreslenie plných obdĺžnikov a štvorcov

### Príkaz: SCALE

Tvar: SCALE x1,x2,y1,y2

kde x1 je dolná hranica nášho súradnicového syst. na vodorovnej osi  
x2 je horná hranica nášho súradnicového syst. na vodorovnej osi  
y1 je dolná hranica nášho súradnicového systému na zvislej osi  
y2 je horná hranica nášho súradnicového systému na zvislej osi

Príkaz oznamuje počítaču, akú mierku má používať pri kreslení na obrazovke. Používame ho stále na začiatku grafického programu, pred ostatnými grafickými príkazmi. Bez príkazu SCALE sú ostatné grafické príkazy neúčinné.

Príklad: 20 SCALE 0,255,0,242

Tento riadok spôsobí to, že body na vodorovnej osi sú očíslované zľava doprava od 0 do 255 a na zvislej osi zdola nahor od 0 do 242. Súradnice bodov, používané v tomto programovom riadku sa zhodujú so súradnicami grafického formátu obrazovky.

Príklad: 20 SCALE 0,210,0,170

Toto rozdelenie zodpovedá približne veľkosti pracovného priestoru obrazovky v milimetroch, ak používame monitor PMD-60.1, alebo televízny prijímač MERKUR. Pri tejto mierke sa veľmi ľahko programujú rôzne obrázky, mapy a pod., ktoré máme nakreslené na milimetrovom papieri v obdĺžniku 210x170 milimetrov. Príkaz SCALE môže mať aj iné mierky, napr.:

20 SCALE 0,100,0,100  
20 SCALE -10,10,-10,10

### Príkaz: MOVE

Tvar: MOVE x,y

kde x je vodorovná súradnica požadovaného bodu (podľa mierky SCALE)  
y je zvislá súradnica požadovaného bodu

Príkaz nastavuje začiatok kreslenia. Za x a y môžeme dosadiť aj aritmetické výrazy. V bode x,y sa začne kresliť podľa nasledujúcich grafických príkazov.

Príklad: 20 SCALE 0,100,0,100  
30 MOVE 0,0

Kresliť sa začne v ľavom dolnom rohu pracovnej časti obrazovky.

30 MOVE 50,50

Kresliť sa začne v strede obrazovky.

### Príkaz: PLOT

Tvar: PLOT x,y,k

kde x je vodorovná súradnica bodu, ktorý chceme zobraziť

y je zvislá súradnica tohto bodu

k je parameter spôsobu zobrazenia - ak k = 1 zobrazí sa iba bod (x, y)

Ak  $k < 1$  alebo nie je uvedené, zobrazí sa bod  $(x,y)$  spolu s úsečkou, spájajúcou ho s posledným bodom, v ktorom sme naposledy prestali kresliť, alebo s bodom  $(x,y)$  v príkaze MOVE, napísanom pred príkazom PLOT. Súradnice  $x,y$  v príkaze PLOT udávajú v mierke zadanej príkazom SCALE. Zjednodušené - príkaz PLOT kreslí čiaru od bodu  $x,y$  v príkaze MOVE do bodu  $x,y$  v príkaze PLOT.

Príklad: 100 GCLEAR  
110 SCALE 0,100,0,100  
120 MOVE 0,0  
130 PLOT 100,100

Táto časť programu nakreslí čiaru z ľavého dolného rohu pracovnej časti obrazovky do pravého horného rohu pracovnej časti obrazovky.

140 MOVE 0,0  
150 PLOT 100,0

Tieto dva riadky spôsobia vykreslenie čiar z ľavého dolného do pravého dolného rohu pracovnej časti obrazovky.

160 MOVE 0,50  
170 PLOT 100,50

Po spracovaní týchto riadkov sa vykreslí vodorovná čiara cez stred obrazovky.

Poznámka: Ak chceme kresliť viacero čiar za sebou, potom stačí ak zadáme ich koncové body v príkaze PLOT v tvare: PLOT  $x_1,y_1$ ;  $x_2,y_2$ ;  $x_3,y_3$ ;  $x_4,y_4$ ; .... a súradnice bodov oddelíme bodkočiarkou.

Príklad: 50 GCLEAR  
60 SCALE 0,210,0,170  
70 MOVE 0,0  
80 PLOT 210,0;210,170;0,170;0,0

Táto časť programu spôsobí vykreslenie obdĺžnika s rozmermi 210x170 milimetrov (orámuje vlastne pracovnú časť obrazovky).

#### Príkaz: AXES

Tvar: AXES  $x,y$

kde  $x$  je vodorovná súradnica a  $y$  je zvislá súradnica bodu, v ktorom sa majú pretínať súradnicové osi. Súradnice  $x,y$  sa zadávajú v mierke podľa príkazu SCALE. Pri spracovaní príkazu AXES sa najprv vykreslí vodorovná súradnicová os (zľava doprava) a potom zvislá os (zhora nadol). Pri zadaní ďalšieho grafického príkazu by kreslenie pokračovalo na dolnom konci zvislej súradnicovej osi.

Príklad: 40 AXES 0,0  
Nakreslí súradnicové osi prechádzajúce ľavým dolným rohom pracovnej časti obrazovky.  
50 SCALE 0,100,0,100  
60 AXES 50,50  
Nakreslí súradnicové osi prechádzajúce podľa zadanej mierky stredom obrazovky.

#### Príkaz: LABEL

Tvar: LABEL  $z_1,z_2$ ; zoznam

kde  $z_1$  je zväčšenie znakov vo vodorovnom smere (podľa mierky SCALE)  
 $z_2$  je zväčšenie znakov v zvislom smere  
zoznam je zoznamom znakov, ktoré sa majú príkazom LABEL zväčšiť

Ak chceme zväčšovať text, uvedieme ho do úvodzoviek. Základný formát zobrazenia jedného znaku (číslica, písmeno alebo znamienko) je 6x8 bodov. Jeden znak zaberá na obrazovke miesto šírky 6 bodov a výšky 8 bodov. Prikazom LABEL zobrazujeme znaky vo formáte, ktorý je celočíselným násobkom tohto základného formátu (8x6bodov), pričom môžeme programovať rôzne zväčšenia formátu vo vodorovnom a zvislom smere. Zjednodušene - znak napríklad číslo 8 môžeme zväčšiť 3x, 5x, 20x do šírky aj do výšky. Miesto, kde sa majú tieto zväčšené znaky na obrazovke vypisovať určujeme prikazom MOVE x,y , v súradniciach zadanych prikazom SCALE.

Priklad: 100 GCLEAR  
110 SCALE 0,10,0,10  
120 MOVE 0,4  
130 LABEL 3,6;"AHOJ"

Táto časť programu po spracovaní vypíše text AHOJ hneď zľava a približne v strede výšky obrazovky, zväčšený vodorovne 3x a zvisle 6x. Každý znak zaberá teraz už plochu nie 8x6, ale 48x18 bodov. Zmenou parametrov x1 a x2 v prikaze LABEL môžeme meniť výšku a šírku zobrazovaných znakov.

#### Prikaz: FILL

Tvar: FILL x,y;maska

Prikaz FILL kreslí plošné obdĺžniky a štvorce. Číslo x je počet bodov vo vodorovnom smere a číslo y je počet bodov v zvislom smere vykresľovaného obdĺžnika. Na mieste maska pre jednoduchosť budeme používať číslo 1 (iné čísla umožňujú kreslenie viac obrázkov a pod.). Pred prikazom FILL používame prikaz MOVE, ktorý určí, kde bude ľavý dolný roh vykresľovaného obdĺžnika.

Priklad: 50 GCLEAR  
60 SCALE 0,100,0,100  
70 MOVE 50,50  
80 FILL 20,60;1

Táto časť programu vykreslí plošný obdĺžnik šírky 20 bodov a výšky 60 bodov, pričom ľavý dolný roh obdĺžnika bude podľa prikazu MOVE v strede pracovnej časti obrazovky.

Pozor! V prikaze MOVE, ktorým nastavujeme bod začiatku kreslenia, používame súradnice v mierke zadanej prikazom SCALE. V prikaze FILL bez ohľadu na mierku v prikaze SCALE uvádzame rozsah kreslenia vždy vo formáte 256x243 bodov (ako keby bolo SCALE0,255,0,242).

#### PRÍKAZY CYKLU

Pri tvorbe programu väčšinou zistíme, že niektoré časti by sa mnohonásobne opakovali. Aby sme ich nemuseli zakaždým písať, použijeme príkazy, ktoré nami zvolenú časť programu zopakujú toľkokrát, koľko potrebujeme. Potom takúto opakujúcu sa časť programu nazývame cyklus.

Na vytvorenie cyklu použijeme dvojicu príkazov FOR TO STEP a NEXT. Cyklus začíname príkazmi FOR TO STEP a ukončíme príkazom NEXT.

Tvar: FOR p=v1 TO v2 STEP v3  
kde p je riadiaca premenná cyklu  
v1 je aritmetický výraz,určujúci začiatočnú hodnotu premennej p  
v2 je aritmetický výraz, určujúci koncovú hodnotu premennej p  
v3 je aritmetický výraz, ktorý určuje krok zmeny premennej p

Ak krok (STEP)  $vj=1$ , možno časť STEP v začiatku cyklu vynechať. Za príkazom FOR TO STEP nasleduje telo cyklu, t.j. časť programu, ktorá sa má opakovať. Nakonci tejto časti je príkaz, ktorý ukončuje cyklus: NEXT.

Tvar: NEXT p

kde p je názov riadiacej premennej cyklu, použitej v začiatočnom príkaze cyklu FOR.

V programe cyklus vyzerá napr. takto:

```
50 ...
60 FOR I=1 TO 5
    .....
    (telo cyklu)
    .....
130 NEXT I
140 ...
```

V tomto cykle sa telo bude opakovať 5-krát. Ak príde program na riadok 60, zistí, že riadiaca premenná I sa bude meniť od 1 do 5 s krokom 1 (lebo STEP chýba). Riadiacej premennej I priradí hodnotu jedna a vykoná všetky príkazy až po riadok 130.

V tomto riadku zisťuje, či už má skončiť cyklus, t.j. či  $I=5$  (koncovej hodnote cyklu). Keďže I ešte nie je rovné 5, zvýši hodnotu riadiacej premennej o krok, teda o 1 a nasmeruje vykonávanie programu na riadok za príkazom FOR.

Vykoná sa znova telo cyklu, zisťuje sa či  $I=5$ , zväčší sa I o 1 (teraz je už  $I=3$ ), atď... Keď už bude  $I=5$  (cyklus sa vykonal 5-krát), bude vykonávanie programu pokračovať na riadku nasledujúcom za príkazom NEXT, teda na riadku s číslom 140.

Príklad: 

```
50 FOR K=1 TO 10
60 PRINT K, 5*K
70 NEXT K
```

Spracovaním tejto časti sa do desiatich riadkov pod seba, do dvoch stĺpcov vypíšu čísla tak, že v ľavom stĺpci bude číslo a v pravom jeho päťnásobok. Keby sme nepoužívali cyklus, museli by sme riadok s číslom 60 (kde namiesto K by sme písali čísla od 1 do 10) opakovať v programe 10-krát. Použitím cyklu sme ušetrili 7 riadkov programu.

Príklad: 

```
100 FOR L=1 TO 20
110 BEEP
120 NEXT L
```

Táto časť programu spustí 20-krát zvukové návestie (pípnutie).

#### PRÍKAZY PRE ÚDAJE

Programy pri svojej činnosti potrebujú rôzne číselné a textové údaje (data), ktoré majú spracovávať. Tieto do programu vstupujú pomocou nasledujúcich príkazov.

#### Príkaz: INPUT

Tvar: INPUT zoznam

Je to príkaz pre zadávanie údajov z klávesnice mikropočítača. Zoznam, obsahuje zoznam premenných, ktorým cheme priradiť hodnoty vkladané z klávesnice. Používame ho na zadávanie údajov pri vykonávaní programu. Pri spracovaní príkazu INPUT sa vykonávanie programu preruší, ozve sa

zvukový signál a počítač čaká na vloženie údajov. Tie vkladáme z klávesnice do dialógového riadku obrazovky v poradí, ktorý zodpovedá zoznamu v príkaze INPUT. Vložené údaje odošleme do pamäte počítača stlačením EOL. Po tomto ťkone program pokračuje ďalej už so spracovávaním nami zadávaných údajov. Pred príkazom INPUT je vhodné použiť príkaz DISP alebo ?, kde napíšeme, aké údaje žiada príkaz INPUT.

Príklad: 60 DISP"Zadaj stany obdĺznika A,B a stlač EOL"  
70 INPUT A, B

Do dialógového riadku sa vypíše oznam v príkaze DISP a počítač čaká na zadanie hodnôt A, B. Napíšeme hodnotu A napr. 8.25, za ňu dáme čiarku a napíšeme hodnotu B napr. 32.5. Po napísaní oboch hodnôt stlačíme EOL. Program tým pokračuje ďalej vo svojej činnosti. Ak v príkaze INPUT zadávame viacero hodnôt, oddeľujeme ich od seba čiarkou.

#### Zadávanie údajov zo zoznamu v programe.

Údaje možno uchovávať priamo v programe, v tzv. zoznamoch konštánt alebo textov, zadávaných v príkazoch DATA. Hodnoty konštánt z týchto zoznamov možno priradovať premenným príkazmi READ. Miesto v zozname, od ktorého sa majú čítať konštanty príkazom READ možno nastaviť pomocou príkazu RESTORE.

#### Príkaz: DATA

Tvar: DATA zoznam

Používa sa na ukladanie zoznamu konštánt do programu. Zoznam, je zoznam číselných a textových konštánt, oddelených čiarkami. Príkazy DATA môžu byť umiestnené kdekoľvek v programe. Pre väčšiu prehľadnosť ich však píšeme na konci programu za príkazom END.

Príklad: 300 DATA LUBICA, JANA, PETER, 19, 4, 1, 8  
310 DATA PRIEZVISKO, MENO, DATUM, ROK, 20, X1, 29  
320 DATA 41, 4187, 3.015, 62.1

V tom istom príkaze DATA môžeme zapisovať číselné aj textové údaje. Pri zápise textov v zozname DATA je nutné uvádzať texty do úvodzoviek iba vtedy, ak text obsahuje medzery, čiarky alebo dvojbodky.

#### Príkaz: READ

Tvar: READ zoznam

Príkaz READ priraduje premenným zo svojho zoznamu hodnoty konštánt v príkaze DATA. Môžu to byť čísel. aj text. premenné, oddelené čiarkami.

Príklad: 200 READ A, C1, M, N

Tento riadok v programe pri jeho vykonávaní spôsobí to, že zo zoznamu DATA sa načítajú prvé štyri hodnoty čísel a priradia sa premenným A, C1, M, a N.

Príkazy READ a DATA v programe spolupracujú takto: Konštanty zo všetkých príkazov DATA v danom programe tvoria akoby jeden súvislý zoznam v tom poradí, v akom sú v programe uložené príkazy DATA. Príkazy READ postupne priradzujú premenným hodnoty konštánt z tohto zoznamu v tom poradí, v akom sú vykonávané (prvý príkaz READ číta konštanty zo začiatku zoznamu, druhý príkaz READ číta konštanty zo zoznamu od miesta,



v ktorom skončil prvý príkaz READ, atď.). Vzájomne si zodpovedajúce premenné zo zoznamu príkazov READ a konštanty zo zoznamu z príkazov DATA musia byť rovnakého typu. Číselné konštanty priradujeme číselným premenným a textové konštanty textovým premenným.

Príklad: 100 READ M, N, C1\$, C2\$, A, B  
          .....  
          300 DATA 24.1, 8, LUBICA, DANA, 1.21, 41

#### Príkaz: RESTORE

Tvar: RESTORE cr

Tento príkaz u možňuje opakované čítanie konštánt zo zoznamu v príkaze DATA. Umožňuje tiež zmenu poradia čítania údajov, ak uvedieme číslo riadku, od ktorého sa majú čítať. Ak príkaz použijeme v tvare: RESTORE t.j. bez čísla riadku, budú sa údaje čítať hoci aj vo štvrtom príkaze READ od začiatku zoznamu v bloku DATA.

Príklad: 200 RESTORE  
Príklad: 300 RESTORE 500

Bude sa čítať zo zoznamu DATA, ktorý začína v riadku 500.

#### Príkaz: CLEAR

Tvar: CLEAR

Je príkazom na vynulovanie všetkých premenných a polí, vyskytujúcich sa v programe.

### POLIA

Premenné zapísané iba menom (napr. A, B, X1, Y) nazývame jednoduchými premennými. Pri programovaní zložitejších úloh, v ktorých pracujeme s veľkými skupinami premenných, by sme s jednoduchými premennými nevysťahčili. Napríklad na označenie všetkých prvkov tabuľky, ktorá má 50 riadkov a 50 stĺpcov, by sme potrebovali 2500 rôznych názvov premenných. Programy na spracovanie údajov takejto tabuľky by boli príliš zdlhavé a vo väčšine prípadov by sa nezmestili do pamäte počítača. Na prácu s väčšími skupinami údajov rovnakého typu používame tzv. polia a indexované premenné.

Poľom, nazývame usporiadanú skupinu premenných s rovnakým menom, ale rôznym indexom. V basicu na mikropočítačoch PMD-85 a MA70 môžeme používať jednorozmerné, dvojrozmerné a trojrozmerné polia. Na prvky poľa sa v programe odvolávame menom poľa s uvedením jedného, dvoch, alebo troch indexov (čiže celých čísel, určujúcich polohu prvku v poli), oddelených navzájom čiarkami a uzavretých v okrúhlych zátvorkách. Prvky polí nazývame preto aj indexovanými premennými. V jednotlivých rozmeroch ich číslujeme od nuly. Napríklad jednorozmerné pole s menom A, ktoré obsahuje 5 prvkov, si môžeme predstaviť ako riadok: A(0) A(1) A(2) A(3) A(4).

Dvojrozmerné pole 9 prvkov s menom B si môžeme predstaviť ako tabuľku:

B(0,0)	B(0,1)	B(0,2)
B(1,0)	B(1,1)	B(1,2)
B(2,0)	B(2,1)	B(2,2)

Trojrozmerné pole si môžeme predstaviť ako skupinu tabuliek, kde tretí rozmer vždy udáva poradové číslo tabuľky.

Podobne ako jednoduché premenné, aj polia môžu byť dvoch typov: číselné a textové. Typ poľa je určený jeho menom, vytváraný podľa rovnakých pravidiel ako mená jednoduchých číselných a textových premenných a pridaním indexu do okrúhlych zátvoriek.

#### Prikaz: DIM

Tvar: DIM zoznam

DIM je príkazom na oznámenie počítaču, že budeme v programe pracovať s poľami. Príkaz DIM rezervuje v pamäti počítača miesto pre tieto polia.

Zoznam obsahuje mená a rozsah polí, oddelených čiarkami. Hovoríme potom o deklarovani (pomenovaní) polí.

Priklad: 20 DIM A(20), B(15,5), T\$(60)

Tento programový riadok deklaruje (pomenováva) jednorozmerné číselné pole A, ktoré rezervuje miesto pre 21 premenných, dvojrozmerné pole B, ktoré rezervuje miesto pre 96 premenných (16 riadkov a 6 stĺpcov) a jednorozmerné reťazcové pole T\$, ktoré rezervuje miesto v pamäti pre 61 textových premenných (napr. mien športovcov alebo žiakov).

Poznámka: Príkaz DIM na deklaráciu polí používame v úvodnej časti programu, vždy pred prvým použitím týchto premenných v programe. Ak by sme to neurobili počítač by to oznámil v dialógovom riadku ako chybu.

#### PRÍKAZY NA RIADENIE PROGRAMU

Sú to príkazy, ktoré môžu spôsobiť to, že programové riadky sa nebudú spracovávať podľa narastajúcich čísel, ale podľa týchto príkazov. V nasledujúcich riadkoch budú popísané tieto príkazy:

GOTO	príkaz skoku
IF THEN	príkaz podmieneného skoku
ON GOTO	prepínač skokov

K príkazom na riadenie programu patria aj príkazy cyklu, podprogramov a prepínač podprogramov. Patria sem aj niektoré špeciálne funkcie, napr. funkcia INKEY (pozri ďalej).

#### Prikaz: GOTO

Tvar: GOTO cr

kde cr je číslo riadku, kde má pokračovať spracovanie programu.

Priklad: 10 REM RIESENIE ROVNICE

```
.....  
70 GOTO 10  
80 PRINT X=A/B
```

Po spracovaní riadku 70, nebude nasledovať spracovanie 80-tého riadku, ale bude sa vykonávať program znova od riadku s číslom 10.

### Prikaz: ON GOTO

Tvar: ON v GOTO cr1, cr2, cr3, ....., crN

v je aritmetický výraz, ktorý nadobúda hodnoty 1, 2, 3, .....N  
cr sú čísla riadkov programu, kde sa má pokračovať.

Prikaz umožňuje vielnásobné vetvenie programu. Ak bude hodnota výrazu v=1, bude vykonávanie programu pokračovať na riadku s číslom bezprostredne uvedeným za prikazom GOTO, teda na cr1. Ak bude hodnota výrazu v=2, bude sa pokračovať na riadku cr2, ak bude v=3, bude pokračovať na riadku s číslom cr, atď.

Príklad: 50 ON Z GOTO 60, 80, 100, 120, 140  
60 PRINT "VYBORNÝ"  
70 END  
80 PRINT "CHVALITEBNÝ"  
90 END  
100 PRINT "DOBRY"  
110 END  
120 PRINT "DOSTATOCNY"  
130 END  
140 PRINT "NEDOSTATOCNY"  
150 END

Táto časť programu využíva prepínač skoku na vypísanie známky slovom. Hodnota známky je uložená v premennej Z. Program po vypísaní hodnoty známky slovom ukončí svoju činnosť. Ak je Z=1, pokračuje na riadku 60, ak Z=2 na riadku 80, ak Z=3 na riadku 100, atď.

### Prikaz: IF THEN

Tvar: IF pod THEN cr ' alebo IF pod THEN prikaz

kde pod je nejaká podmienka, napr. aritmetická relácia alebo logický výraz (napr. A>0).  
cr je číslo riadka, ktorý sa má ihneď spracovať v prípade, že podmienka je splnená,  
prikaz je prikaz, ktorý sa má vykonať keď je splnená podmienka

Ak je podmienka v prikaze IF THEN splnená, potom sa vykoná skok na riadok s číslom cr, alebo sa vykoná prikaz uvedený za slovom THEN. Ak podmienka nie je splnená, spracovanie programu pokračuje nasledujúcim riadkom.

Príklad: 200 IF A=0 THEN 40  
Ak hodnota A=0, vráti sa vykonávanie programu na riadok s číslom 40, ak A je rôzne od nuly, pokračuje sa na ďalšom riadku. Ten istý účinok vykoná aj programový riadok:

200 IF A=0 THEN GOTO 40

Príklad: 300 IF C=1 THEN GCLEAR  
Ak je hodnota premennej C=1, vymaže sa obrazovka, ak nie pokračuje sa hneď nasledujúcim riadkom.

### PODPROGRAMY

Pri tvorbe programov zisíme, že v nejakom programe sa bude niekoľko programových riadkov často opakovať (napr. vykresľovanie rovnakých ob-

rážkov). Aby sme ich zakaždým nemuseli písať, napíšeme ich iba raz. V programe potom použijeme príkazy, ktoré v prípade potreby nasmerujú vykonávanie programu do tejto časti a po jej vykonaní vrátia vykonávanie programu naspäť na nasledujúci riadok. Túto časť programu nazývame podprogram. Pre prácu s podprogramami používame príkazy:

GOSUB	volanie podprogramu
RETURN	ukončenie podprogramu a návrat do hlavného programu
ON GOSUB	prepínač volania podprogramov

Podprogramy pre zvýšenie prehľadnosti programu, píšeme až na konci programu za príkaz END. V prvom riadku podprogramu je zvyčajne príkaz REM s názvom podprogramu.

#### Príkaz: GOSUB

Tvar: GOSUB *cr*

Je to príkaz v hlavnom programe na privolanie podprogramu, *cr* je číslo riadka s prvým príkazom podprogramu, ktorý chceme privolať. Po vykonaní príkazu GOSUB sa odovzdá riadenie prvému riadku podprogramu, ktorý sme nim vyvolali.

Príklad: 200 GOSUB 1000  
210 .....

Privolá sa podprogram začínajúci na riadku s číslom 1000 a po jeho vykonaní sa pokračuje na riadku s číslom 210.

#### Príkaz: RETURN

Tvar: RETURN

Je posledným príkazom v podprograme. Môže ich byť aj viac a môžu sa nachádzať v ľubovoľnej časti podprogramu. Len čo sa pri vykonávaní podprogramu vykoná príkaz RETURN, riadenie programu nasleduje riadkom, ktorý sa nachádza za riadkom, ktorý vyvolal podprogram.

Príklad: 450 RETURN

Ak potrebujeme v programe viackrát kresliť obdĺžnik, na jeho vykresľovanie použijeme podprogram. Podprogram a jeho začlenenie do programu môže vyzeráť napr. takto:

```

10 REM nazov programu
.....
50 GOSUB 300
.....
170 GOSUB 300
.....
290 END
300 REM PODPROGRAM NA KRESLENIE OBDLZNIKA
310 SCALE 0,100,0,100
320 MOVE 10,10
330 PLOT 80,10
340 PLOT 80,50
350 PLOT 10,50
360 PLOT 10,10
370 RETURN

```

### Príkaz: ON GOSUB

Tvar: ON v GOSUB cr1, cr2, cr3, .....crN

kde v je aritmetický výraz (alebo premenná), ktorý nadobúda hodnoty 1, 2, 3, .....N  
cr je číslo riadka s prvým príkazom podprogramu, ktorý sa má vykonávať v závislosti od stavu premennej v.

Príklad: 50 ON K GOSUB 600, 700, 800, 900

Tento programový riadok určí, ktorý podprogram sa má vykonávať, v závislosti od stavu premennej K. Ak bude hodnota K=1, potom sa vykoná podprogram začínajúci na riadku s číslom 600. Ak bude hodnota K=2, vykoná sa podprogram začínajúci na riadku s číslom 700, atď. Program v básicu môže obsahovať ľubovoľný počet podprogramov. Každý podprogram môže ešte privolať ďalší podprogram.

### NIEKTORÉ FUNKCIE A POMOCNÉ PRÍKAZY

BASIC G obsahuje aj rôzne matematické a textové funkcie. Uvedieme si aspoň niektoré, hlavne tie, s ktorými sa stretávame aj na hodinách matematiky a budeme ich používať pri tvorbe programov.

#### Funkcia: ABS(X)

Určuje absolútnu hodnotu z čísla X.

Príklad: 20 PRINT ABS(-28.3)

Vypíše sa absolútna hodnota čísla X uvedeného v zátvorke, na obrazovku.

#### Funkcia: SQR(X)

Určuje druhú odmocninu z čísla X uvedeného v zátvorke.

Príklad: 40 PRINT SQR(25)

Vypíše druhú odmocninu z čísla 25 na obrazovku.

#### Funkcia: INT(X)

Vyberie celočíselnú časť z čísla X tak, že odtrhne jeho desatinné čísla a výsledok vypíše na obrazovku.

Príklad: 20 PRINT INT(36.375)

Výsledkom spracovania tohto riadku je vypísanie čísla 36 na obrazovku monitora. Ak číslo X v zátvorke bude záporné, potom výsledkom bude najbližšie menšie celé číslo.

Príklad: 20 PRINT INT(-28.432)

Výsledkom spracovania tohto riadku je výpis čísla -29 na obrazovku monitora.

Poznámka: funkcia INT sa môže používať napr. na zaokrúhľovanie čísla X na najbližšie celé číslo pomocou výrazu:  $\text{INT}(X+0.5)$

#### Funkcia: RND(X)

Vygeneruje náhodné číslo z intervalu 0 až 1. Na mieste X používame obyčajne číslo 1.

Priklad: `20 PRINT RND(1)`

Na obrazovku vypíše nejaké číslo z intervalu 0 až 1.

Ak potrebujeme vygenerovať náhodné číslo z iného intervalu, napr.  $\langle A, B \rangle$ , môžeme to dosiahnuť pomocou vzťahu:  $(B-A)*\text{RND}(1)+A$ .

Priklad: `20 PRINT INT((40-5)*RND(1)+5)`

Spracovaním tohto riadku sa vypíše na obrazovku náhodné číslo z intervalu  $\langle 5, 40 \rangle$ .

#### Funkcia: TAB(X)

Sa používa na nastavenie začiatku tlače výrazu v riadku, kde X je celočíselná hodnota. Riadok pre výpis na obrazovku má dĺžku 48 znakov. Pri výpise na tlačiareň pozri v ďalšej časti.

Priklad: `50 PRINT TAB(5)A; TAB(20)B`

Hodnota A sa zobrazí za piatou pozíciou v riadku a hodnota B sa zobrazí za dvadsiatou pozíciou v riadku (čísľujeme od čísla 0). Funkciu TAB používame v príkaze PRINT hlavne na výpis výsledkov vo forme tabuliek.

#### Príkaz: DISP

Tvar: `DISP zoznam`

Je príkazom pre výpis údajov (na mieste zoznamu) do dialógového riadka. Ak chceme vypísať text, uvedieme ho v zozname príkazu DISP do úvodzoviek. Príkaz DISP môžeme použiť napríklad na priebežný výpis údajov a výsledkov na obrazovku počas behu programu, alebo na výpis informácie aké vstupné údaje sa majú zadávať v príkaze INPUT.

Priklad: `80 DISP "Zadaj nove hodnoty premennych !"`  
`200 DISP B, 125*4`  
`300 DISP "zadaj stranu obdĺznika !"`

Každý nový príkaz DISP v programe spôsobí prepísanie doterajšieho obsahu dialógového riadku.

#### Príkaz: ?

Tvar: `? zoznam`

Má podobný význam ako DISP, avšak po vypísaní do dialógového riadka spôsobí pozastavenie vykonávania programu. Vykonávanie programu obnovíme (napr. po prečítaní správy v dialógovom riadku) stlačením ľubovoľnej klávesy. Zároveň sa tým vymaže aj obsah dialógového riadka.

Priklad: 100 ?"Ak chces pokracovat stlac lubovolnu klavesu !"

V riadku 100 sa pozastavi vykonavanie programu a obnovi sa až po stlačení ľubovoľnej klávesy. Tento príkaz môžeme výhodne používať napr. na konci zobrazovanej stránky v programe (pri zobrazovaní textov a pod.).

Funkcia: INKEY

Tvar: INKEY

Umožňuje načítať hodnotu stlačeného funkčného klávasu od 0 do 11, podľa čísla stlačeného klávesu. Ak nie je stlačený žiadny kláves, funkcia INKEY má hodnotu 255. Funkcia platí iba pre funkčné klávesy K0, K1, až K11 (nazývame ich aj programové kľúče). Ak je stlačený kláves K0, má funkcia hodnotu 0, ak je stlačený kláves K5, má hodnotu 5, atď. Vyskúšame si to na príklade:

```
10 A=INKEY
20 PRINT A
30 GOTO 10
```

Po spustení programu povelom RUN a stlačením EOL sa na obrazovke budú pod sebou zobrazovať čísla 255. Je to preto, lebo nie je stlačený žiadny kľúčový kláves K0 až K11. Ak stlačíme kláves K7, funkcia INKEY bude mať hodnotu 7, ktorá sa priradí premennej A. Na obrazovke sa bude vypisovať číslo 7, pokiaľ bude stlačený kláves K7. Ak stlačíme kláves K3, bude sa vypisovať číslo 3, atď. Beh tohto ukázkového programu môžeme zastaviť iba stlačením klávesy STOP. Funkciu INKEY používame predovšetkým na riadenie chodu programu (testy, hry a pod.).

Funkcia: LEN

Tvar: LEN(X\$)

Funkcia priradí ľubovoľnej číselnej premennej hodnotu, ktorá sa rovná počtu znakov v reťazci X\$.

Priklad: 100 A\$="SKOLA"  
110 A=LEN(A\$)  
120 PRINT A

Po spracovaní riadkov 100 až 120 sa na obrazovke monitora vypíše číslo 5 preto, lebo reťazec A\$ má 5 znakov.

Priklad: 100 C\$="PRINCEZNA JULIENKA"  
110 PRINT LEN(C\$)

Po spracovaní týchto riadkov sa na obrazovke monitora vypíše číslo 18 preto, lebo reťazec C\$ má 18 znakov. Medzeru medzi slovami predstavuje prázdny znak, ktorý sa tiež započítava do dĺžky reťazca. Funkciu LEN môžeme používať napríklad pri usporadúvaní reťazcov podľa dĺžky, prihľadani najkratšieho alebo najdlhšieho mena, názvu a pod.

#### GONIOMETRICKÉ FUNKCIE

BASIC G umožňuje používať i goniometrické funkcie. Pri ich používaní si musíme uvedomiť v akých uhlových mierach je vyjadrená hodnota argumentu X (uhla). Po spustení programu povelom RUN, ako aj po použití príkazu CLEAR sa automaticky nastavuje režim argumentu X na radiány. Na prepínanie režimov používame príkazy:

RAD prepne režim na RADIANY  
DEG prepne režim na STUPNE

Funkcia: SIN(X)

Vypočíta hodnotu sínusu čísla X.

Priklad: 30 DEG  
40 PRINT SIN(45)

Na obrazovke monitora vypíše hodnotu sínusu uhla 45 stupňov.

Funkcia: COS(X)

Vypočíta hodnotu kosínus čísla X.

Priklad: 100 DEG  
110 PRINT COS(60)

Na obrazovke monitora vypíše hodnotu kosínusu uhla 60 stupňov.

Funkcia: TAN(X)

Vypočíta hodnotu tangens čísla X.

Priklad: 20 DEG  
30 C=TAN(30)  
40 PRINT C

Premennej C priradí hodnotu tangensu uhla s veľkosťou 30 stupňov a jej veľkosť vypíše na obrazovke monitora.

Funkcia: ATN(X)

Vypočíta hodnotu arcus tangens čísla X.

Priklad: 30 DEG  
40 PRINT ATN(0.5)

Vypíše na obrazovku arcus tangens čísla 0,5.

Goniometrické funkcie môžeme používať napríklad na kreslenie kružnic, kruhov a podobných útvarov. Nasledujúci program nakreslí kružnicu do stredu obrazovky:

```
10 GCLEAR
20 SCALE -2, 2, -1.6, 1.6
30 DEG
40 MOVE 0,0
50 FOR I=1 TO 360
60 PLOT COS(I),SIN(I),1
70 NEXT I
80 END
```



## PRÁCA S KAZETOPÁSKOVOU PAMÄŤOU (MAGNETOFÓNOM)

Aby sme nemuseli zadávať program počítaču stále cez klávesnicu po každom zapnutí počítača, potrebujeme ho nejako uložiť. Počítač si totiž po jeho vypnutí alebo zadaní povelu NEW nezapamätá predchádzajúci program. Na tento účel používame kazetový magnetofón. Po prepojení určených miest počítača a magnetofónu spojovacími káblami, môžeme pomocou nasledujúcich povelov uchovať program na páske a v prípade potreby ho kedykoľvek prekopírovať z pásky do pamäte počítača. Sú to tieto povel:

SAVE    nahratie programu na pásku magnetofónu  
LOAD    prehratie programu z pásky magnetofónu do počítača  
CHECK   kontrola programu na páske

### Povel: SAVE

Tvar: SAVEn\*názov

Povel umožňuje uložiť program na pásku kazetového magnetofónu. *n* je poradové číslo programu na páske. Môže to byť číslo od 1 do 99. *názov* je názov programu a môže mať najviac 8 znakov.

#### Postup pri nahrávaní programu:

1. Do dialógového riadka napíšeme napr.: SAVE 04"HRATENIS (na páske sa bude ukladať pod číslom 4 s názvom HRATENIS).
2. Nastavíme podľa počítadla magnetofónu pásku na miesto, kde chceme program nahráť.
3. Zapneme nahrávanie magnetofónu.
4. Stlačíme kláves EOL.

Po správnom nahratí programu sa v dialógovom riadku vypíše OK. Vtedy vypneme magnetofón. Správnosť nahrávky môžeme ešte overiť nasledujúcim povelom.

### Povel: CHECK

Tvar: CHECK n

kde *n* je číslo programu na páske, ktorý chceme kontrolovať.

#### Postup pri kontrole programu na páske magnetofónu:

1. Nastavíme pásku podľa stavu počítadla magnetofónu tesne pred začiatok programu.
2. Napíšeme do dialógového riadku povel CHECK4 (ak kontrolujeme program na páske s číslom 4).
3. Stlačíme kláves EOL.
4. Spustíme magnetofón.

Ak počítač zachytí začiatok programu, vypíše to v dialógovom riadku a potom kontroluje správnosť nahrávky. Po celý tento čas je v dialógovom riadku vypísané číslo programu a názov programu na páske magnetofónu. Ak po skončení kontroly sa vypíše správa OK, je program nahratý správne, ak sa vypíše + file error + , musíme program nahráť znova povelom LOAD.

Povel: LOAD

Tvar: LOAD n

Je to povel na prekopírovanie programu z pásky magnetofónu do pamäte počítača. *n* je číslo programu, pod akým je program na kazete uložený.

Postup pri prehrávaní programu do počítača:

1. Podľa počítača magnetofónu a podľa údajov na kazete o programoch nastavíme pásku tesne pred začiatok programu.
2. Do dialógového riadka vypíšeme napr.: LOAD04 (ak chceme nahrávať program uložený na páske pod číslom 4).
3. Stlačíme klávesu EOL.
4. Spustíme magnetofón.

Ak sa program začne nahrávať, počítač to oznámi do dialógového riadka výpisom správy napr.: 04\>HRATENIS. Po úspešnom prekopírovaní programu do pamäte sa vypíše OK. Vtedy môžeme program spustiť povelom RUN a stlačením klávesy EOL. Ak sa vypíše chybové hlásenie, pokúsime sa prehrať program znovu povelom LOAD.

Povel: MGLD

Tvar: MGLD n

Je povelom pre prehrávanie programov, ktoré nie sú nahrané v basicu ale v strojovom kóde mikropočítača. Takto sa nahráva väčšina výkonnejších programov a hier, *n* je číslo programu pod akým je program uložený na páske.

Postup pri prehrávaní programov v strojovom kóde:

1. Počítač uvedieme do stavu, keď sa prihlási riadiaci program počítača MONITOR (pozri úvodné časti textov). U PMD-85 to môžeme vykonať napr. povytiahnutím ROM modulu a stlačením RESET. U Maťa to vykonáme stlačením kombinácie kláves buď CNT a MON alebo CNT a RST. Žiadany stav počítača sa prihlási v dialógovom riadku správou \*\* OS READY \*\*.
2. Do dialógového riadka napíšeme napr.: MGLD05 (ak chceme prehrávať program uložený na páske pod číslom 4).
3. Podľa stavu počítača magnetofónu nastavíme pásku pred začiatok programu.
4. Stlačíme klávesu EOL.
5. Spustíme magnetofón.

Keď počítač zachytí začiatok programu, vypíše to v dialógovom riadku správou napr.: 05/? "ZABY. Otáznik za číslom programu a lomítkom znamená, že program je nahraný na páske v strojovom kóde. Keď sa ukončí úspešné prehrávanie programu, vypíše to počítač v dialógovom riadku správou ++ READING OK ++. Ak sa vypíše hlásenie o chybe, pokúsime sa program prehrať znova.

Poznámka: Program v strojovom kóde spustíme povelom: JUMP adr, kde *adr* je štartovacia adresa programu v pamäti počítača. Adresa je zadávaná v šestnástkovej sústave. Pri štartovaní programu s adresou napr. 01FA napíšeme: JUMP01FA a stlačíme EOL.

**PRÁCA S TLAČIARŇOU**

Ak máme k počítaču pripojenú tlačiareň, napr. D100D, D100M alebo star LC-10 (prípadne inú), môžeme ju ovládať pomocou nasledujúcich príkazov a povelov:

CONTROL príprava počítača a tlačiarne pre spoluprácu  
 LIST# výpis programu uloženého v pamäti (tzv. listing programu na tlačiareň).  
 OUTPUT výstup premenných a konštánt z programu na tlačiareň. Tvary nasledujúcich slov platia pre mikropočítače PMD-85, MAŤO a Didaktik Alfa, pripojené na tlačiarne uvedené v úvode tejto časti.

#### Príkaz: CONTROL

Tvar: CONTROL 4,3;160,13

Tento príkaz používame v programe ako prvý príkaz pre prácu s tlačiarňou. Zabezpečí prípravu počítača a tlačiarne pre spoluprácu. Bez tohto príkazu nie sú účinné ďalšie príkazy a povel pre prácu s tlačiarňou.

#### Povel: OUTPUT

Tvar: OUTPUT 403;

Používa sa po príkaze CONTROL, ktorý tiež môžeme použiť ako povel, na kontrolu pripojenia tlačiarne. P jeho vypísaní do dialógového riadku a odoslani sa posunie valec tlačiarne "o jeden riadok". Ak sa to nestane, je chyba v pripojení tlačiarne. Ak za bodkočiarku napíšeme do úvodzoviek nejaký text, po stlačení EOL sa ten ihneď vypíše na tlačiarňu. Takto môžeme vypisovať krátke správy, pričom tvar povelu je vhodné si uložiť do niektorého programového kľúča (K0 až K11).

#### Príkaz: OUTPUT

Tvar: OUTPUT403;zoznam

Je príkazom na výpis hodnôt aritmetických alebo textových premenných, uvedených v zozname, z programu na tlačiareň.

Príklad: Ak cheme vypísať na tlačiarňu tabuľku, kde budú čísla a vedľa nich ich druhé a tretie násobky, potom napr. pre 10 čísel použijeme tento program:

```
10 GCLEAR
20 CONTROL4,3;160,13
30 OUTPUT403;"CISLA 1 AZ 10 A ICH DRUHE A TRETIE NASOBKY"
40 OUTPUT403;
50 FOR I=1 TO 10
60 OUTPUT403; I, 2*I, 3*I
70 NEXT I
80 END
```

#### Povel: LIST#

Tvar: LIST#403;cr

Je povelom na vytlačenie programu uloženého v pamäti počítača na tlačiarňu. cr je číslo programového riadka, od ktorého sa má program tlačiť. Ak cr neuvedieme, vytlačí sa celý program.

**Poznámka:** Pomocou takto napísaného povelu bude jeden riadok obsahovať 40 znakov. Ak by sme chceli zmeniť počet znakov v riadku, použijeme predtým povel `POKE 46,cr`, kde `cr` znamená počet znakov v riadku pri výpise na tlačiarňu.

**Príklad:** Ak chceme vypísať na tlačiarňu celý program a v riadku má byť 80 znakov, napíšeme a odošleme takýto viacnásobný povel:

```
CONTROL4,3;160,13 : POKE 46,80 : LIST#403;
```

#### 4. TVORBA A LADENIE PROGRAMOV

Keď poznáme význam slov v basicu, môžeme veľmi ľahko napísať jednoduché programy. Ak potrebujeme program napísať program na výpočet druhých a tretích mocnín čísel, napr. od 1 do 20 a ich výpis na obrazovku, píšeme:

```
10 REM VYPOCET DRUHYCH A TRETICH MOCNIN CISEL 1 AZ 20
20 GCLEAR
30 PRINT "DRUHE A TRETIE MOCNINY CISEL 1 AZ 20"
40 PRINT
50 FOR K=1 TO 20
60 PRINT K, K^2, K^3
70 NEXT K
80 END
```

Vidíme, že vytvorenie takéhoto programu je veľmi jednoduchá záležitosť. Keby sme však mali zostaviť program pre výpočet priemeru známok zo školskej úlohy z matematiky, nezostavíme ho už tak rýchlo. Pri takejto a ďalších oveľa zložitejších úlohách (a na tie hlavne chceme počítač využívať), musíme postupovať takto:

##### Postup pri tvorbe programu zložitejších úloh:

1. Najprv si musíme pozorne prečítať úlohu, pre ktorú máme zostaviť program. Musíme si uvedomiť:
  - a) aké údaje budú do riešenia vstupovať
  - b) aké údaje a koľko ich bude riešením úlohy
  - c) ako potrebujeme zobrazíť výstupné údaje (miesto na obrazovke, prípadne tlačiarňu)
  - d) ako budeme vkladať vstupné údaje - či budú zadané v programe alebo sa budú vkladať z klávesnice
  - e) aké matematické operácie použijeme
  - f) aké príkazy a funkcie programovacieho jazyka použijeme.
2. Po dôkladnom rozbere úlohy si musíme uvedomiť postupnosť jednotlivých krokov riešenia úlohy a načrtujeme schému riešenia úlohy vo forme vývojového diagramu.
3. Na vývojovom diagrame si úvahou overujeme správnosť nášho riešenia úlohy.
4. Prepíšeme riešenie úlohy z vývojového diagramu do programu, pomocou slov jazyka BASIC.

#### VÝVOJOVÉ DIAGRAMY

Vývojový diagram si zjednodušene môžeme predstaviť ako zápis riešenia úlohy pomocou značiek, čiar a skratiek. Keď máme zapísané riešenie úlohy vo forme vývojového diagramu, veľmi ľahko sa potom píše program. Značky, používané vo vývojových diagramoch sú uvedené na obr.1 za prílohami. Riešenie zložitejších úloh (rozbor úlohy, vývojový diagram, program) si ukážeme na príkladoch:

### PRÍKLAD A:

Zostav program na delenie dvoch čísel A a B. Hodnoty čísel A, B sa budú zadávať z klávesnice. Výsledky s príslušnými hodnotami A a B zobraz na obrazovke monitora.

Riešenie: Najprv si pozorne viackrát prečítame úlohu, aby sme si ujasnili čo vlastne máme riešiť. Zapišeme si vstupné a výstupné údaje:

vstupné údaje: číslo A  
                  číslo B - z klávesnice  
výstupné údaje: podiel A/B  
                  číslo A  
                  číslo B - zobrazíť na obrazovke

V úlohe nie je zadany počet dvojíc čísel A,B. Preto si ich počet musíme zvolit, napr. 20. Zapišeme vzťah pre delenie:  $X=A/B$ . Môžeme zadávať ľubovoľné čísla? Čo sa stane ak hodnota čísla B bude rovná 0? Ak  $B=0$ , počítač vypíše chybu, lebo nulou deliť nevie a v programe sa nedá pokračovať. Preto program musíme zostaviť tak, aby aj pri  $B=0$  pokračoval ďalej. Program musí zisťovať, či po zadaní čísel A a B nie je  $B=0$ . Ak áno, nech sa vypíše text, že dané čísla nemôžeme deliť a nech si žiada ďalšie A a B. Potom nech vypočíta podiel A/B a vypíše na obrazovku A, B a A/B. Ak to ešte neurobil 20-krát, nech si žiada ďalšie A a B. Teraz skúsime načrtnúť riešenie vo forme vývojového diagramu (pozri obr.2 za prílohami).

Program v basicu bude mať takýto tvar:

```
10 REM PROGRAM NA DELENIE DVOCH CISEL
20 GCLEAR
30 FOR I=1 TO 20
40 DISP "zadaj cisla A , B a stlac EOL"
50 INPUT A, B
60 IF B=0 THEN PRINT "B=0, zadaj nove A, B" : GOTO 90
70 X=A/B
80 PRINT "A="A, "B="B, "A/B="X
90 NEXT I
100 END
```

### PRÍKLAD B:

Zostav program na výpočet priemeru známok triedy, z kontrolnej práce z fyziky. Známky zadávajú postupne z klávesnice, pričom na začiatku nevieš koľko ich bude. Výsledný priemer zobraz na obrazovke monitora.

Riešenie: Zo zadania úlohy určíme vstupné a výstupné údaje:

vstupné údaje: Z hodnota známky  
výstupné údaje: PT priemer triedy

Nevieme koľko bude známok, preto musíme určiť koniec zadávania známok. Po poslednej zadanej známke pridržíme kľúč K0. Počas zadávania známok musíme spočítavať hodnoty známok, tiež aj spočítavať počet známok, aby sme mohli určiť priemer. Určíme si na to dve premenné. Premenná SZ bude uchovávať súčet hodnôt známok a premenná PZ bude uchovávať počet známok. Pre priemer triedy potom bude platiť vzťah:  $PT=SZ/PZ$ . Nakreslíme vývojový diagram (pozri obr.3 za prílohami).

Zapišeme program v basicu:

```
10 REM VYPOCET PRIEMERU ZNAMOK
20 GCLEAR
30 DISP "Zadaj znamku a stlac EOL !"
40 INPUT Z
50 SZ=SZ+Z
60 PY=PZ+1
70 DISP "posledna znamka ? ANO-stlac KO !"
80 PAUSE 10
90 A=INKEY
100 IF A=0 THEN 120
110 GOTO 30
120 PT=SZ/PZ
130 PRINT "PRIEMER TRIEDY JE: " PT
140 END
```

#### LADENIE PROGRAMOV

Po zapísaní programu do pamäte počítača a jeho spustení povelom RUN, nemusí program hneď správne pracovať. Zistíme to tak, že sa po odoslaní povelu RUN v dialógovom riadku vypíše druh chyby. Môže sa stať, že niektorý príkaz zapišeme chybne, napr. namiesto PRINT napíšeme PRUNT a pod. Takúto chybu počítač oznámi ako syntaktickú chybu a vypíše: + syntax err +, čiže chybný zápis príkazu.

Takúto chybu odstraníme veľmi ľahko pomocou povelu LLIST cr, kde cr je číslo riadku v ktorom je chyba. Po opravení riadku a jeho odoslaní do pamäte počítača, pokúsime sa spustiť program znova. Ak sa objaví podobný výpis znova, opravíme chybu takým istým postupom.

Syntaktické chyby (gramatické) - sa opravujú veľmi ľahko, horšie je to s logickými chybami. Tým sa vyhneme tak, že vykonáme dôkladný rozbor úlohy a postupu vykonávania jednotlivých krokov riešenia. Ak by sme mali napr. program na kreslenie obrázkov a kreslenie by neprebiehalo podľa našich predstáv, musíme hľadať chybu hlavne v zlej následnosti jednotlivých grafických príkazov, alebo v chybne zadanej niektorej súradnici bodu. Pri hľadaní takejto chyby je vhodné vkladať postupne medzi jednotlivé riadky príkaz STOP. Pomocou neho najskôr nájdeme chybný programový riadok.

Program, ktorý pracuje bez chýb, nazývame odladený program. Ak ho chceme používať neskôr, uchováme ho na magnetofónovej páske, alebo urobíme výpis na tlačiarňu.

#### 5. PRÍKLADY A ÚLOHY

Napiš program v jazyku BASIC na:

1. Vykreslenie obdĺžnika s veľkosťou pracovnej časti obrazovky.
2. Vykreslenie tvaru obálky na obrazovke monitora.
3. Vykreslenie rovnoramenného trojuholníka na obrazovku monitora.
4. Vykreslenie plochy obdĺžnika s veľkosťou pracovnej časti obrazovky.
5. Vypísanie svojho mena na celú plochu obrazovky.
6. Zväčšenie písmena A na celú obrazovku.
7. Desaťnásobé zaznenie zvukového návestia.
8. Vypísanie svojej adresy (meno a priezvisko, číslo domu, obec, PSČ) do stredu obrazovky pomocou príkazu AT.
9. na tú istú činnosť ako v úlohe 8., ale iba s príkazom PRINT.
10. Výpočet obvodu štvorca. Dĺžku strany zadávajú z klávesnice a výsledok vypíš na obrazovku.

11. Výpočet hustoty rovnorodého telesa, ak poznáš jeho hmotnosť a objem. Vstupné údaje zadávajú z klávesnice. Nezabudni na zadanie údajov o jednotkách !
12. Pre výpočet hmotnosti ľubovoľného rovnorodého telesa tvaru kvádra, ak poznáš jeho rozmery a hustotu. Z klávesnice budeš zadávať rozmery telesa a jeho hustotu. Vykonaj dôkladný rozbor úlohy.

Pomocou milimetrového papiera napíš program:

13. Na vykreslenie obrysov domu, v ktorom bývaš.
14. Zameraný na poznávanie trojuholníkov. Najprv nech sa na obrazovke vykreslia trojuholníky (rôzne druhy). Potom po stlačení ľubovoľnej klávesy nech sa pod obrázky vypíšu názvy jednotlivých trojuholníkov.
15. Na vykreslenie mrakov, bleskov a búrky na obrazovke. Využi pri tom prekresľovanie čiar a zvukové návěstie BEEP.

Vyskúšaj si nasledujúce programy a vysvetli ich činnosť:

```
PROGRAM: 1
10 GCLEAR
20 PRINT 3*2
30 END
```

```
PROGRAM: 2
10 GCLEAR
20 PRINT "3*2="3*2
30 END
```

```
PROGRAM: 3
10 GCLEAR
20 A=5
30 B=8
40 C=A+B
50 PRINT C
60 END
```

```
PROGRAM: 4
10 GCLEAR
20 PRINT "zadaj cislo A"
30 INPUT A
40 PRINT "zadaj cislo B"
50 INPUT B
60 PRINT "A="A, "B="B, "A/B="C
70 END
```

```
PROGRAM: 5
10 GCLEAR
20 BEEP
30 BEEP
40 BEEP
50 BEEP
60 BEEP
70 END
```

```
PROGRAM: 6
10 GCLEAR
20 BEEP:BEEP:BEEP:BEEP:BEEP
30 END
```

```
PROGRAM: 7
10 GCLEAR
20 I=0
30 BEEP
40 I=I+1
50 IF I=10 THEN END
60 GOTO 30
```

```
PROGRAM: 8
10 GCLEAR
20 FOR I=1 TO 10
30 BEEP
40 NEXT I
50 END
```

```
PROGRAM: 9
10 GCLEAR
20 FOR I=1 TO 10:BEEP:NEXT I
30 END
```

```
PROGRAM: 10
10 GCLEAR
20 SCALE 0,210,0,170
30 MOVE 0,0
40 PLOT 210,0
50 PLOT 210,170
60 PLOT 0,170
70 PLOT 0,0
```

```

PROGRAM: 11
 10 GCLEAR
 20 SCALE 0,210,0,170
 30 MOVE 0,0
 40 PLOT 210,0;210,170;0,170;0,0
 50 END

```

```

PROGRAM: 12
 10 GCLEAR
 20 SCALE 0,210,0,170
 30 AXES 0,0
 40 AXES 210,170
 50 END

```

```

PROGRAM: 13
 10 GCLEAR
 20 SCALE 0,10,0,10
 30 FOR I=0 TO 10
 40 AXES I,I
 50 NEXT I
 60 END

```

```

PROGRAM: 14
 10 GCLEAR
 20 SCALE 0,210,0,170
 30 MOVE 0,0
 40 FILL 255,242;1
 50 END

```

```

PROGRAM: 15
 10 GCLEAR
 20 SCALE 0,255,0,242
 30 MOVE 0,0
 40 FILL 255,242;1
 50 MOVE 1,1
 60 FILL 253,240;1
 70 END

```

```

PROGRAM: 16
 10 GCLEAR
 20 SCALE -2,2,-1.6,1.6
 30 DEG
 40 FOR I=1 TO 360 STEP 4
 50 MOVE 0,0
 60 PLOT COS(I),SIN(I)
 70 NEXT I
 80 END

```



## OBRAZOVÁ PŘÍLOHA 1

### ZNAČKY VÝVOJOVÝCH DIAGRAMOV

#### Texty k obr.1:

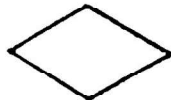
**Spracovanie:** Vykonalie operácií ako napr. násobenie, delenie, umocňovanie, v jednoduchých ale i zložitých výrazoch.



**Vstup/výstup:** Vstup a výstup údajov, napr. vstup cez klávesnicu, výstup na tlačiareň alebo obrazovku.



**Rozhodovanie:** Predstavuje operáciu riadenia programu, napr. porovnanie údajov, iný smer postupu (IF THEN, GOTO, ON).



**Vopred určená činnosť:** Predstavuje činnosti, ktoré v hlavnom programe nie sú bližšie určené. Používa sa na označenie podprogramov v hlavnom programe.



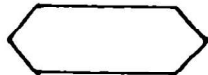
**Hranica:** označenie konca alebo začiatku riešenia.



**Spojka:** prechod na inú časť alebo z inej časti.



**Príprava:** predstavuje prípravu ďalšej činnosti, napr. pomenovanie premenných (DIM) alebo prípravu údajov.



**Spojníca so šípkou:** Spája jednotlivé značky, šípky ukazujú smer spracovávaného budúceho programu.

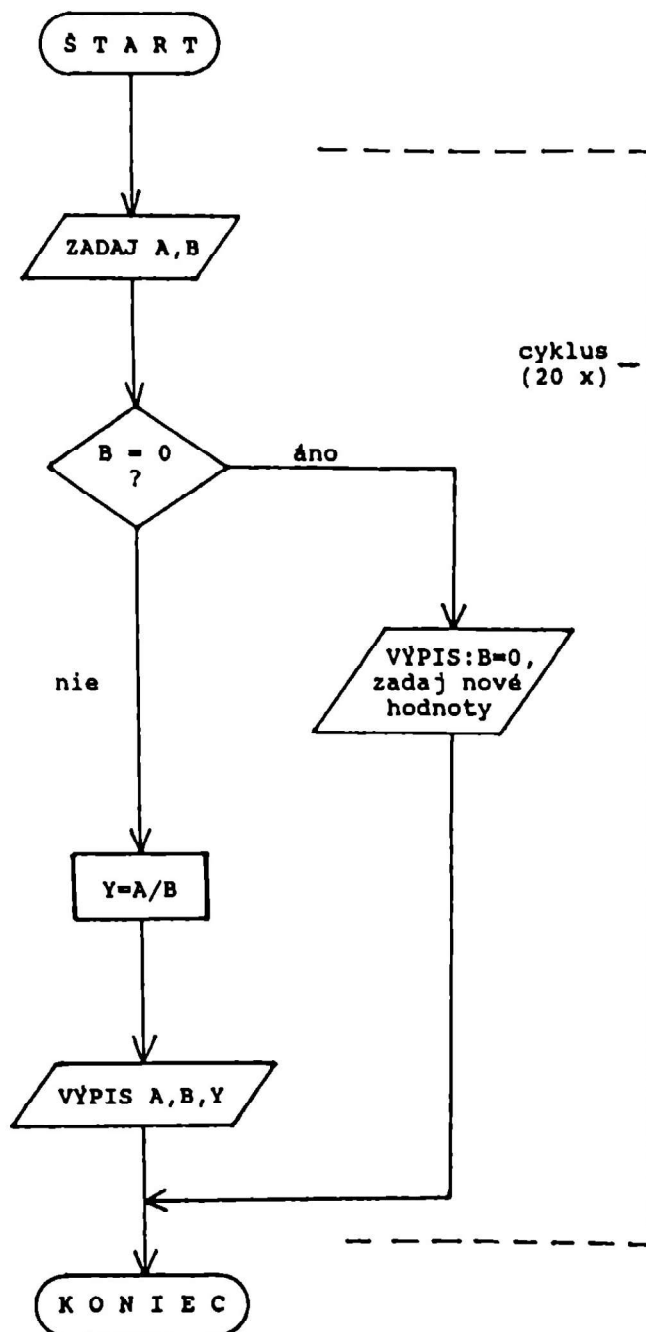


**Poznámka:** Môže byť pripojená k ľubovoľnej značke alebo spojnici a bližšie určuje význam jednotlivých častí.



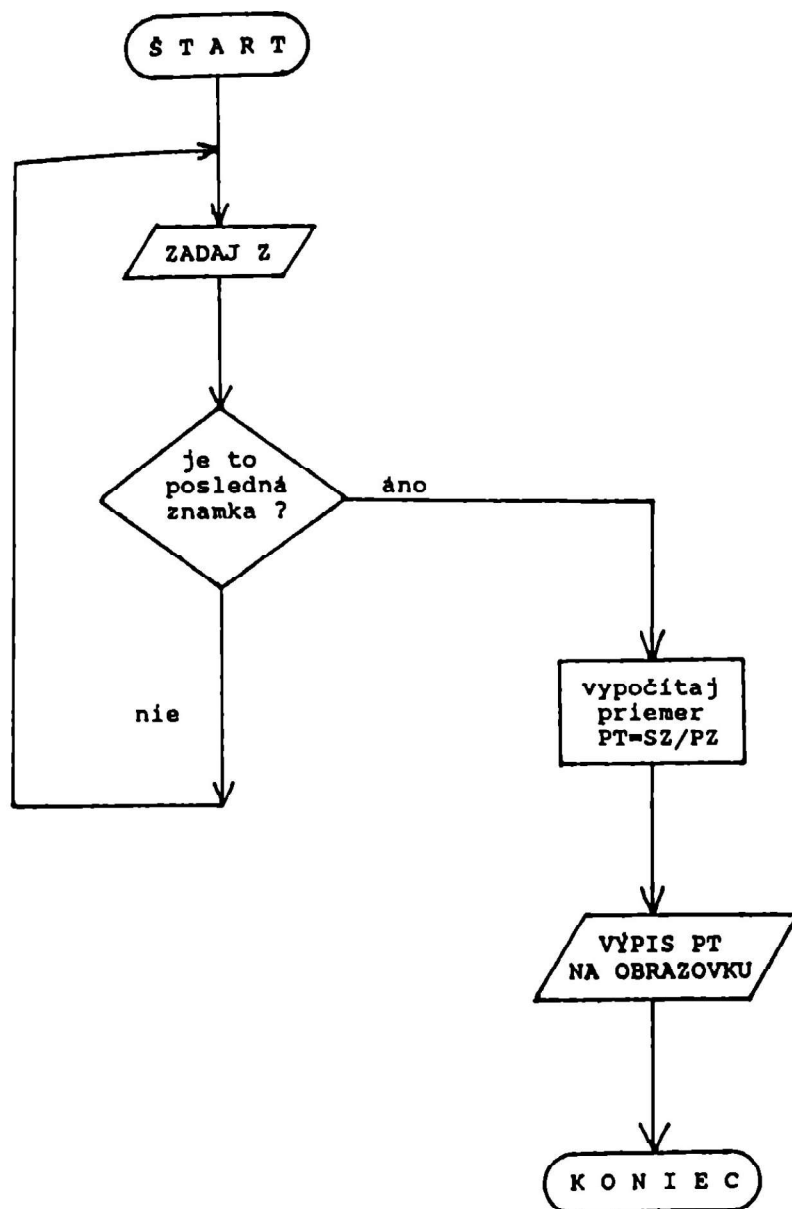
OBRAZOVÁ PŘÍLOHA 2

VÝVOJOVÝ DIAGRAM A



OBRAZOVÁ PŘÍLOHA 3

VÝVOJOVÝ DIAGRAM B



## PRÍLOHA: 1

### CHYBOVÉ HLÁSENIA.

Pri zadávaní príkazov mikropočítaču sa môžeme dopustiť rôznych chýb. Niektoré z nich mikropočítač nájde sám a oznámi to v dialógovom riadku (hlavne syntaktické a numerické chyby). Oznam o chybe je písaný v anglických skratkách a má najviac 11 znakov. Ak pracujeme v programovom režime, za oznamom je ešte číslo riadku, kde sa nachádza chyba. Ak sa v dialógovom riadku objaví chybové hlásenie, činnosť programu sa zastaví. Vtedy môžeme pomocou povelu `LLIST` chybu opraviť a opravený riadok odoslať do pamäte stlačením `EOL`.

Priklad chybového hlásenia:     + *syntax err* +

Slovo odoslané do pamäte bolo zapísané inak ako v `BASICU`, alebo je použitý neznámy výraz alebo číslo, napr.: `PRANT` namiesto `PRINT`, `BEP` namiesto `BEEP` a pod.

Priklad:     + *syntax err at line 50* +

Zle napísaný výraz sa nachádza v programe na riadku s číslom 50.

### CHYBOVÉ HLÁSENIA V JAZYKU BASIC G

#### Najčastejšie hlásenia a ich význam:

##### syntax err

Chybne zapísaný príkaz alebo použitý neznámy príkaz (skontrolovať správnosť-syntax zápisu).

##### overflow

Prekročený výpočtový rozsah. Pri výpočte vzniklo tak veľké alebo tak malé číslo, že počítač ho nemôže spracovať (upraviť program alebo zadať iné číselné hodnoty v príkaze `INPUT`).

##### input err

Chyba pri vkladani údajov (napr. namiesto čísla sme vložili písmeno).

##### only in pg

V priamom režime ste použili príkaz, ktorý je prípustný iba v programovom režime (napr. `INPUT`, `RETURN`, atď.).

##### no for stm

Nesprávne vytvorený cyklus (najčastejšie k príkazu `FOR` chyba príslušný príkaz `NEXT`, alebo sú zle vložené viacnásobné cykly).

##### data exhau

Chýbajúce údaje pre príkaz `READ` (dopísať do programu príslušné údaje do príkazu `DATA`).

##### field lost

Prekročený žiadaný počet vstupných údajov, ktoré požaduje napr. príkaz `INPUT`.

##### stop

Prerušenie činnosti programu (program môže pokračovať vo vykonávaní ďalej po odoslaní povelu `CONT`).

return err

Chyba pri návrate z podprogramu, alebo sa vyskytol príkaz RETURN bez predchádzajúceho GOSUB.

file error

Chyba pri prenose údajov z kazety (skúsiť prehrávať program ešte raz).

con't cont

Chybný povel CONT na pokračovanie v prerušenom programe. Pred ním bol použitý iný povel, ktorý už spôsobil zmenu v programe).

str. algrt

Reťazcový výraz je príkaz príliš dlhý alebo zložitý (je potrebné ho rozdeliť do niekoľko jednoduchých príkazov).

file boud

Rekročený maximálny počet záznamov magnetofónu 99.

pg too big

Príliš dlhý program, nezmestí sa do pamäte (pri úprave použite viac-príkazové riadky, skráťte reťazce, vynechajte nadbytočné medzery a pod.).

type conv

Chybný prevod, napr. medzi číselnou a reťazcovou premennou.

numb nonex

Programový riadok s uvedeným číslom neexistuje (upraviť číslo riadku).

Ostatné hlásenia a ich význam:

inc. param

Nesprávny parameter funkcie.

subscr. rng

Výraz v príkaze alebo funkcia je mimo definovaného rozsahu, napríklad v príkaze DIM.

arr alloq

Chybná dimenzia poľa alebo viackrát dimenzované pole.

string long

Dĺžka reťazca je viac ako 255 znakov.

no str. spc

Nie je už žiadne voľné miesto pre reťazce.

file small

Súbor údajov na kazete má iné deklarované pole, ako je príslušné pole v programe.

## PRÍLOHA: 2

### ANGLICKO - SLOVENSKÝ SLOVNÍČEK

Pri programovaní a obsluhu počítača používame anglické slová a skratky. Ak sa naučíme ich slovenský význam, naša práca bude oveľa rýchlejšia. Slovníček obsahuje najpoužívanejšie anglické slová v týchto textoch s ich slovenským významom.

adress - adresa	go to - ísť smerom
and - a	home computer - domáci počítač
basic - základný	if - keď
bit - jednotka informácie	input - vstup
byte - slabika	integer - celé číslo
call - volať	jump - skok
clear - čistý	key - kľúč
computer - počítač	let - nech
constant - konštanta	line - riadok
continue - pokračovať	load - naplniť
digit - číslica	memory - pamäť
dimension - rozmer, dimenzia	microcomputer - mikropočítač
display - displej	monitor - 1. systémový program
edit - upravovať text	2. prístroj s obrazovkou
end - koniec	move - presunúť
error - chyba	not - nie
false - nepravdivý	or - alebo
field - zóna, pole	output - výstup
fill - naplniť	overflow - preplnenie, pretečenie
for - pre	personal computer - osobný počítač
to - až k	poke - vložiť niekde niečo
step - krok	print - tlačiť
next - ďalší	printer - tlačiareň
priority - prednosť	save - uschovať
program - program	shift - posun
random - náhodný	space - medzera
read - čítať	string - reťaz, reťazec
remark - poznámka	subroutine - podprogram
restore - obnoviť	tape - páska
return - návrat	then - potom, tak
round - zaokrúhliť	true - pravdivý
run - bežať	zero - nula

## PRÍLOHA: 3a

### K L Á V E S N I C A   mikropočítača   M A T O

Klávesnica mikropočítača MATO sa nachádza na jeho vrchnej strane. Upúta nás množstvom popísaných 'gombíkov', ktoré nazývame klávesy. Každá klávesa má nejaké označenie, napr. A, C, M, I, STP atď. Tieto označenia nazývame znaky. Väčšina kláves má takýchto znakov viac ako jeden. Hovoríme, že sú to viacfunkčné klávesy. Všetkých kláves je spolu 55. V prvom riadku je 12 kláves, v druhom 11, treťom 13, vo štvrtom 14 a v poslednom 5 kláves. Klávesy sa tiež líšia farbou. Okrem šedých sú tam aj červené a modré, ktoré plnia osobitnú funkciu. Na červených klávesách sú napísané skratky z troch písmen. Tieto skratky znamenajú:

RST - RESET	STP - STOP	SHF - SHIFT
CNT - CONT	EOL - END OF LINE	

Klávesy EOL sú dve (majú rovnaký význam) preto, lebo sa často používajú. Na modrých klávesách sú šípky. Používame ich na ovládanie programu, napr. pri hrách alebo kreslení. V poslednom riadku je veľká klávesa bez označenia. Túto nazývame klávesa SPACE, alebo jednoducho - medzerník. Na zobrazovanie alebo mazanie znakov na obrazovke používame hlavne šedé klávesy. Stláčajme postupne klávesy v prvom riadku klávesnice. Zobrazia sa znaky:

1 2 3 4 5 6 7 8 9 0 - \

Za deviatkou nasleduje prečiarknutá nula (0). Je to preto, aby sme si ju nemýlili s veľkým písmenom O.

Pri postupnom stláčaní kláves v druhom riadku sa zobrazia znaky:  
Q W E R T Z U I O P @

Po stlačení šedých kláves v treťom riadku sa zobrazia znaky:  
A S D F G H J K L ; :

Po stlačení šedých kláves vo štvrtom riadku sa zobrazia znaky:  
Y X C V B N M , . /

Keď si lepšie prezrieme šedé klávesy, vidíme, že vľavo hore majú ešte nejaké znaky. Tieto znaky zobrazíme tak, že stlačíme klávesu SHIFT, ktorú pridržiavame a potom stlačíme klávesu so žiadaným znakom. Takto zobrazíme aj malé písmená. Pridržíme teraz stlačenú klávesu SHIFT a stláčajme postupne šedé klávesy. Po stlačení kláves v prvom riadku sa zobrazia znaky: ! # \$ % & ' ( ) \_ =

Po stlačení kláves v druhom riadku sa zobrazia znaky:  
q w e r t z u i o p

Po stlačení kláves v treťom riadku sa zobrazia:  
a s d f g h j k l + \*

a po stlačení vo štvrtom riadku sa zobrazia:  
y x c v b n m < > ?

Šedé a modré klávesy majú v dolnej v dolnej časti ďalšie znaky. Klávesy vo štvrtom riadku majú v dolnej časti skratky funkcií, ktoré uľahčujú prácu pri obsluhu klávesnice. Všetky dolné významy kláves platia pri súčasnom stlačení klávesy CONT a prísl.funkcie.

Význam radiacích funkcií vo štvrtom riadku:

- BCNL - posunie kurzor na začiatok riadku
- INVR - prepína inverzné zobrazenie obrazovky(po zapnutí počítača sa zobrazujú svetlé písmená na tmavom podklade. Po stlačení CNT+INVR bude text tmavý, na svetlo pozadí).
- WORK - text v dialógovom riadku za kurzorom sa zapíše do kľúča. Po napísaní textu pridržíme CONT a potom stlačíme na chvíľu klávesy WORK a kľúč, do ktorého chceme riadok uložiť - K0 až K11.
- RCI - vypíše sa obsah posledne odoslaného dialógového riadka (používame vtedy, ak chceme opraviť posledne odoslaný text, alebo napísať viac rovnakých riadkov).
- C-D - napíše sa posledný oznam mikropočítača do dialógového riadku napr. chybové hlásenie a pod.
- CLS - vymaže pracovnú časť obrazovky
- CLL - vymaže dialógový riadok
- DELT - vymaže znak na mieste kurzoru a ostatné znaky v riadku od tohto miesta napravo posunie doľava
- INST - vloží znak na mieste kurzoru, pričom ostatné znaky od kurzoru napravo posunie o jeden znak doprava
- MON - opustí basic a prihlási sa operačný systém MONITOR správou  
\*\* OS READY \*\*
- DELL - vymaže text od kurzoru do konca dialógového riadka
- PTL - vypne-zapne kopírovanie dialógového riadka do pracovnej časti obrazovky (používame ho napr. vtedy, ak chceme vykonať tzv. haard-copy obrazovky - vytlačenie obsahu obrazovky na tlačiareň . Vtedy by zobrazený dialógový riadok porušil obrázok na obrazovke, ktorý práve chceme tlačiť.
- BEEP - vypne-zapne akustickú signalizáciu stlačenia kláves. Pre kontrolu práce s klávesnicou používame akustické návestie preto, aby sme zbytočne nemuseli silno tlačiť na klávesy, a tým ich predčasne opotrebovať.
- ENDL - posunie kurzor na koniec riadku.

Mikropočítač MATO umožňuje zobrazíť rôzne grafické znaky. Zobrazíme ich tak, že pridržíme klávesu CONT a stlačíme príslušnú klávesu v druhom alebo treťom riadku.

Ak by sme potrebovali napísať viacero rovnakých znakov za sebou, nemusíme stále stláčať klávesu. Stačí ak klávesu pridržíme stlačenú. Znaky sa budú zobrazovať až do jej uvoľnenia.



## PRÍLOHA: 3b

### ODLIŠNOSTI KLÁVESNIC MIKROPOČÍTAČOV PMD-85 a MAŤO

Klávesnica PMD-85 má spolu 76 kláves. Na pravej strane klávesnice je skupina klávesov riadiacich funkcií. Sú to riadiace klávesy pomocou ktorých možno opravovať text a riadiť polohu kurzora v dialógovom riadku. Funkčné klávesy sú na PMD farebne odlišené (zväčša modré) od ostatných kláves (šedé alebo čierne).

V prvom riadku je prvých 12 kláves označených K0 až K11. Sú to programové kľúče. Za nimi nasleduje klávesa WRK, ktorá tieto programové kľúče aktivuje. Ďalej v riadku nasledujú funkčné klávesy C-D, RCL a RST.

V druhom riadku sú klávesy s číslami a špeciálnymi znakmi. Desiata a jedenásta klávesa na tomto riadku má oproti Maťovi navzájom prehodené významy [-] a [\_] (pomlčka a podčiarkovanie). PMD má klávesu so znakmi umocňovania a ľavého lomítka o riadok nižšie ako Maťo. Navyše má PMD za klávesou s hviezdíčkou a dvojbodkou ešte klávesu pre ľavú a pravú hranatú zátvorku.

Klávesa SHIFT u PMD má označenie (šípka hore), u Maťa - SHF. Funkciu BGNL u Maťa na PMD vykoná klávesa (šípka vľavo hore). Funkciu DELL u Maťa nahradí u PMD súčasné stlačenie kláves SHIFT a CLR. PMD nemá riadiace klávesy so šípkou hore a šípkou dolu, ktoré Maťo má. PMD však má klávesy (šípka vľavo so zarážkou) a (šípka vpravo so zarážkou), ktoré vykonávajú po stlačení posun riadka vľavo alebo vpravo o jeden znak. U Maťa tieto činnosti vykonávajú šípky vpravo a vľavo spolu s CONT, ale až po naplnení viditeľného obsahu dialógového riadka, teda 48 znakov.

## PRÍLOHA: 4

### ZOZNAM SLOV V BASICU, KTORÉ POUŽÍVAME V PROGRAMOCH AKO PRÍKAZY:

ABS	AXES	AT	ATN	CLEAR
CONTROL	COS	DATA	DIM	DISP
END	FOR TO STEP	FILL	GCLEAR	GOSUB
GOTO	IF THEN	INKEY	INPUT	INT
LABEL	LEN	LET	MOVE	NEXT
ON GOSUB	ON GOTO	OUTPUT	PAUSE	PLOT
PRINT	READ	REM	RND	RESTORE
RETURN	SCALE	SIN	SQR	STOP
TAB	TAN	?		

### ZOZNAM SLOV BASICU, KTORÉ POUŽÍVAME V PRIAMOM REŽIME AKO POVELY:

AUTO	BEEP	CONT	CONTROL	GCLEAR
CHECK	LIST	LIST#	LLIST	LOAD
NEW	OUTPUT	PRINT	SAVE	

Tieto príkazy a povely sú vysvetľované v týchto textoch.

### Niekoľko slov na záver.

Po prečítaní týchto textov by sa mohlo niekomu zdať, že sú neúplné, alebo zasa priveľmi náročné. Závisí to tiež aj od názoru toho - ktorého jedinca na vyučovanie tohoto predmetu na základnej škole. Nie sú určené pre konkrétny ročník ani neobsahujú návrh rozčlenenia učiva.

Výuka obsluhy a programovania výpočtovej techniky si ešte len vytvára svoj obsah na základnej škole a preto verím, že tieto texty budú vhodnou pomôckou pre ich užívateľov (učiteľov, žiakov) práve v čase, keď výkonná výpočtová technika pre svoju finančnú náročnosť na základných školách chýba. Pri tvorbe týchto textov som vychádzal z ich hlavného poslania, ktorým je:

1. Poskytnúť konkrétnu informáciu o výpočtovej technike všetkým žiakom už od 6. ročníka základnej školy.
2. Hľadať talenty.
3. Rozvíjať individuálnu prácu žiakov a vytvárať nové spôsoby získavania a upevňovania poznatkov.

Úlohou textov je pomôcť žiakom a učiteľom pri vytváraní obsahu nového predmetu, ktorý bude mať pevné miesto v novej školskej štruktúre v budúcom období. Prvá verzia týchto textov bola spracovaná na mikropočítači MATO a tlačiarňi D100M vo formáte A5.

Túto, ktorú ste práve dočítali už spracovalo "pécéčko".

Záverom želim veľa elánu a trpezlivosti do ďalšej práce všetkým tým, v ktorých som vzbudil záujem o výpočtovú techniku ako o nevyhnutnú súčasť našej prítomnosti.

A u t o r

## L I T E R Á T U R A:

- [1] Turnovec F.: Skúste to s basicom. Bratislava, Smena 1989.
- [2] Vojtek V., Návrat P., Drlik P.: Programovanie v jazyku BASIC, diel IV.a. Bratislava, vydalo oddelenie závodných organizácií SÚV SZM 1985.
- [3] Mannová B.: Programovanie, výpočtová technika - pre 1.ročník gymnázia. Bratislava, SPN 1987.
- [4] Franek M., Mečiar I.: PROGRAMOVANIE pre 3.ročník gymnázia, JAZYK BASIC. Bratislava, SPN 1987.
- [5] Kroha P., Slavík P.: Základy programování pro učitele základních škol. Praha, ČSVTS 1986.
- [6] Osobný mikropočítač MATO. Popis a návod na obsluhu. Štátny majetok Závadka nad Hronom, š.p., 1989.
- [7] Barát L., Palkovič F.: Príručka užívateľa Didaktik Alfa. Banská Bystrica, Učebné pomôcky, n.p., 1985.
- [8] Osobný mikropočítač PMD 85. Návod na použitie a obsluhu mikropočítača radu PMD. Bratislava, Tesla 1985.
- [9] Gábriš P., Haverlik I., Snorek M.: Svet mikropočítačov. Bratislava, Smena 1988.

Poznámky:

Koncept textu - autor: Miroslav Vojtek  
Zpracování textů, úprava, příprava a tisk předloh, distribuce:  
Soukromá firma  
PC DATA, Palackého ulice 150/34, 541 01 TRUTNOV  
Jakékoliv rozmnožování, pořizování kopií či následný prodej je povolen  
jen se souhlasem firmy PC DATA, která zastupuje i autora.