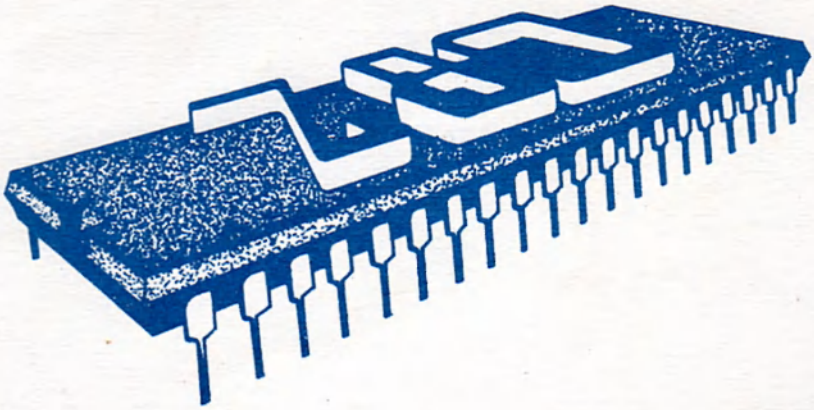
ZX SPECTRUM
BEZ HRANĪC



Táto publikácia je určená pre všetkých majiteľov niektorého z širokej rodiny počítačov vychádzajúcich zo štandartu SINCLAIR ZX SPECTRUM:

ZX Spectrum /+/128K/+2/+2A/+3, Delta, Didaktik GAMA, Didaktik M.

Z X S P E C T R U M

B E Z H R A N Í C

(VIAC AKO 100 PROGRAMOV A PODPROGRAMOV)

O B S A H

Klávesnica 009
Príkaz PRINT 010
Formy PRINT a TAB 011
Doplnok TAB 012
Raketa 012
Nahranie programu 014
VERIFY a MERGE 015
PRINT AT 016
Farba a grafika 018
PLOT 025
DRAW 026
CIRCLE 026
Kreslenie línií 028
Kobercový vzor 029
POINT 030
Tlačiareň 031
Náhodné čísla 031
Býčí zápas 032
Číselné a reťazcové premenné 034
Cvrčkov program 035
INPUT 036
Hádanie čísiel 036
Výpočet úrokov 037
GO TO 041
IF - THEN 042
IF - THEN (ELSE) 046
FOR - TO - NEXT slučky 050
Vnorené slučky 051
STEP 052
GO SUB a RETURN 053
Zvuk a hudba 055
DEF FN 057
DIM a polia 058
Hádanie čísla 060
Reťazcové polia 061
Práca s reťazcami 062
Použitie funkcie LEN 063
Použitie funkcie STR$ 063
INKEY$ 065
READ (DATA) RESTORE 067
Užívateľom definovaná grafika 069
Dotman 071
Vymazanie časti obrazovky 074
Scrollovanie po obrazovke 074
Ponechanie riadkov na obrazovke 075
Rolovanie po obrazovke 075
Pohyblivá grafika 076
Pretekár 078
Horiaci šíp 079
Handrár 081
Manipulácia s reťazcami 082
INKEY$ a CAPS SHIFT 086

Úvod do počítačovej aritmetiky 087
Prvočísla 088
Štatistika 088
Zajace a líšky 093
Funkcie 093
Trigonometrické funkcie 094
Pseudonáhodné čísla 094
Prenos cudzích dialektov BASIC 095
PEEK a POKE 101
Obchodné programy 104
Spracovanie textu 104
Knižný index 106
Vlastné vylepšovanie programov 107
Program, program, program 109
Psie závody 109
Biliardová guľa 110
Farebné piškvorky 110
Generácia 112
Kolónia buniek 113
Zápalky 114
Hrací automat 114
Búranie steny 116
Galaxia 117
Prevod z binárneho kódu na dekadický 118

K L Á V E S N I C A

Na prvý pohľad vo Vás zanechajú tlačitka Spectra dojem veľkej komplikovanosti, ale veľmi rýchlo ich pochopíte, ak budete iba trochu systematicky pracovať.

Dve dôležité tlačitka sú CAPS SHIFT (najnižšia rada vľavo) a SYMBOL SHIFT (najnižšia rada, druhé z prava). Našli ste obe? Upozorňujú na seba farbou tak, ako to potrebujete. Zapnite Spectrum, stlačte CAPS SHIFT a súčasne tlačitko "2". Teraz sa budú vypisovať všetky písmená ako veľké. CAPS LOCK je trvalá funkcia CAPS SHIFT, druhé stlačenie oboch tlačitiek vráti pôvodnú poziciu.

Iný vzťah pomocou tlačitiek 1 až 4 chceme vysvetliť neskôr. Pozrite sa na červené tlačitko SYMBOL SHIFT ešte raz. Stlačte ho súčasne s iným tlačitkom (okrem ENTER, BREAK SPACE a CAPS SHIFT). Na obrazovke sa zobrazí slovo alebo symbol, ktorý je na tlačitku zvýraznené červene. Napr. "$" alebo šípka nahor.

Teraz skúsime čo sa stane, keď stlačíme len jedno tlačitko. Vypíše sa jedno slovo, napr. IF, NEXT alebo DIM. Toto sú kľúčové slová, ktoré dávame na začiatok programového riadku. S nimi sa budeme zapodievať zachvíľu.

Zelené kľúčové slová na tlačitkach docielime, ak stlačíme súčasne tlačitka CAPS SHIFT a SYMBOL SHIFT a potom stlačíme odpovedajúce tlačitko. Červené slová a znamienka dolu na tlačitku sa budú vypisovať vtedy, ak súčasne stlačíme tlačitka CAPS SHIFT a SYMBOL SHIFT a potom pri stálom držaní tlačitka SYMBOL SHIFT stlačíme odpovedajúce tlačitko.

A teraz dôležité funkcie:

EDIT - túto funkciu potrebujeme, ak meníme alebo opravujeme riadok v programe. Najskôr sa nastavte pomocou CAPS SHIFT a tlačítok 6 a 7 (kurzorové šípky) na riadok, ktorý chcete opravovať. Potom pomocou EDIT (t.j. CAPS SHIFT a 1) prenesie riadok do editačnej zóny na spodný okraj obrazovky. Teraz sa môžete pohybovať kurzorom pomocou tlačitiek 5 a 8 (opäť spolu s CAPS SHIFT) v riadku a robiť ľubovoľné korekcie.

ENTER - stlačte ho a programový riadok, ktorý je napísaný na dolnom okraji obrazovky sa prenesie do programu hore. Tiež pre príkazový vstup (napr. RUN) je ENTER užitočný. Po jeho stlačení sa príkaz vykoná.

GRAPHICS - ak stlačíte CAPS SHIFT a 9 uvidíte, že sa kurzor zmení na "G". Ak stlačíte teraz číslicové tlačítka v najhornejšom rade najskôr bez a potom s tlačítkom CAPS SHIFT, uvidíte, že sa zobrazuje príslušná grafika. Druhá možnosť grafického módu bude vysvetlená neskôr.

DELETE - je tak povediac "guma" vo Vašom počítači. Napíšte niečo a stlačte potom CAPS SHIFT a 0. Posledný znak bude vymazaný. Ak tlačítka držíte stále, mení sa v trvalú funkciu.

Nenechajte sa počiatočnou zložitosťou ovládania klávesnice odradiť. Budeme sa snažiť vysvetliť všetko čo najjednoduchšie, ale mnohé sa Vám bude zo začiatku zdať dosť náročné. Je tu jediná možnosť: Pracujte čo možno najviac so Spectrom, skúšajte a hľadajte sami všetky variácie ovládania. Uvidíte sami, že za krátky čas bude pre Vás obsluha prístroja predstavovať len druhoradý problém.

P R I K A Z P R I N T

PRINT je najpotrebnejší príkaz v BASICu. Umožňuje počítaču, aby s Vami komunikoval. Napíšte: PRINT 5 a stlačte potom ENTER. Počítač napíše ihneď číslicu 5 vlavo hore na obrazovke. Môžete tiež previesť výpočet. Zadajte PRINT 5-3 a po ENTER sa ukáže výsledok 2. Touto priamou metódou môžeme prevádzať i komplikované úlohy. Skúšajte (vždy s ENTER) rôzne príklady.

```
PRINT SQR (8+1)
```

Vypočítajte odmocninu súčtu čísiel medzi dvoma zátvorkami. Dostanete výsledok 3. Pomocou príkazu PRINT môžeme zobrazovať nielen čísla, výsledky výpočtov, ale práve tak slová a písmená. Stlačte CAPS SHIFT a 2, aby ste prepli na veľké písmená, a zadajte PRINT HALLO. Po odoslaní klávesou ENTER hlási počítač chybu : "2 variable not found". Riešenie je jednoduché: Slová, ktoré má počítač napísať na obrazovku, musia byť v úvodzovkách. Skúste teraz:

```
PRINT "HALLO"
```

Teraz zasvieti HALLO na Vašej obrazovke.

Normálnym spôsobom práce pre počítač je však programovanie. A teraz je čas začať písať jednoduché programy.

Zadajte (s ENTER na konci v každého riadku) nasledujúce riadky:

```
10 REM PROGRAM 1
20 PRINT "TOTO JE TESTOVACI PROGRAM"
30 PRINT 1
40 PRINT 2
50 PRINT "TU JE KONIEC PROGRAMU"
```

Keď teraz stlačíte RUN (pre odštartovanie programu) a ENTER dostanete nasledujúci výsledok:

```
TOTO JE TESTOVACI PROGRAM
1
2
TU JE KONIEC PROGRAMU
```

Teraz sa musíme mnohému naučiť. Stlačte LIST (tlačítko K) a ENTER a program sa nám vylistuje (listing). Každý riadok má na začiatku číslo. Prvým v našom programe je 10 pri slove REM. To čo nasleduje za REM, je počítačom celkom ignorované, ale pri listingu je zobrazené. Toto je veľmi praktické, pretože napr. môžeme zaviesť do programu vysvetlivky, čo demonštruje program na nasledujúcich riadkoch.

Môžeme zadať vždy číslo riadku, REM a potom ľubovoľný text:

```
10 REM VYPOCET BODU
20 REM UHLOVY PREPOCET
```

Ak je potreba inštrukcie REM vo viacerých programových riadkoch, musí

byt na začiatku každeho riadku. Priklad:

```
60 REM TU BUDE VYSLEDOK
70 REM NASOBENIE
80 REM A a B SPOCITANE
```

A teraz nieco nove: Napíšte jednoducho 10 a stlačte ENTER. Čo sa stalo? Riadok 10 sa stratil. Pre vymazanie riadku alebo pre napisanie noveho nemusíte použiť DELETE tlačitko, ale celkom jednoducho napísať len číslo riadku.

```
20 PRINT "TOTO JE TESTOVACI PROGRAM"
30 PRINT 1
40 PRINT 2
50 PRINT "TU JE KONIEC PROGRAMU"
```

Napíšte teraz 10 REM zase do programu a potom:

```
15 PRINT "TOTO JE NOVY RIADOK"
```

Po ENTER vhupne riadok presne na svoje miesto do programu zodpovedajúce číslu riadku. Toto je velmi dôležite pre spracovanie programu pri zmenách smerom k vyšším alebo nižším riadkom.
Túto výhodu požijeme neskôr (ako dalej uvidíme) pre príkazy zavádzané dodatočne do programov.

```
10 REM
15 PRINT "TOTO JE NOVY RIADOK"
20 PRINT "TOTO JE TESTOVACI PROGRAM"
30 PRINT 1
40 PRINT 2
50 PRINT "TU JE KONIEC PROGRAMU"
```

Niekedy môže byt praktické či užitočné, ak môžeme prácu počítača na určitom mieste prerušiť. K tomu máme STOP príkaz (vyvoláme ho pomocou SYMBOL SHIFT a A tlačitka). Zadajte 25 STOP, uvedte program do chodu a uvidíte, že Spectrum skutočne urobí príkaz STOP. Na spodnom okraji obrazovky zasvieti hlásenie "9 STOP statement 25 : 1" To znamená: Návestie bolo prevedené 1. príkazom na riadku 25.
Skôr ako sa zoznámime s dalšími možnosťami príkazu PRINT, zoznámime sa s príkazom NEW. Po stlačení tlačitka A (bez SHIFT tlačitka) v príkazovom móde (K) sa zobrazí príkaz NEW. Po jeho odoslaní zabudne počítač všetko, čo priamo alebo pomocou kazety prijal a vyčisti stôl pre budúci program. Toto je velmi dôležite ak si nieste istí, či nezostali v pamäti jednotlivé riadky z minulého programu, ktoré by ste mohli preniesť do noveho. Zadajte teda teraz (a vždy keď píšete nový program) NEW.

F O R M Y P R I N T A T A B

Skôr ako sa zoznámime s dalšími možnosťami príkazu PRINT stlačte opäť CAPS LOCK pre velké písmená a zadajte tento program:

```
10 REM NAVODY
20 PRINT
```

```
30 PRINT
40 PRINT
50 PRINT
60 PRINT "HALLO ";60
70 PRINT "HALLO ",70
80 PRINT 12
90 PRINT 1,2
100 PRINT,1
110 PRINT,1;2
```

Starostlivo napište, čo je predtlačené v knihe, objasni Vám to veľa nového. Prejdeme riadok po riadku:

10 Titul programu je v REM príkaze. Nehrá v programe žiadnu rolu.
20-50 Každý PRINT - riadok bez textu vypíše prázdny riadok a posúva PRINT pozíciu o jeden riadok dole. Odtiaľ pochadza odstup, ktorý pozorujeme na hornej časti obrazovky, keď prebehne program.
60 PRINT tu vypíše slovo "HALLO" a za nim (s odstupom jedného znaku) číslo 60, čím vieme v ktorom riadku je teraz.
70 Čiarka presunie PRINT pozíciu na druhu polovicu doprava na televíznej obrazovke.
80 Tento zápisový spôsob zobrazuje obe čísla 1 a 2 za sebou. Ak budete čísla písať bez znaku medzi nimi, bude počítač pri prevádzaní programu zobrazovať čísla rovno za sebou.
90 Čiarka určuje pre obe čísla pozície na oboch poloviciach obrazovky.
100 Ako u "HALLO" v riadku 70, presunie čiarka text v PRINT pozícii do druhej polovice obrazovky.
110 Stredník zaistí písanie oboch čísiel priamo za sebou.

Môžete využiť stredník a čiarku pre rôzne usporiadanie výpisu na obrazovke. Ďalej si to predvedieme. Zadajte NEW a pozorujte, čo sa pri nasledujúcich programoch zobrazí, keď sa určitým spôsobom zapisujú rôzne čísla (hodnoty 1 - 10). Program 2A zoraďuje čísla do jednej rady pod seba. Program 2B priamo vedľa seba, program 2C vo dvoch radách a 2D zobrazuje čísla s rozostupom.

```
10 REM PROGRAM 2A
20 FOR j=1 TO 10
30 PRINT j
40 NEXT j

10 REM PROGRAM 2B
20 FOR j=1 TO 10
30 PRINT j;
40 NEXT j

10 REM PROGRAM 2C
20 FOR j=1 TO 10
30 PRINT j,
40 NEXT j

10 REM PROGRAM 2D
20 FOR j=1 TO 10
```

```
30 PRINT j;" ";
40 NEXT j
```

D O P L N O K T A B

TAB sa obyčajne kombinuje s PRINT a posuva poziciu PRINT o toľko medzier doprava, koľko je zapísané za príkazom TAB. Napíšte programy 2E a 2F a pozrite sa na ne:

```
10 REM PROGRAM 2E
20 FOR j=1 TO 10
30 PRINT TAB j;j
40 NEXT j
```

```
10 REM PROGRAM 2F
20 FOR j=1 TO 10
30 PRINT TAB 3*j;j
40 NEXT j
```

R A K E T A

Náš nasledujúci program "RAKETA" ukazuje krajšie, ako dômyselne bude príkaz TAB presúvať PRINT pozíciu. Najlepšie bude, keď si tento program napíšete do počítača a budete pritom opäť pozorne sledovať knihu.

```
 10 REM RAKETA
 20 DIM a$(6,5)
 25 BORDER 0
 30 FOR j=10 TO 0 STEP -1
 40 PRINT TAB 3*j;INK 2;j
 50 FOR a=1 TO j
 52 BEEP .02,5*j
 60 NEXT a
 70 NEXT j
 71 FOR b=1 TO 6
 72 READ q$
 73 LET a$(b)=q$
 74 NEXT b
 80 REM ** Hlavny program **
 90 LET q=INT (RND*25)+1
100 FOR r=1 TO 6
110 BEEP .02,10*r
120 PRINT "("; TAB q; INK r;a$(r); INK 0; TAB 31;")"
130 NEXT r
140 POKE 23692,-1
150 LET priestor =q/3
180 FOR p=1 TO priestor
190 PRINT "("; TAB 31;")"
200 NEXT p
210 GO TO 90
220 DATA "  `","  #","  =O#","  I_I",")<*>(","  * *"
```

Najdôležitejšie riadky pre našu diskusiu sú riadky 120 a 190, pretože

to su TAB prikazy pre obrazovku. Riadok 120 má nasledujúce funkcie:

Znak "("	- ohraničí hraciu plochu na ľavom okraji obrazovky.
TAB q	- q je náhodné číslo medzi 1 a 25, ktoré je vygenerované na riadku 90. Určuje o koľko pozícií bude nasledujúca PRINT pozícia v riadku presunutá doprava.
INK r	- určuje, akú farbu dostane raketa.
a$(r)	- určuje, ktorá časť rakety bude následne vykreslená. Využíva časti reťazca a$, ktorý je príkazom READ načítaný na riadku 71 až 74. Nenechajte sa popliesť, k tomuto komplexu sa neskôr ešte vrátime.
INK 0	- vracia peru späť čiernu farbu.
TAB 31	- Po nakreslení rakety sa vracia pozícia PRINT na postavenie 32 počítané zľava od nuly, t.j. tesne k pravému okraju obrazovky. Tam bude zobrazené pravé ohraničenie hracej plochy.

Teraz sa pozrieme na riadok 190:

Riadok 190 je súčasť slučky FOR - NEXT, ktorá začína na riadku 180 a riadkom 200 končí. Počet priebehov tejto slučky je závislý na náhodnom čísle q, ktoré je vypočítané vyššie.

PRINT "("; TAB 31;")" zobrazí prvú zátvorku celkom vľavo na obrazovke, presunie sa príkazom TAB na 31. pozíciu a zobrazí tu druhú zátvorku. Úlohou riadku 190 je vygenerovanie rôznych odstupov medzi raketami. Riadok 140 s POKE príkazom sa starať, aby sa počítač zakaždým nepýtal na scroll, toto ešte vysvetlíme podrobne.

N A H R A N I E P R O G R A M U

Možno sa Vám program "Raketa" zapáčil tak, že si ho budete chcieť uložiť pre neskoršie používanie. Potom si ho nahrajte na kazetu. Pripravte si magnetofón na nahrávanie a napíšte príkaz SAVE nasledovaný menom programu v úvodzovkách.

SAVE "Raketa"

Stlačte ENTER, spustite nahrávanie na magnetofóne a stlačte ešte raz ľubovoľné tlačítko. Za pár sekúnd máte zaistenú "Raketu" na večnosť.

Môžeme Vám dať ešte niekoľko dobrých rád:

Pre ukladanie programov používajte kazety kvalitných značiek. Šetrenie na páskach by sa Vám nemuselo vyplatiť a mohlo by sa vrátiť v podobe straty niekoľkohodinovej práce. Na uloženie často používaných programov používajte radšej krátku pásku, na ktorej Vám nebude robiť problém program rýchlo nájsť. Pre prácu s kazetami používajte výhradne kvalitný magnetofón (najlepšie špeciálny datarekorder), so zabudovaným počítadlom. Ak si na dlhšiu pásku uložíte program, starostlivo si poznamenajte stav počítadla na začiatku programu. Ušetrí Vám to veľmi veľa času pri vyhľadávaní programov.

Magnetofónová hlava by mala byť často čistená liehom a nie čistiacou kazetou pri chode počítača, pretože by sme mohli dostať nečistý signál.

V E R I F Y A M E R G E

Ak je program na kazete, mame možnost vyskúšať, či je program riadne nahraný, či pri jeho zápise nedošlo k chybe.
V našom prípade overíme funkčnosť programu pretočením nahrávky späť na začiatok a zadaním:

```
VERIFY "Raketa"
```

Spustíme prehrávanie na magnetofóne, necháme pásku bežať a sledujeme pásy na okraji obrazovky. Ak prebehla kontrola bezchybne obdržíme nakoniec hlásenie: "0 OK". Tak sme zistili, že je nahrávka v poriadku.
V opačnom prípade obdržíme hlásenie "R Tape loading error" (chyba pri nahrávaní). To znamená, že vznikla chyba pri nahrávaní do počítača, alebo je nahrávka na páske chybná a treba ju nahrať znova.

Existuje možnosť nahrávania nového programu do počítača, pričom starý program je ešte v pamäti. Toto nahrávanie prebehne tak, že keď sú v programe, ktorý je v pamäti rozličné čísla riadkov ako v programe na páske, staré riadky ostanú neporušené a program sa doplní novými. V prípade identických čísiel riadkov nové riadky prepíšu staré.

Demonštrujeme si to na nasledujúcom príklade:

Napíšte tento krátky program, uložte ho na kazetu pomocou príkazu SAVE "jedna", a vymažte pamäť počítača príkazom NEW.

```
10 REM TESTOVACI PROGRAM - JEDNA
20 REM LINE 20
30 REM LINE 30
40 REM LINE 40
```

Potom napíšte druhý program:

```
 5 REM TESTOVACI PROGRAM - DVA
15 REM LINE 15
25 REM LINE 25
35 REM LINE 35
```

Ponechajte ho v pamäti a zadajte MERGE "jedna", pretože prvý program bol nahraný pod týmto menom. Teraz spustite prehrávanie na magnetofóne (po spätnom pretočení) a za sekundu získate tento výsledok:

```
 5 REM TESTOVACI PROGRAM - DVA
10 REM TESTOVACI PROGRAM - JEDNA
15 REM LINE 15
20 REM LINE 20
25 REM LINE 25
30 REM LINE 30
35 REM LINE 35
40 REM LINE 40
```

MERGE je veľmi dobrá vec, ak chcete Vaše skôr písané programy prihrávať k novým. Aby ste sa vyhli nebezpečiu prepísania dôležitých riadkov, doporučujeme Vám, aby ste takýto častokrát používaný podprogram

začali na vyšších riadkoch (napríklad od riadku 9000). Takto sa podprogram automaticky zaradí pod hlavný program.

Veľakrát môžete mať ťažkosti pri nahrávaní programov. Tu je niekoľko rád, ktoré Vám môžu pomôcť:

1. Vytiahnite spojovací kábel počítača a magnetofónu v priebehu práce, keď nepoužívate oba prístroje.
2. Používajte ako zdroj magnetofónu radšej sieťové napätie než batérie.
3. Stavajte počítač a magnetofón čo možno najďalej od televízora.
4. Skúšajte rôzny stupeň hlasitosti na magnetofóne. Mnohé kazety dávajú značne slabé alebo silné signály. Experimentujte s tónovou clonou.
5. Skúste, či nemá kábel vadný kontakt, či nie je nejakým spôsobom porušený, alebo pájkované miesto poškodené.
6. Prekontrolujte, či ste správne zapojili prepojovací kábel.

Teraz sa vráťme späť k príkazu TAB. Spomínate si ešte, že následné číslo pre TAB určuje pevnú polohu PRINT pozície.

Pomocou TAB môžeme pracovať vždy v novom riadku. Po príkaze AT nasledujú dve čísla, o ktoré bude PRINT presunutý. Prvé číslo určuje na ktorý riadok sa bude písať, druhé číslo hovorí, aký bude odstup od ľavého okraja.

PRINT AT 10,6 znamená, že PRINT pozícia bude na 10. riadku a bude posunutá na 6. pozíciu do prava. PRINT AT 0,31 píše do horného pravého okraja obrazovky. PRINT AT 0,0 píše v ľavom hornom rohu obrazovky atď.

Riadok Specra má šírku 32 znakov, ak začíname počítať od nuly, pričom stĺpec 31 je najkrajnejšia pravá pozícia.

P R I N T A T

Nasledujúci program SQUASH využíva PRINT AT x,y pre loptu (v riadku 620); pre pálku (v riadku 150); pre nastavenie do správnej pozície. Pálkou môžeme pohybovať pomocou tlačítiek "O" a "P". Ako dlho je lopta v pohybe a počet úderov ukazuje bodový ukazovateľ. Ak je hra na konci, stlačí sa akékoľvek tlačítko a hra sa spustí odznova.

```
 10 REM SQUASH
 20 GO SUB 690
 25 REM premiestňovanie pálky pomocou tlačítok "o" a "p"
 27 LET lopta=550
 30 LET pole=300
 35 LET palka=460
 40 BRIGHT 1: BORDER 4
 45 PAPER 7: CLS
 50 LET body=0
 60 GO SUB pole
 70 REM .....................................
 80 PRINT AT 7,15; INK 6; PAPER 2;"BODY: ";body
110 LET a$=INKEY$
130 IF a$="o" OR a$="p" THEN GO SUB palka
140 GO SUB lopta
150 PRINT AT 19,b+11; INK 2;b$
```

```
160 GO TO 80
290 REM .....................................
300 REM hracie pole
310 LET x=1
320 PRINT AT 10,10; INK 1;"##################"
330 FOR t=0 TO 10
340 PRINT AT t+10,10; INK 1;"#"; AT t+10,30;"#"
350 NEXT t
360 LET b$=" "+b$+" "
380 LET y=1
385 LET l=1
390 LET m=1
400 LET b=10
410 PRINT AT 19,11+b;b$
430 RETURN
450 REM ...........................
460 REM pohybovanie palkou
480 IF a$="p" AND b=16 THEN RETURN
490 IF a$="o" AND b=0 THEN RETURN
510 IF a$="p" THEN LET b=b+1
520 IF a$="o" THEN LET b=b-1
530 RETURN
540 REM ...........................
550 REM pohyb lopty
570 PRINT AT 11+y,11+x;" "
580 IF l+x>18 OR l+x<0 THEN LET l=-1: BEEP .01,50
590 IF m+y>8 OR m+y<0 THEN LET m=-m: BEEP .01,20
600 LET x=x+l
610 LET y=y+m
620 PRINT AT 11+y,11+x; INK 4;d$
622 IF y<>8 THEN RETURN
630 IF y=8 AND ABS (b-x)<=2 THEN BEEP .1,1: LET body=body+10: RETURN
640 PRINT AT 5,15; INK 7; PAPER 3; "KONIEC HRY"
650 FOR g=1 TO 50
660 BEEP .01,g: BEEP .01,60-g
670 NEXT g
680 RUN
690 REM dajte medzi uvodzovky pismeno "M" v mode GRAPH !!!
700 LET d$="M"
710 FOR h=0 TO 7
715 BEEP .01,h*h
720 READ n
730 POKE USR "M"+h,n
740 NEXT h
750 REM dajte medzi uvodzovky pismeno "B" v mode GRAPH !!!
760 LET b$-"B"
770 FOR h-0 TO 7
775 BEEP .01,50/(h+1)
780 READ n
790 POKE USR "B"+h,n
795 NEXT h
800 RETURN
1000 DATA BIN 00111100,BIN 01111110,BIN 11111111,BIN 11111111,
     BIN 11111111,BIN 11111111,BIN 01111110,BIN 00111100
1010 DATA BIN 01111110,BIN 11111111,BIN 11111111,BIN 01111110,
```

BIN 00000000,BIN 00000000,BIN 00000000,BIN 00000000

Na prvý pohlad sa Vám bude zdať program komplikovaný. Ak ho však pozorne prepíšete z knihy, po jeho spustení sa vrátite späť k výpisu a prehliadnete si príkazy, budete žasnúť, ako Vám bude všetko zrazu jasné.

F A R B A A G R A F I K A

Vaše Spectrum má celú radu príkazov na prácu s farbou a grafikou, ktorými si môžete skrášliť Vaše programy. Tieto príkazy sú celkom jednoduché a umožňujú vytvárať veľmi zaujímavé efekty.

Pomocou príkazov na zmenu farieb môžeme meniť tri veci:

BORDER - okraj obrazovky
PAPER - vlastnú plochu obrazovky, na ktorú sa zobrazuje
INK - znaky, ktoré na túto plochu píšeme

Na použitie máme 8 farieb, ktoré sú označené príslušnými číslami:

0 - čierna (black)
1 - modrá (blue)
2 - červená (red)
3 - fialová (magenta)
4 - zelená (green)
5 - bledomodrá (cyan)
6 - žltá (yellow)
7 - biela (white)

Celkom jednoduché: čím je nižšie číslo, tým tmavšia je farba. Tiež na čiernobielej obrazovke môžeme rozoznať odstupňované tóny šedi.
Ak zapnete svoje Spectrum, majú PAPER a BORDER číslo 7 a INK číslo 0. Toto predurčí celkom bielu farbu obrazovky, na ktorej vidíme čierne znaky.

Farby PAPER a INK sa dajú použiť dvoma spôsobmi:

1. Pre celú obrazovku. Zadajte riadok:

PAPER 6: INK 2: CLS

Všetky znaky budú teraz zobrazené červene na žltom pozadí.

2. Lokálne použitie. Zadajte NEW a napíšte riadok:

PRINT PAPER 1; INK 7;"HALLO"

Nápis "HALLO" teraz zasvieti ako biele písmená na modrom pruhu. Okolo pásu sa ale farba nezmení.

Rovnako možno použiť zmenu farby v kombinácii s príkazom INPUT.

INPUT INK 2; PAPER 6; "Ako sa voláte? ";a$

Otázka zažiari červeno na žltom pase. Príkazy farieb si lepšie

ukážeme na nasledujucom programe:

```
 20 REM DEMONSTRACIA FARIEB
 30 FOR b=0 TO 7
 40 FOR p=0 TO 7
 50 FOR i=0 TO 7
 60 BORDER b
 70 PAPER p: CLS
 80 INK i
 90 PRINT AT 10,10;"BORDER ";b; TAB 10;"PAPER ";p; TAB 10;"INK ";i
100 FOR u=1 TO 60: NEXT u: BEEP .15,p*i+2*b
110 NEXT i
120 NEXT p
130 NEXT b
```

Tento program ukazuje všetky farebné kombinácie pre BORDER, PAPER a INK a ako vidíte, je treba dosť času, aby sa ukázali všetky možné kombinácie ktorých je 8 * 8 * 8.

Samozrejme mnohé kombinácie su vyložene neatraktívne. (Je logické, že biely okraj, biely podklad a biele písmo je dosť hlupa kombinácia, zvlášť nevhodná pre čítanie). Ine kombinácie sa Vam budu rozhodne viac páčiť.

Vyplatí sa vziať si pero a papier a poznamenať si najvhodnejšie kombinácie, ktoré môžete využiť v dalších programoch.

Mazanie obrazovky (CLS) na riadku 70 je nevyhnutné, ak chceme vymeniť farbu PAPER.

Teraz môžete meniť farby priamo podľa predpísaného postupu. Zapamätajte si: V programe platí príkaz INK alebo PAPER automaticky pre celý displej, ak za nim nasleduje príkaz CLS. Farby sú vymedzené lokálne keď ich použijeme v jednom riadku s príkazmi PRINT alebo INPUT.

Po lokálnom použití príkazu INK sa farba INK vracia späť k pôvodne nastavenej.

Nechajte prebehnúť nasledujúci program. Ukáže Vám pôsobivo, aké môžu byť pekné farby volené funkciou generátora náhodných čísiel.

```
  5 REM PYRAMIDA
 10 BORDER 7
 15 CLS
 20 LET b=16
 50 LET t=0
 60 LET s=0
 70 LET l=20
 80 LET t=t+b
 90 FOR n=s TO s+b*2-2
100 PRINT AT l,n; INK INT (RND*6)+1;"≠"
105 BORDER INT (RND*6)+1
110 NEXT n
120 LET l=l-1
130 LET b=b-1
140 LET s=s+1
150 IF b>0 THEN GO TO 80
155 BORDER 1
160 PAUSE 0
```

Program zobrazí pyramídu zloženú z rôznofarebných znakov "≠". Okraj obrazovky bliká počas celého priebehu a na konci sa nastaví (riadok 155)

na modrý. Riadok 160, ktorý bude vykonávať pauzu až do stlačenia ľubovolného tlačítka je tu preto, aby nebol program ukončený známym hlásením "0 OK" v dolnej časti obrazovky.

Namiesto znaku "-" môžete zadať malý štvorček z hrubej grafiky Spectra. Prevediete to takto: Zapnite mod GRAPH (CAPS SHIFT a 9) a potom stlačte CAPS SHIFT a tlačítko 8. Nakoniec nezabudnite mod GRAPH zasa vypnúť (CAPS SHIFT a 9).

Nahodnú zmenu farieb predvadza aj nasledujúca demonštrácia.

```
10 REM DEMONSTRACIA
15 PAPER 5
17 CLS
20 FOR g=1 TO 100
30 INK RND*7
40 PAPER RND*7
50 BORDER RND*7
70 PRINT AT RND*21,RND*9;"TOTO JE DEMONSTRÁCIA"
80 NEXT g
```

Nasledujúci program "HÁDANIE FARIEB" ukazuje varianty farieb, ktoré necháme vstúpiť do hry. Prebieha podobne ako hra "LOGIC" (Mastermind).

Prepíšte pozorne program z knihy do počítača a vyskúšajte ho. Potom vezmite zasa knihu do ruky a preštudujte si, ako sú farebne a grafické príkazy prevádzané.

```
10 REM HADANIE FARIEB
20 POKE 23690,100
30 DIM c(4)
40 DIM g(4)
50 DIM h(4)
60 INK 0: BORDER 0: PAPER 7: CLS
70 PRINT TAB 1; "Myslim si kombinaciu 4 farieb"
80 PRINT TAB 1; "zvolenu z tychto 6 farieb:"
100 PRINT
105 FOR c=1 TO 6
110 PRINT TAB 4+c; INK 0;c;" = "; INK c;"#"
120 NEXT c
125 PRINT
130 PRINT TAB 1;"Kazda z 4 farieb je ina a mas"
135 PRINT TAB 1;"10 pokusov, aby si ju uhadol."
140 PRINT AT 21,1;"Ak mozme zacat, stlac klavesu!"
150 PAUSE 4E4
160 CLS
170 PRINT AT 0,2;
180 FOR c=1 TO 6
190 PRINT INK 0;c;"="; INK c;"#";"  ";
200 NEXT c
210 PRINT ''
220 LET c(1)=INT (RND*6)+1
230 LET z=1
240 LET z=z+1
250 LET c(z)=INT (RND*6)+1
260 LET j=0
270 LET j=j+1
280 IF c(j)=c(z) THEN GO TO 230
```

```
290 IF j<z-1 THEN GO TO 270
300 IF z<4 THEN GO TO 240
310 FOR g=1 TO 10: POKE 23692,-1
320 PRINT AT 2,1; INK 0;"Hadaj kombinaciu mas ";g;". pokus!"
330 INPUT "Vloz 4 cisla (1-6): ";a
350 FOR z=1 TO 4
360 LET g(z)=a-10*INT (a/10)
370 LET h(z)=g(z)
380 LET a=INT (a/10)
390 NEXT z
400 LET b=0: LET w=0
410 FOR z=1 TO 4
420 IF c(z)<>g(z) THEN GO TO 450
430 LET b=b+1: BEEP .2,b*15
440 LET g(z)=0
450 NEXT z
460 FOR z=1 TO 4
470 IF g(z)=0 THEN GO TO 520
480 FOR j=1 TO 4
490 IF c(z)<>g(j) THEN GO TO 510
500 LET w=w+1: BEEP .2,60-b*15
510 NEXT j
520 NEXT z
525 PRINT AT g+4,1;
530 FOR t=4 TO 1 STEP -1
540 PRINT INK h(t);"■";
550 NEXT t
560 PRINT INK 0;"  ";b;" x cierna";"   ";
580 PRINT w;" x biela"
610 IF b=4 THEN PRINT '" Vyborne uhadol si na ";g;". pokus!"
630 IF b<>4 THEN NEXT g
650 PRINT '"' Kombinácia bola: ";
660 FOR h=1 TO 100: NEXT h
670 FOR t=4 TO 1 STEP -1
680 FOR h=1 TO 50: NEXT h
690 BEEP .2,t*10: PRINT INK c(t);"■";
700 NEXT t
710 FOR h=1 TO 60: BEEP .01,h: NEXT h
720 POKE 23692,-1
730 PRINT '"' Chces hrat znova ? (A/N)"
740 LET a$=INKEY$: IF INKEY$="" THEN GO TO 740
750 IF CODE a$<>CODE "n" AND CODE a$<>CODE "N" THEN RUN
760 CLS
770 PRINT '' INK RND*6; TAB RND*15;"O.K. az potom..."
780 POKE 23962,-1
790 FOR h=1 TO 25
800 NEXT h
810 IF INKEY$="" THEN GO TO 770
820 RUN
```

Riadok 20 (POKE 23962,100) meni zvuk, ktorý počujeme pri stlačení tlačitka, na trochu dlhšie pipnutie.

Riadok 60 nastavi okraj obrazovky a všetky znaky čierne (0) a pozadie biele (7). Program na riadkoch 105 až 120 kresli kocku pre každú farbu v diagonálnej linii a k tomu číslo, prislúchajúce farbe.

Riadok 150 nastavuje dlhu pauzu, ktoru prerušíme ak stlačíme niektoré tlačítko.

Riadky 180 a 200 ukážu šesť farieb s ich číslami na obrazovke.

Riadok 210 posuva PRINT poziciu pomocou apostrofu o dva riadky nižšie.

Program v riadkoch 220 až 300 generuje nahodnu kombinaciu farieb pre našu hru a stara sa, aby sa jedna farba neobjavila dvakrát za sebou.

Riadok 310 nastavuje slučku pre 10 pokusov hadania. Jeho druhá polovica (POKE 23692,-1) zabezpečí, že ak je obrazovka plná, text sa automaticky odroluje.

Riadok 320 vypisuje číslo pokusu, na ktorý skúšame kombináciu uhádnuť a príkaz INPUT v riadku 330 si od nás vypýta štyri čísla farebnej kombinacie, pričom čísla farieb zodpovedajú číslam na klavesnici Spectra.

Vaše pokusy zadavajte jednoducho vo forme štvormiestneho čísla, každé číslo pre jednu farbu, pričom sa jedna farba nesmie opakovať.

Program uložený v 350 až 390 riadku, rozkladá Vaše štvorčíslo opäť do jednotlivých čísiel. Na riadku 400 vynuluje premenné "b" pre čierne a "w" pre biele koliky.

Potom počítač porovná Vašu kombináciu s tou, ktoru "si mysli" a vyhodnoti ju takto: Za každú uhadnutu farbu na správnom mieste dostanete čierny kolík a za každú farbu, ktorú ste síce uhadli ale umiestnili ste ju na nesprávne miesto dostanete biely kolík. Za nesprávnu farbu nedostanete nič. Vašou úlohou je uhádnuť správnu kombináciu na čo najmenší počet pokusov.

Pozn. koliky v tomto prípade nie sú znázornené graficky, ale vypisujú sa len slovom ako napr. 1 x čierna 3 x biela.

Už dopredu viete, že ak napíšete PRINT AT 4,7; "TEST", slovo TEST sa vypíše na 4. riadok a 7. stĺpec smerom doprava. ASCII kód 22, ktorý je riadiaci znak pre obrazovku vykoná to isté, ale s určitým rozdielom v zadávaní. Pre rovnaký efekt ako PRINT AT musíte zadať:

```
PRINT CHR$ 22+CHR$ 4+CHR$ 7;"TEST"
```

ZX Spectrum umožňuje spojenie viac reťazcov pomocou znamienka +. Pozrite sa, ako to funguje v nasledujúcom príklade:

```
10 LET a$=CHR$ 22+CHR$ 4+CHR$ 7
20 PRINT a$;"TEST"
```

Pre TAB je taktiež dané elegantné riešenie pomocou funkcie CHR$. Spojte CHR$ 23 dohromady s reťazcom CHR$ n, pričom n je číslo pozície + 1, na ktorú chcete posunúť doprava PRINT obsah. Na to sa pozrieme v budúcom príklade.

Pri tomto spôsobe písania nebude vypisovaný prvý znak nasledovaný medzerou. Napíšete síce v zátvorkách "x TEST", ale na obrazovke sa objaví iba slovo "TEST". Toto môžete využiť napríklad na číslovanie reťazcov.

```
10 LET a$=CHR$ 23+CHR$ 4
20 PRINT a$;"x TEST"
30 GO TO 20
```

Na začiatku tejto kapitoly sme písali o 8-mich farbách a ich aplikácii v príkazoch BORDER, PAPER a INK. Tieto farebne kombinácie môžu

byť ešte modifikovane dvojakými príkazmi: BRIGHT a FLASH. Ako a naćo sa používajú, ukazuje nasledujúci príklad, ktorý zadajte do počítača a spustite pomocou príkazu RUN.

```
40 PRINT INK 4;"NORMAL       "
45 PRINT BRIGHT 1; INK 4;"BRIGHT       "
50 PRINT INK 4; PAPER 2;"NORMAL       "
55 PRINT BRIGHT 1; INK 4; PAPER 2;"BRIGHT       "
60 PRINT FLASH 1; INK 4;"FLASH       "
65 PRINT BRIGHT 1; FLASH 1; INK 4;"BRIGHT+FLASH"
70 PRINT FLASH 1; PAPER 2; INK 4;"FLASH       "
75 PRINT BRIGHT 1; FLASH 1; PAPER 2; INK 4;"BRIGHT+FLASH"
```

Teraz už vidíte, že príkaz BRIGHT zjasní farbu, čím dostávame na Spectre vlastne 16 farieb. Príkaz FLASH nám zase ľubovoľnú farebnú kombináciu rozblíka, čo je vhodné napríklad na písanie dôležitých hlásení.

Už viete, že je možné pre nastavenie farby BORDER, PAPER a INK použiť čísla 0-7. Pri príkazoch PAPER a INK je však možné použiť aj dve zvyšné číslice. Číslo 8 znamená, že bude použitá posledná nastavená farba. Číslo 9 nastavuje kontrastnú farbu, čiže pre tmavý podklad biele písmená a pre svetlý podklad čierne písmená a naopak.

Na nasledujúcom príklade si ukážeme použitie hodnoty 9 pre nastavenie farby INK a taktiež sa zoznámime s novým príkazom OVER.

```
60 PAPER RND*6
70 CLS
80 INK 9
100 FOR g=1 TO 300
110 PRINT AT RND*20,RND*30; CHR$ (65+INT (RND*26));
120 NEXT g
130 GO TO 60
```

Ak program vyskúšate, zameňte riadok 110 za tento:

```
110 PRINT AT RND*20,RND*30; OVER 1; CHR$ (65+INT (RND*26));
```

Príkaz OVER spôsobuje to, že nové písmená nevymazávajú staré, ale prepisujú sa cez ne, takže výsledný znak je prekrytím obidvoch písmen.

Skúste si nasledujúci program. Demonštruje opäť použitie príkazu OVER, tentokrát na dvojiciach konkrétnych znakov, ktoré si zadáte. Ťažko predvídať výsledok, ale niektoré kombinácie sú celkom zaujímavé. Napr. malé "o" a malé "w" v kombinácii dajú znak podobný veľkému "T". Zaujímavé sú aj kombinácie "[" + "]" alebo "%" + "/".

```
10 OVER 1
20 FOR g=1 TO 16
30 INPUT "zadaj 1. pismeno: ";a$
40 INPUT "zadaj 2. pismeno: ";b$
50 PRINT AT g,g;a$; CHR$ 8;b$
60 NEXT g
```

Spomeňte si, že príkazy PRINT AT a TAB môžeme nahradiť použitím príkazov CHR$ 22 a CHR$ 23 tak, že takýto príkaz pripojíme k reťazcu, ktorý chceme zobraziť. S týmto postupom sme sa už viackrát pri zostavovaní programov stretli. Pomocou príkazu CHR$ nasledovaného riadiacim znakom

môžeme však meniť aj farbu a to nasledujúcimi riadiacimi znakmi:

```
CHR$ 16 = INK        CHR$ 17 = PAPER     CHR$ 18 = FLASH
CHR$ 19 = BRIGHT     CHR$ 20 = OVER
```

Za nimi nasleduje vždy kód, ktorý je príslušný pre zvolenú farbu. Ako sa to dá využiť v praxi uvidíte na nasledujúcom programe:

```
20 INPUT PAPER 6; INK 1;"zadaj farbu INK: ";ink
30 INPUT INK 2;"zadaj farbu PAPER: ";paper
40 INPUT FLASH 1; BRIGHT 1; INK 4; PAPER 2;"vloz slovo: ";a$
50 LET a$=CHR$ 16+CHR$ ink+CHR$ 17+CHR$ paper+a$
60 PRINT AT 10,10;a$
```

Všetko, čo sa týka riadiacich znakov, je obšírne vysvetlené v príručke ZX Spectra. V tabulárnom usporiadaní tam nájdete všelijaké možnosti, ktoré môžu vyvolať rôzne optické efekty.

Ak sa chcete pozrieť, koľko bláznivých farebných efektov môžete vyvolať celkom jednoduchým programom, spustite nasledujúci program.
Pre prípad, že by Vás intenzita pípania znervóznovala, vymažte jednoducho riadky 90 a 100. Ak by bol program pre Vás príliš pomalý, zmeňte hodnotu 7 v riadku 40 na 6 a tak nebude biely znak vytlačený.

```
 10 PAPER 7: BORDER 0: CLS
 20 LET a=RND*10
 30 LET b=RND*16
 40 LET z=RND*7
 50 PRINT AT a,b; INK z;"#"
 60 PRINT AT 21-a,b; INK z;"#"
 70 PRINT AT 21-a,31-b; INK z;"#"
 80 PRINT AT a,31-b; INK z;"#"
 90 IF RND>RND THEN GO TO 20
100 BEEP RND/30,RND*60-RND*60
110 GO TO 20
```

Keď sa už dostatočne vynadívate, zmeňte program takto:

```
 10 PAPER 7: BORDER 0: CLS
 20 LET a=RND*10
 25 LET f=RND
 30 LET b=RND*16
 40 LET z=RND*17
 50 PRINT AT a,b; FLASH f; BRIGHT 1; INK z;"#"
 60 PRINT AT 21-a,b; FLASH f; BRIGHT 1; INK z;"#"
 70 PRINT AT 21-a,31-b; FLASH f; BRIGHT 1; INK z;"#"
 80 PRINT AT a,31-b; FLASH f; BRIGHT 1; INK z;"#"
 90 IF RND>RND THEN GO TO 20
100 BEEP RND/30,RND*60-RND*60
110 GO TO 20
```

Vidíte, že každý znak je teraz v BRIGHT móde zjasnený a v priebehu programu je zabudované náhodne volené blikanie.
Ak chcete do príkazu FLASH zadávať náhodné číslo, nemusíte číslo náhodne vygerované pomocou funkcie RND zaokrúhľovať príkazom INT. Náhodné

číslo na riadku 25 bude v tomto prípade automaticky zaokrúhlené.

Teraz si prezrite program "Grécka písmenková polievka". Prečo je takto pomenovaný, pochopíte, až ho spustíte.

```
 5 REM "GRECKA PISMENKOVA POLIEVKA"
10 PAPER 7: BORDER 0: CLS
20 LET a=RND*10
25 OVER 1
30 LET b=RND*16
40 LET z=RND*7
45 LET a$=CHR$ (65+RND*26)
50 PRINT AT a,b; BRIGHT 1; INK z;a$
60 PRINT AT 21-a,b BRIGHT 1; INK z;a$
70 PRINT AT 21-a,31-b; BRIGHT 1; INK z;a$
80 PRINT AT a,31-b; BRIGHT 1; INK z;a$
90 GO TO 20
```

P L O T

Ako uvidíte v programoch "galaxia" alebo "sinusoida" prinášа PLOT príkaz výkonnú grafiku na Vašu obrazovku. Ako už pravdepodobne viete, jemná rozlišiteľnosť Spectra obsahuje 256 x 176 bodov. Farebné rozlíšenie je však len pre 32 x 24 štvorcov hrubej grafiky. Každý takýto (atribútový) štvorec obsahuje 8 x 8 bodov jemnej grafiky a môže obsahovať maximálne dve farby. To znamená, že farby budú tak povediac nanášané na nakreslený obraz. Napriek tomuto malému nedostatku v grafike Spectra, možno ešte vždy docieliť zaujímavé farebné efekty.

```
 10 REM GALAXIA
 20 PAPER 0: BORDER 0: CLS
 30 LET c=255: LET d=175
 40 INK RND*7
 50 LET a=c*RND
 60 LET b=d*RND
 70 PLOT a,b: PLOT a,d-b
 80 PLOT c-a,b: PLOT c-a,d-b
 90 IF RND>.5 THEN GO TO 60
 95 INK RND*7
100 GO TO 50
```

```
 10 REM SINUSOIDA
 30 BORDER 2: CLS
 40 FOR x=0 TO 63 STEP .5
 50 LET y=20*SIN (x/32*PI)
 60 IF y=0 THEN GO TO 100
 70 FOR n=0 TO y STEP SGN y/4
 80 PLOT INK RND*6; x*3+30,3*(n+30)
 90 NEXT n
100 BEEP .1,x: NEXT x
```

D R A W

Naš nasledujúci program "Sklenený močiar" ukazuje zaujímavu grafiku s použitím príkazu DRAW. Na riadku 30 sa nastaví biely podklad. Farby znakov sú náhodne vygenerované na riadkoch 50 a 60. Riadok 70 bezpečne zaistí, že budú farby rozdielne. Riadkom 100 bude vymazaná obrazovka a potom bude určený náhodne súradnicový pár. V strede obrazovky bude nakreslený bod na náhodne vypočítané súradnice.

Riadok 140 s príkazom DRAW určí, ako bude priamka dlhá a ktorým smerom bude vedená. Celý program môže fungovať len v tom prípade, keď je predom pevne určený štartovací bod čiary. Toto sa vykoná pomocou príkazu PLOT na riadku 130.

```
 10 REM SKLENENY MOCIAR
 30 PAPER 7
 50 LET a=INT (RND*8)
 60 LET b=INT (RND*7)
 70 IF a=b THEN GO TO 60
 80 BORDER a
 90 INK b
100 CLS
110 LET c=INT (RND*256)-128
120 LET d=INT (RND*172)-85
130 PLOT 128,86
140 DRAW c,d
145 BEEP .01,RND*100-50
150 IF RND>.02 THEN GO TO 110
160 RUN
```

Príkaz DRAW nakreslí rovnú priamku, ak za DRAW nasledujú dve hodnoty. Pomocou tohoto príkazu sa však dajú kreslit aj krivky a to tak, že dosadíme tretiu hodnotu, ktorá udáva zakrivenie. Program "Krivkový zlom" je principiálne identický "Sklenenému močiaru", až na riadok 140, kde číslo PI / 2 nastavuje uhol zakrivenia.

```
 10 KRIVKOVY ZLOM
 30 PAPER 7
 50 LET a=INT (RND*8)
 60 LET b=INT (RND*7)
 70 IF a=b THEN GO TO 60
 80 BORDER a
 90 INK b
100 CLS
110 LET c=INT (RND*256)-128
120 LEt d=INT (RND*172)-85
130 PLOT 128,86
140 DRAW c,d,PI/2
145 BEEP .01,RND*100-50
150 IF RND>.015 THEN GO TO 110
160 RUN
```

C I R C L E

Je grafický príkaz, pomocou ktorého sa dá kreslit kruh. Program

"Tunel" zvoli svetlomodru farbu pozadia a bielu farbu obrazovky a kresli kruhy nahodne volenymi farbami s nahodne volenym stredom. Prvy parameter pre prikaz CIRCLE je x-ova a druhy y-ova suradnica stredu kruhu, tretie cislo urcuje polomer.

```
10 REM TUNEL
15 BORDER 5: PAPER 7: CLS
20 CIRCLE INK RND*6;128+RND*10-RND*10,86+RND*7-RND-7,RND*65
30 IF RND>.92 THEN CLS
40 BEEP RND/4,RND*100-30
50 GO TO 20
```

```
10 REM TUNEL S PAVUCINOU
20 CIRCLE INK RND*6;128+RND*10-RND*10,86+RND*7-RND-7,RND*65
25 PLOT 128,86
30 DRAW INK RND*6; OVER 1; 32-RND*64,21-RND*42
40 BEEP RND/4,RND*50-30
50 IF RND>.33 THEN GO TO 30
60 IF RND<.93 THEN GO TO 20
70 RUN
```

A este niekolko programov, ktore Vam chcu ukazat rozne graficke možnosti Vaseho pocitaca:

```
10 REM NAHODNE FARBY
20 BORDER 2: PAPER 7: CLS
30 PRINT INK RND*5+1; AT RND*21,RND*31;"#"
40 GO TO 30
```

```
10 REM STVOREC
15 BORDER 1+RND: PAPER 7: CLS
20 RANDOMIZE
25 LET p=85: LET s=45
30 LET x=RND*p: LET y=RND*p
35 INK RND*6
37 FOR g=1 TO RND*6
40 PLOT p+x+s,p+y
50 PLOT p+y+s,p+x
60 PLOT p-x+s,p+y
70 PLOT p+y+s,p-x
80 PLOT p-x+s,p-y
90 PLOT p-y+s,p-x
100 PLOT p+y+s,p-y
110 PLOT p-y+s,p+x
120 LET x=x+RND+RND-1
130 LET y=y+RND+RND-1
135 NEXT g
140 IF RND<.2 OR ABS x>p OR y>p THEN GO TO 40
150 IF RND<.1 THEN GO TO 25
160 GO TO 30
190 RUN
```

```
10 REM VODNY VIR
20 BORDER 7: PAPER 7: INK 0: CLS
50 FOR a=1 TO 10
```

```
 60 FOR j=1 TO 7 STEP .1
 70 PLOT a*j*COS (j)+110,a*j*SIN (j)+100
 80 NEXT j
 90 INK a/2
100 NEXT a
```

```
 10 REM SLNKO
 20 BORDER 7: PAPER 7: INK 0: CLS
 50 FOR a=5 TO 19
 60 FOR j=1 TO 7 STEP .3
 70 PLOT a*j*COS (j)/2+110,a*j*SIN (j)+100
 80 NEXT j
100 NEXT a
```

```
 10 REM SILOVE POLE
 15 REM (program prebieha velmi pomaly)
 20 BORDER 7: PAPER 7: INK 0: CLS
 50 FOR a=2 TO 19
 60 FOR j=1 TO 7 STEP .01
 70 PLOT a*(j+4*SIN (j))*COS (j)+110,a*(j+2*SIN (j))*SIN (j)+110
 80 NEXT j
 90 INK a/3.5
100 NEXT a
```

K R E S L E N I E L Í N I Í

Tento program vypočítava na obrazovke dva body (súradnice x,y a 1,m) a kresli medzi nimi línie.

Tu je popis práce jednotlivých riadkov:

```
  5 voli náhodnú farbu línií na bielom pozadí
 10 voli náhodne x súradnicu pre prvý bod
 20 voli náhodne súradnicu y pre prvý bod
 30 voli náhodne 1 súradnicu pre druhý bod
 40 voli náhodne súradnicu m pre druhý bod
 50 voli premenné tempo pre body
 60 definuje generátor náhodných funkcií
100 volá podprogram na riadku 1000; určuje náhodné tempo zobrazovania
115 premenná "teraz" určuje začiatočné číslo kroku, skôr, ako je
    zvolený rozptyl bodov
116 ak teraz=0 bude volit podprogram v riadku 1000 nové tempo
120 zobrazenie bodu na súradniciach x,y
130 prepojenie bodov x,y a 1,m priamkou
140 ak je súradnica x na okraji obrazovky, nastane zmena smeru
150 viď riadok 140
160 viď riadok 140
170 viď riadok 140
180 pričíta tempo prvého bodu k jeho súradniciam
190 to isté druhý bod
210 vygeneruje náhodné číslo medzi 0 a 199
220 keď cont=1, program sa spusti odznova
230 keď je premenná cont rôzna od hodnoty 1, kresli sa nová línia
1000 udáva pre x-ovu súradnicu prvého bodu náhodne tempo
```

```
1010 vid riadok 1000
1020 vid riadok 1000
1030 vid riadok 1000
1040 volí náhodne číslo "teraz", a zaisťuje, aby toto číslo nebolo 0
1050 vracia sa späť do hlavného programu
```

Vplyvom zmeny veľkosti u a v na riadku 50 môžeme zväčšovať alebo zmenšovať rozostup medzi jednotlivými líniami. Zaujímavý je výsledok, ak príkazu DRAW v riadku 130 pridáme tretiu hodnotu, čiže bude kreslená krivka. Ale pozor, pri tejto variante môže ľahko nastať chyba "B Integer out of range".

```
  1 REM KRESLENIE LINII
  5 BORDER 7: PAPER 7: INK RND*4: CLS
 10 LET x=INT (RND*256)
 20 LET y=INT (RND*176)
 30 LET l-INT (RND*256)
 40 LET m=INT (RND*176)
 50 LET u=15: LET v=7
 60 DEF FN r(x)=INT (RND*x)
100 GO SUB 1000
110 REM opakovanie
115 LET teraz=teraz-1
116 IF teraz=0 THEN GO SUB 1000
120 PLOT x,y
130 DRAW l-x, m-y
140 IF x+a>255 OR x+a<0 THEN LET a=-a
150 IF y+b>175 OR y+b<0 THEN LET b=-b
160 IF l+c>255 OR l+c<0 THEN LET c=-c
170 IF m+d>175 OR m+d<0 THEN LET d=-d
180 LET x=x+a: LET y=y+b
190 LET l=l+c: LET m=m+d
200 REM ohranicenie
210 LET cont=FN r(200)
220 IF cont=1 THEN RUN
230 GO TO 110
1000 LET a=FN r(u)-v
1010 LET b=FN r(u)-v
1020 LET c=FN r(u)-v
1030 LET d=FN r(u)-v
1040 LET teraz=FN r(20)+10
1050 RETURN
```

K O B E R C O V Ý V Z O R

Tento program kreslí zaujímavé kobercové vzory. Jeho trik spočíva v tom, že séria línií vychádzajúcich zo stredu obrazovky a vedených k niektorému bodu na okraji, je prepisovaná príkazom OVER a tým vzniká efekt, ktorý vidíme na obrazovke.

```
  1 REM KOBERCOVY VZOR
  5 BORDER 0: PAPER 0: INK 7: CLS
 10 FOR x=0 TO 255
 20 PLOT x,0
```

```
30 DRAW OVER 1;255-x*2,175
40 NEXT x
50 FOR y=0 TO 175
60 PLOT 0,y
70 DRAW OVER 1;255,175-y*2
80 NEXT y
```

Ak chcete zvolit iny stredový bod, ako je stred obrazovky, v tom prípade musíte upraviť cyklus na riadkoch 10 a 50. Môžete prípadne aj upraviť STEP na hodnotu okolo 4. Program potom kreslí ešte krásnejšie vzory.

Nasledujúci program simuluje otáčanie trojuholníkom v priestore.

```
  5 REM OTACANIE TROJUHOLNIKA
  9 BORDER 0: PAPER 0: INK 6: CLS
 10 LET l=RND*256
 20 LET m=RND*176
 30 LET n=RND*256
 40 LET o=RND*176
 50 LET p=RND*256
 60 LET q=RND*176
 70 LET a=RND*5
 80 LET b=RND*5
 90 LET c=RND*5
100 LET d=RND*5
110 LET e=RND*5
120 LET f=RND*5
130 REM opakovanie
140 PLOT l,m
150 DRAW n-l,o-m
160 DRAW p-n,q-o
170 DRAW l-p,m-q
171 FOR x=1 TO 50: NEXT x
180 IF l+a>255 OR l+a<0 THEN LET a=-a
190 IF m+b>175 OR m+b<0 THEN LET b=-b
200 IF n+c>255 OR n+c<0 THEN LET c=-c
210 IF o+d>175 OR o+d<0 THEN LET d=-d
220 IF p+e>255 OR p+e<0 THEN LET e=-e
230 IF q+f>175 OR q+f<0 THEN LET f=-f
240 CLS
250 LET l=l+a: LET m=m+b
260 LET n=n+c: LET o=o+d
270 LET p=p+e: LET q=q+f
280 REM ohranicenie
290 GO TO 130
```

P O I N T

Funkcia POINT dokáže zistiť, či na určitom mieste obrazovky je alebo nie je bod.

Tento program Vám ukaže, ako funkcia pracuje. Pomocou prikazu PLOT kreslíme na obrazovku množstvo bodov (ich suradnice su zvolené náhodným procesom). Potom skúšame funkciou POINT na práve tak náhodne zvolených

suradniciach, či tam nie su predom zobrazene body. Zakaždym, keď je takýto bod nájdený, počítač pipne. Výsledok uspešne najdených bodov pri takomto hladaní robi sotva 1%.

```
10 REM PLOT A POINT
20 FOR a=1 TO 100
30 PLOT RND*100,RND*100
40 NEXT a
50 LET citac=0
60 FOR g=1 TO 1000
70 IF POINT (RND*100,RND*100)=1 THEN BEEP 1,1: LET citac=citac+1:
   PRINT AT 1,1;"Uspesnost je ";citac/g*100;"%    ";AT 3,1;"Hladanie
   cislo: ";g
80 NEXT g
```

Ak je na súradniciach x,y bod, POINT (x,y) =1, inak POINT (x,y)=0.

T L A Č I A R E Ň

Na Spectre máme tri príkazy, ktoré spolupracujú s tlačiarňou.

LLIST, LPRINT a COPY

LLIST - Tento príkaz funguje rovnako ako LIST, s tým rozdielom, že listing programu sa nezobrazí na obrazovke, ale na tlačiarni. Nemôžeme bohužiaľ docieliť, aby počítač vytlačil len časť programu.

LPRINT - S týmto príkazom dosiahneme to isté, čo s príkazom PRINT. Výstup ide samozrejme aj tentokrát na tlačiareň a nie na obrazovku. Príkaz môžeme použiť priamo napr. LPRINT "Halo" alebo v programovom riadku, napr. 10 LPRINT "Odpoved je ";a

COPY - Tento príkaz vytlačí presnú kópiu obrazovky čo sa dá využívať napr. pre tlačenie grafov a obrázkov. Pretože však tlačiareň nemôže reprodukovať farby, ak je obrázok farebný, výsledna kópia nemusí byť úplne zhodná s originálom na obrazovke.

N Á H O D N É Č Í S L A

Generátor náhodných čísiel je súčasťou Vášeho počítača. V tejto kapitole chceme krátko nahliadnuť, ako su nahodne čísla vytvárané. Predvedieme si to na niekoľkých jednoduchých programoch.

Pomocou príkazu RND je vytváraně kladné desatinné číslo, v intervale medzi číslami 0 - 1, pričom hodnotu 1 nikdy nedosiahne. Vyskúšajte si:

```
10 PRINT RND
20 GO TO 10
```

Ak program spustite, obdržíte listing podobný tomuto:

```
0.90386963
0.97032898
```

```
.....
.....
0.93461609
```

Existuje však mnoho aplikacii, kde potrebujeme generovat väčšie nahodne čísla, ako je rozsah medzi 0 a 1. Prirodzene existuje spôsob pre ich získanie. Zadajte napriklad tento prikaz:

```
PRINT INT (RND*30)+1
```

Číslo, ktoré dostanete, je náhodne zvolené medzi 0 a 30. Vyskúšajte si nasledujuci program:

```
10 REM CELE RND CISLA
20 LET a=INT (RND*100)+1
30 PRINT TAB 8;a
40 GO TO 20
```

Počítač berie čísla vo vnútri zátvoriek za argument funkcie a vypočitava náhodné čísla medzi číslom 1 a danym horným ohraničením. To, že ide až po hodnotu 100, sme docielili pripočitaním čísla 1. Takto získané čísla sú kladné. Ak potrebujeme záporné čísla, stačí dosadiť pred funkciu INT znamienko "-".

Generátor náhodných čísiel môže vytvoriť nekonečné varianty vo Vašich programoch. Náš nasledujúci program simuluje vrh šesťstrannou hracou kockou. Vyskúšajte si ho.

```
10 REM KOCKA
20 PRINT "Kolko krat chcete hadzat kockou?"
40 INPUT a
50 CLS
60 PRINT "Vysledky vrhania kockou pri ";a;" pokusoch:"
70 for b=1 TO a
80 LET c=INT (RND*6)+1
90 PRINT c
100 NEXT b
```

B Ý Č Í Z Á P A S

Tu je jeden jednoduchý program, ktorý pre Vás demonštruje chovanie generátora náhodných čísel. Nie je to nejaká komplikovaná hra, ale na demonštráciu funkcie RND postačí a môžete sa pri nej celkom príjemne pobaviť. Vyskúšajte si hru skôr, než si prečítate vysvetlenie k programu. Budete prekvapený, ako ste sa už medzitým veľa naučili.

V našom programe sa stanete toreadorom, ktorý bude 10 x napadnutý nebezpečným býkom. Ako dobre vyviaznete, je závislé na nahode. Pri uhýbaní sa býkovi volíte jednu s troch ciest. Býk si vyberie náhodne tiež jednu z nich. Ak sú obe čísla od seba rôzne, podarilo sa Vám vyhnúť sa býkovi. Vtedy získate 100 bodov a pokračujete v zápase. V opačnom prípade Vás býk naberie na rohy a skončili ste. Na konci sa dozviete Vaše bodové hodnotenie.

Pri zapise programu do počítača dávajte pozor na apostrof v riadkoch 120,140,150,220 a 240. (Zobrazí sa Vám po stlačeni tlačitok SYMBOL SHIFT a

7). Apostrof posuva (ako PRINT prikaz) print poziciu o riadok nižšie.

```
 10 REM BYCI ZAPAS
 20 LET body=100
 30 FOR g=1 TO 10
 40 PRINT AT 4,2; INK 2;"Nahana ta byk!"
 60 PRINT TAB 2;"Ktoru cestu zvolis? (1-3)"
 70 INPUT a
 80 IF a<1 OR a>3 THEN GO TO 70
 90 LET b=INT (RND*3)+1
100 IF a=b THEN GO TO 220
120 PRINT ' INK 4;TAB 2;"Unikol si ";g;". krat."
130 BORDER RND*7
140 PRINT '''INK 2;TAB 2;"Byk isiel cestou ";b
150 PRINT 'INK 4;TAB 2;"Ty si zvolil cestu ";a
155 PAUSE 100
160 FOR h=1 TO 10
170 BEEP .1,RND*50-RND*50
180 NEXT h
190 CLS
200 NEXT g
210 GO TO 230
220 PRINT ' INK 2;TAB 2;"Ako toreador si skoncil!"
230 FOR t=1 TO 50: BEEP .05,RND*50: NEXT t
235 PAUSE 20
240 PRINT ' INK 2,TAB 2;"Dosiahol si ";body*(g-1);" bodov"
```

Teraz si v krátkosti vysvetlime jednotlivé riadky programu.

```
 20 Premennej body sa priradí hodnota 100.
    (O premenných budeme pojednávať v nasledujúcej kapitole)
 30 Pomocou slučky FOR-NEXT počítame desať hracích kôl
 40 Vypíše informačný text.
 60 Pýta sa hráča na cestu, ktorú si vyberie.
 70 Príkazom INPUT berie vložené číslo.
 80 Kontroluje, či je číslo skutočne medzi 1 a 3. V prípade, že nie,
    vracia sa na riadok 70.
 90 Vypočíta náhodné číslo pre cestu býka, medzi hodnotami 1 a 3.
100 Porovnáva a (hráčovu cestu) a b (býkovu cestu). Ak sú zhodné skoči
    na riadok 220 (koniec hry).
120 Hovorí hráčovi, koľkokrát unikol býkovi.
130 Mení náhodne farbu BORDER (okraj).
140 Pripomína hráčovi cestu, ktorou išiel býk.
150 Pripomína hráčovi cestu, ktorú si vybral on.
160 Krátka hudba v prestávke pred nasledujúcim kolom stretnutia.
190 Maže obrazovku.
200 Program skoči späť na začiatok pre ďalšie kolo stretnutia.
210 Sem sa program dostane len ak hráč prežil všetky útoky býka.
220 Vypíše správu pri neúspešnom zakončení hry.
230 Krátka pauza pre hudbu
240 Zobrazenie bodového hodnotenia.
```

Pri čítaní týchto vysvetliviek a ich porovnávaní s listingom ste sa zase trocha priučil programovaniu. Prirodzene sa nájde jeden alebo i viac príkazov, ktoré su Vám ešte stale nejasne, ale pri skúšaní ďalších

programov Vám bude všetko čoraz jasnejšie a jedneho dňa zistíte, že ste už vyrastol z detskych topanok programovania.

ČÍSELNÉ A REŤAZCOVÉ PREMENNÉ

Určite už mate väčšiu istotu z predchádzajúceho programovania a ovladate už bezpečne číselne hodnoty. Pre poriadok si však musíme povedať niekoľko dôležitých pravidiel, ktoré sa vztahuju na premenné. Písmená "a" v riadku 70 a "b" v riadku 90 nadobúdajú vždy v určitom okamžiku hodnotu medzi 1 a 3. Takéto písmená, ktoré môžu obsahovať rôzne (premenne) hodnoty, sa výstižne nazývaju premennými.

Musíme rozlišovať medzi dvoma spôsobmi premenných, číselnou premennou a reťazcovou premennou.

Ako číselna premenná môže slúžiť každa kombinácia písmen a čísel ak začína písmenom. "a" je práve tak prípustná premenna ako "d17", alebo "jasompremmennacislodvestostyridsatosem". Nie su dovolené premenné "2krat" alebo "1e7" (pretože začínajú číslom). V názvoch číselných premenných sa tiež nesmú používať matematické symboly a špeciálne znaky ako napr. "+#@~%&*<>=:". Ostatné povolene premenné môžu slúžiť pre čísla akéjkoľvek veľkosti, ktoré dokáže počítač spracovať. Spectrum ignoruje pritom medzery medzi jednotlivými časťami premennej (napr. "novy den" je to isté, čo "novyden") a nerozlišuje medzi malými a veľkými písmenami ("a" je to iste ako "A").

Reťazcová premenná (string) môže byť označená výhradne jedným písmenom z abecedy "a" - "z" nasledovaným znakom "$". Napr. "a$", "w$". Takáto reťazcová premenná môže byť naplnená ľubovolným písmenom, číslom alebo ich kombináciou. Ak je však naplnená číslom, nie je brana do uvahy jeho numerická hodnota, ale je brane iba ako text.

Ako už vieme, môžeme používať počítač pre písanie veľkých čísel vedeckou notáciou teda, že napíšeme základ čísla, potom symbol "E" (ako Exponent) a potom číslo, ktoré je násobkom 10.

Zadajte nasledujúci demonštračný program, ktorý Vám ukáže používanie premennej pri zavedení čísla, ktore je vždy násobené 10-timi a potom je zobrazený výsledok.

```
10 REM VEDECKA NOTACIA
20 LET a=1234
30 PRINT a
40 LET a=10*a
50 GO TO 30
```

Na obrazovke sa Vám zobrazí:

```
1234
12340
123400
1234000
12340000
1.234E+8
1.234E+9
1.234E+10
```

```
.........
.........
1.234E+38
```

V našom programe ste si iste všimli niekoľko ďalších zasadných vecí. Premenným sa prideľuje ich hodnota pomocou príkazu LET, nasledovaného menom premennej, rovnítkom a hodnotou premennej. Tak príkaz LET a=99 priradí premennej "a" hodnotu 99.

Ako vidíte v našom príklade na riadku 40, do premennej možno dosadzovať aj tú istú premennú, alebo jej násobky a podobne. Táto vlastnosť sa mnohokrát ukáže ako veľmi užitočná.

C V R Č K O V P R O G R A M

Je daná závislosť medzi vonkajšou teplotou a frekvenciou, koľkokrát cvrčky každú minútu zacvrlikajú. Nasledujúci program ukazuje pri dlhodobej variácii priebeh a vypočíta cvrčiace číslo za minútu v závislosti na teplote (vo Fahrenheitoch). Vidíte ako cvrčiace číslo začína v 20-tom riadku vložením konštanty 80. Teplotná relácia je zadaná na riadku 30. Výsledok bude zobrazený na obrazovke riadkom 40. Na riadku 60 je generovaná cvrčiaca premenná náhodným číslom medzi 1-7. Po krátkej pauze (riadok 70 a 80) sa vracia program späť na riadok 30 a začína opäť od začiatku. Program beží do nekonečna (pokiaľ ho nezastavíte stlačením BREAK).

```
 10 REM CVRCKOV PROGRAM
 20 LET cvrk=80
 30 LET teplota=INT ((cvrk/4)+40.5)
 40 PRINT ' "Ked je teplota "; INK 2;teplota; INK 0;" ' Fahrenheita,"
 50 PRINT "cvrcek zacvrlika "; INK 2;cvrk; INK 0;" krat."
 60 LET cvrk=cvrk+INT (RND*7)
 70 FOR j=1 TO 100
 80 NEXT j
 90 POKE 23692,-1
100 GO TO 30
```

Ak chceme ušetriť pamäť, je lepšie zavádzať krátke premenné ako napríklad a1,a2,a3,...a9. Pri dlhších programoch sú však premenné s dlhými názvami oveľa prehľadnejšie, pretože sami napovedajú, čo sa pod nimi skrýva. Samozrejme ale zaberajú viacej pamäti. Aké je teda riešenie? Pri kratších programoch používať dlhé premenné a pri zložitejších dlhých programoch, kde nám záleží na šetrení pamäte používať krátke jednopísmenové premenné s číslom. V tomto prípade je však nutné si prehľadne do zvláštneho zošita písať, ktorá premenná na čo slúži, aby bol program aj po dlhšej dobe zrozumiteľný.

```
10 REM PREMENNE
20 LET a$="zadane cislo: ":LET b$="2.mocnina: ":LET c$="odmocnina: "
30 INPUT "zadaj cislo: ";cislo
40 PRINT a$;cislo
50 PRINT b$;cislo^2
60 PRINT c$;SQR (cislo)
70 GO TO 30
```

Retazcovým premennym sa priraduje obsah tak isto pomocou prikazu LET ako u číselnych premennych, s tým rozdielom, že priradovany obsah musi byť v zátvorkach. Napr. LET a$="retazec".

I N P U T

Prikaz INPUT je nutný všade tam, kde chceme v priebehu programu vložit do počítača nejaku informáciu. Akonáhle je spracovávaný prikaz INPUT, Spectrum zastavi beh programu a čaka na vstup z klavesnice. Až po zadani vstupu program beži dalej.

Skuste si nasledujúci krátky program, ktory Vam ukazuje prikaz INPUT v akcii. Počítač čaká, až zadáte prve číslo (a odošlete ho klávesou ENTER), pýta si druhé číslo a na konci predklada súčet oboch čísel.

```
10 REM INPUT
20 INPUT x
30 INPUT y
40 LET z=x+y
50 PRINT x;" + ";y;" = ";z
```

Tento program Vám iste nerobi žiadne ťažkosti, lebo ste sa s podobnými v knižke už zoznámili. Elegantnejšie je prirodzene, ak obsahuje prikaz INPUT zároveň i správu pre obsluhu počítača. Predošlý program potom bude vyzerať takto:

```
10 REM INPUT
20 INPUT "zadaj prve cislo: ";x
30 INPUT "zadaj druhe cislo: ";y
40 LET z=x+y
50 PRINT x;" + ";y;" = ";z
```

Vidite, že program teraz už aj vypíše, aku hodnotu od Vás očakáva, čo robi program prehľadnejším. Ďalšie využitie tohoto prikazu si budeme demonštrovať na nasledujúcej hre.

H Á D A N I E Č Í S I E L

```
 5 REM HADANIE CISIEL
 6 POKE 23609,100
10 LET body=0
15 FOR j=1 TO 20
20 PRINT AT 1,8; INK j/4;"Pokus cislo: ";j
30 INPUT INK j/4;"Zadaj cislo (1-10): ";a
40 IF a<1 OR a>10 THEN GO TO 30
50 PRINT AT 10,0; INK 3;"Tvoje cislo je ";a;
60 FOR g=1 TO 4
70 LET b=INT (RND*10)+1
80 PRINT AT 5,15;INK 2;b;" "
85 FOR m=1 TO 10: BEEP .01,3.5*m: NEXT m
90 IF b=a THEN GO TO 110
100 NEXT g
110 IF a=b THEN LET body=body+1: PRINT AT 21,6; INK 4;
```

```
    "Vyborne, uhadol si!"
140 IF a<>b THEN PRINT AT 21,4; INK 0; FLASH 1;
    "Mas smolu, neuhadol si!"
150 PRINT AT 12,0; INK 2;"Tvoj bodovy stav: ";bcdy
160 IF body=5 THEN GO TO 250
170 FOR t=1 TO 20
180 BEEP .01,2*t: BEEP .01,20-t: NEXT t
190 CLS
200 NEXT j
210 PRINT AT 1,10; BRIGHT 1; INK 2;"Koniec hry"
220 PRINT INK 2; AT 3,2;"Pocet dosiahnutych bodov: ";body
230 PRINT INK 1;"Tvoje zasahy mali uspesnost ";body*5;" %"
240 STOP
250 CLS
260 FOR m=1 TO 4: BEEP .01,RND*50: PAUSE 3: NEXT m
270 PRINT ' INK RND*6;"      ";FLASH 1;"Mas stastie, vyhral si!"
280 POKE 23692,-1
290 GO TO 260
```

V tomto programe si zvolite číslo medzi 1 a 10, potom zvolí počítač pomocou generátora náhodných čísiel štyri čísla tiež medzi 1 a 10. Ak je aspoň jedno s týchto čísliel rovnaké s tým, ktoré ste zvolili, dostanete bod. Ak obdržíte pri 20-tich priebežných hrách dohromady 5 bodov, vyhrávate. Ak nie, stratili ste výhru a vypíše sa vám počet zásahov a uspešnosť vypočítaná v %. Ak ste si program vyskúšali, prezrite si vysvetlivky k jednotlivým riadkom.

```
 6 mení pípnutie pri stlačení klávesy
10 vynuluje počet bodov
15 začiatok cyklu FOR-NEXT pre priebeh hry
20 vypíše číslo pokusu
30 očakáva Vaše číslo
50 vypíše Vami zvolené číslo
60 tu začína slučka FOR-NEXT, kde sa generujú štyri náhodné čísla
110 zvýši bodový stav a vypíše hlásenie, ak ste uhádli číslo
140 vypíše hlásenie ak ste neuhádli číslo
150 zobrazi bodový stav
170 krátka prestávka s pípaním pred ďalším priechodom hry
190 zmazanie obrazovky
200 koniec slučky FOR-NEXT pre priebeh hry
210 koniec hry pri prehre
250 koniec hry pri výhre
```

V Ý P O Č E T Ú R O K O V

Tento program ukazuje opäť INPUT príkaz v akcii. V programe sa používajú dlhšie mená premenných, ľahko zrozumiteľne. Vyskúšajte si ho:

```
10 REM VYPOCET UROKOV
30 INPUT INK 1;"Vlozena suma: ";vklad
40 INPUT INK 2;"Urokova sadzba: ";urok
50 INPUT INK 4;"Na kolko rokov: ";roky
60 PRINT INK 2;"ROK";TAB 7;"SUMA";TAB 17;"PRIRASTOK"
70 LET prir=0
```

```
 80 LET suma=vklad
 90 POKE 23962,-1
100 PRINT INK 1;"--------------------------------"
110 FOR m=1 TO roky
120 LET suma=suma+((suma/100)*urok)
130 LET prir=suma-vklad
140 PRINT m;TAB 7;suma;TAB 17;prir
150 NEXT m
```

Tento program počita úroky a ročný prírastok z Vašeho počiatočného vkladu za počet rokov, ktorý zadáte.

Ešte niekoľko rád k príkazu INPUT:

Ak si prajete program prerušiť, pokiaľ príkaz INPUT čaká na Váš vstup, budte si istý, že BREAK nebude fungovať. Pokiaľ čaká počítač na číselný vstup bez úvodzoviek, môžete namiesto čísla zadať príkaz STOP (SYMBOL SHIFT/A). Pri reťazci, ktorý má rozsah určený úvodzovkami, musíte kurzor postaviť doľava pred úvodzovky a až potom zadať príkaz STOP).

Ak chcete využívať Váš program častejšie, alebo ho použiť pre ďalšie aplikácie, potom zaveďte na koniec programu nasledujúce riadky:

```
9998 INPUT "Chcete program znova ? (ano/nie): ";r$
9999 IF r$(1)="a" OR r$(1)="A" THEN RUN
```

Na takúto otázku môžete odpovedať celým slovom (ano alebo nie), alebo len začiatočným písmenom odpovede (a alebo n) pričom môžete použiť malé i veľké písmená. Ak však namiesto odpovede stlačíte len klávesu ENTER, program sa síce zastaví, ale vypíše chybu: "3 Subscript wrong"

```
9997 DIM r$(1)
9998 INPUT "Chcete program znova ? (ano/nie): ";r$
9999 IF r$(1)="a" OR r$(1)="A" THEN RUN
```

Tu je jednoduchá metóda, s ktorou sa môžete tohoto chybového hlásenia vyvarovať. Pretože je reťazec r$ dimenzovaný príkazom DIM na dĺžku 1, vytvorí sa prázdny reťazec r$=" ". Zároveň, keďže je reťazec r$ dimenzovaný na dĺžku 1, môžeme zadať napríklad takýto reťazec:

"Ano, mily pocitac, bud tak dobry a spusti ten program este raz"

Program si do reťazca r$ vyberie len prvý znak ("A") a ostatné bude ignorované.

```
9997 DIM r$(1)
9998 INPUT "Chcete program znova ? (ano/nie): ";r$
9999 IF CODE r$=CODE "a" OR CODE r$=CODE "A" THEN RUN
```

Tiež táto metóda je funkčná. Počítač porovná ASCII kód znaku, ktorý ste zadali s tým, ktorý má v úvodzovkách a keď sú zhodné vykoná RUN.

Čítanie prvého písmena v INPUT príkaze je teda ako vidíte, nepatrný problém. Len o málo ťažšie bude, ak budeme chcieť obmedziť počet znakov, ktoré má počítač akceptovať. Nasledujúci program zoberie len ľubovoľné

písmeno malej abecedy, ostatné znaky bude ignorovať.

```
10 INPUT a$
15 IF a$="" THEN BEEP .1,30: GO TO 10
20 FOR a=1 TO LEN a$
30 IF a$(a)<"a" OR a$(a)>"z" THEN BEEP .1,30: GO TO 10
40 NEXT a
50 PRINT a
```

Riadok 15 zabezpečuje, že prázdny vstup (len ENTER) sa netoleruje a vráti program na riadok 10. Slučka začínajúca v riadku 20 kontroluje postupne všetky znaky vstupného reťazca, a ak je niektorý z nich nevyhovujúci, vráti program na riadok 10.

Ešte flexibilnejšie budú Vaše odpovede s nasledujúcim podprogramom:

```
7010 INPUT "Chcete program znova ? (ano/nie): ";a$
7020 FOR a=1 TO LEN a$-2
7030 IF a$ (a TO a+2)="ano" THEN RUN
7040 NEXT a
7050 STOP
```

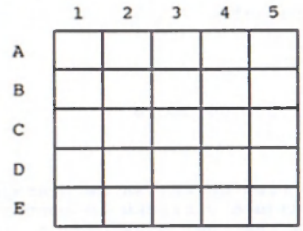
Teraz môžete písať INPUT vstup a to tak dlho, ako to dĺžka reťazca vôbec dovolí. Ak sa bude niekde v reťazci nachádzať slovo "ano", program sa spustí znova.

Ak však chcete, aby program reagoval len na jedno slovo ("ano") osamote (ako záverečné kľúčové slovo), potom treba program zmeniť takto:

```
7010 INPUT "Zadaj hladane slovo: ";s$
7020 INPUT "zadaj prehladavanu vetu: ";a$
7040 LET a$=" "+a$+" "
7050 LET ls=LEN s$
7060 LET la=LEN a$
7070 FOR a=2 TO la-ls
7080 IF a$(a to a+ls-1)=s$ AND (a$(a-1)<"a" OR a$(a-1)>"z")
     AND (a$(a+ls)<"a" OR a$(a+ls)>"z") THEN RUN
7090 NEXT a
```

Pozrieme sa na iný príklad s príkazom INPUT, ktorý sa často používa v spoločenských hrách. Máte napríklad hrúciu plochu (šachovnicu) a chcete zadať políčko, na ktoré sa má potiahnuť (napr. C3).
V zápale hry sa často stáva, že zadáte polohu naopak (ako prvé číslo a až potom písmeno). Program v takomto prípade údaj nezoberie a zahlási chybu. Náš nasledujúci podprogram je schopný zobrať aj takýto obrátený údaj, upraviť si ho do správneho tvaru a vyhodnotiť.

```
10 INPUT a$
20 IF LEN a$<2 THEN BEEP .1,30: GO TO 10
25 LET a$=a$ (TO 2)
30 IF a$(1)>="1" AND a$(1)<="5" AND a$(2)>="a" AND a$(2)<="e" THEN
   LET a$=a$(2)+a$(1)
40 IF a$(1)<"a" OR a$(1)>"e" OR a$(2)<"1" OR a$(2)>"5" THEN
   BEEP .1,30: GO TO 10
50 PRINT a$
```



Tento podprogram pre vstup dvoch znakov pracuje veľmi rychlo a pri zadávani pozicie sa vlastne nemôžeme zmýliť. Pre vstup štvorice znakov (presun z jedného políčka na druhé) slúži nasledujuci program:

```
10 INPUT a$
20 IF LEN a$<4 THEN BEEP .1,30: GO TO 10
30 LET a$=a$(TO 4)
40 IF a$(1)>="1" AND a$(1)<="5" AND a$(2)>="a" AND a$(2)<="e" THEN
   LET a$(TO 2)=a$(2)+a$(1)
50 IF a$(3)>="1" AND a$(3)<="5" AND a$(4)>="a" AND a$(4)<="e" THEN
   LET a$(3 TO)=a$(4)+a$(3)
60 IF a$(1)<"a" OR a$(1)>"e" OR a$(2)<"1" OR a$(2)>"5" OR a$(3)<"a"
   OR a$(3)>"e" OR a$(4)<"1" OR a$(4)>"5" THEN BEEP .1,30: GO TO 10
70 PRINT a$
```

V oboch posledných programoch môžeme ušetriť trochu miesta, ak vstupy dimenzujeme. Napíšte pre prvý program

```
DIM a$(2)
```

a vymažte riadky 20 a 25. Pre druhý program zadajte

```
DIM a$(4)
```

a môžete vymazať riadky 20 a 30. V oboch verziách je tým zaistené, že reťazec nebude dlhší ako požadujeme.

Predošlý program nám teda zaistí, aby nebolo možné zadať nesprávne alebo neexistujúce políčko, ale neošetruje správnosť ťahov. Nasledujúci podprogram zistí nepripustné ťahy v dáme a nedovolí ich vykonať.

```
65 IF (CODE a$(1)<>CODE a$(3)+1) OR (CODE a$(2)<>CODE a$(4)+1) OR
   (CODE a$(2)=CODE a$(4)-1) THEN BEEP .1,30: GO TO 10
```

A ešte niekoľko rád, ktorými by ste sa mali riadiť pri zostavovaní podmienok a podprogramov všeobecne:

1. Premyslite si presne, čo chcete docieliť.
2. Definujte si pevne, čo je dovolené a čo zakázané.
3. Tomu, čo je zakázané môžete zabrániť, alebo keď to predsa bolo zadané, ošetriť iným spôsobom tak, aby to v programe nevadilo.

4. Pamatajte si aplikaciu rôznych príkladov a vhodne ich používajte.
5. Ak sa domnievate, že podprogram už musí fungovať, spustite ho a preberajte všetky aj nezmyselne možnosti (napr. ťahy na neexistujúce políčka). Ak ste s výsledkami spokojný, potom nasleduje posledný test.
6. Nechajte Vášho priateľa, či člena rodiny, aby sa bližšie zoznámil s Vaším podprogramom a vyskúšal ho. Ak podprogram obsahuje nejakú skrytú chybu, najrýchlejšie ju väčšinou objaví úplne nezaujatá osoba.

Nakoniec sa pozrieme na číselný INPUT. Zadajte NEW a napíšte:

```
10 INPUT a
20 PRINT a
30 GO TO 10
```

Keď takýto program spustíte, skúste zadať hodnotu 1. Program ju samozrejme akceptuje. Na tom nie je nič zvláštne. Skúste ale teraz zadať len písmeno a. Program premennú zoberie, pretože už bola predtým definovaná príkazom INPUT a bola jej priradená hodnota 1.
Skúste ale teraz zadať písmeno b. Program sa preruší a zobrazí sa chybové hlásenie "2 Variable not found" (premenná nebola nájdená).

Veľmi dôležitou je v príkaze INPUT funkcia VAL. Jej výhodou je, že možno zadávať písmená i čísla. Spustite tento program

```
10 INPUT a$
20 LET a=VAL a$
30 PRINT a
40 GO TO 10
```

a skúste zadať napríklad INT (2^14-3*PI).

Prekliatím funkcie VAL sú však tiež nedefinované premenné. Ich zadávaniu môžeme zabrániť napríklad tak, že pripustíme iba zadávanie čísiel:

```
10 INPUT a$
20 FOR f=1 TO LEN a$
30 IF a$(f)<"0" OR a$(f)>"9" THEN GO TO 10
40 NEXT f
50 LET a=VAL a$
60 PRINT a
```

Tento podprogram možno ľahko rozvinúť tak, aby boli akceptované napríklad znamienka "+" a "-". Nahraďte riadok 30 nasledujúcim:

```
30 IF (a$(f)<"0" OR a$(f)>"9") AND (a$(f)<>"+" AND a$(f)<>"-")
   THEN GO TO 10
```

G O T O

Pre dômyselné prevedenie Vášho programu je enormne dôležité, aby počítač nespracovával program strnulo od prvého riadku k poslednému, ale

podľa Vašeho priania (aby časť programu preskočil, vracal sa inam atď.).

Prikaz, ktorý nam umožní skoky vo vnútri programu sa vola GO TO. Za tymto prikazom nasleduje číslo riadku, na ktorý chceme presmerovať program. Nemusí to byt len číslo, ale môže to byt aj premenná alebo matematický vyraz. (Napr. GO TO 100, GO TO a, GO TO a+100...)

Prikaz GO TO môžeme použit aj na vytvorenie nekonečnej slučky, čo si ukážeme v nasledujucom kratkom programe:

```
10 PRINT INK RND*6;"nekonecny scroll";
20 POKE 23692,-1
30 BEEP .01,RND*50
40 GO TO 10
```

I F - T H E N

Pomocou prikazovej dvojice IF-THEN môžeme v programe vytvárať rôzne podmienky, čo je veľmi dôležité pre členenie programu a ošetrovanie rozličných situácii.

Velmi často sa tento prikaz kombinuje s prikazom skoku GO TO, ako napr. IF a=b THEN GO TO 100 (keď sa a rovná b potom skoč na riadok 100).

Tu je zoznam matematických znamienok a funkcii, ktoré sa dajú použiť na vytvorenie podmienky s prikazom IF-THEN:

=	rovná sa	AND	logický súčin
>	väčší než	OR	logický súčet
<	menší než	NOT	negácia
<>	nerovná sa		
>=	väčší alebo rovnaký		
<=	menší alebo rovnaký		

A tu si môžeme prehliadnuť riadok s takouto podmienkou:

```
70 IF z>=10 THEN GO TO 100
80 GO TO 200
```

Počítač bude v tomto prípade porovnávať, či premenná z je rovnaká alebo väčšia ako 10. Ak je podmienka splnená, potom skočí na riadok 100, ak nie, program pokračuje na riadku 80.

Pod týmto sa práve rozumie umelá "inteligencia" počítača, totiž že počítač sa rozhoduje na základe určitých predpokladov, ktoré interpretuje.

Nasledujúci príklad ukazuje použitie viacerých príkazov IF-THEN.

```
10 INPUT INK RND*6;"Zadaj číslo od jednej do troch: ";a
30 IF a=1 THEN PRINT "jedna"
40 IF a=2 THEN PRINT "dva"
50 IF a=3 THEN PRINT "tri"
60 GO TO 10
```

Ako sa spája viacej podmienok do jednej pomocou funkcie AND si ukážeme na nasledujúcom kratkom programe:

```
10 INPUT a
```

```
20 INPUT b
30 IF a=1 AND b=1 THEN PRINT "podmienka je splnena"
40 GO TO 10
```

Preskúšajte tento program s rôznymi vstupnými hodnotami. Zbadate, že podmienka je skutočne splnená len vtedy, ak su splnene obe jej časti.

Teraz zmeňte riadok 30 tak, že dosadíte funkciu OR a znova vyskúšajte zadávať rôzne hodnoty.

```
10 INPUT a
20 INPUT b
30 IF a=1 OR b=1 THEN PRINT "podmienka je splnená"
40 GO TO 10
```

Po chvíli skúšania zbadate, že podmienka je v tomto prípade splnená, ak je splnená ľubovoľná z jej časti.

Prehliadnite si prosím ešte tento program:

```
10 INPUT a
20 INPUT b
30 IF a=NOT b THEN PRINT "podmienka je splnena"
40 GO TO 10
```

Skúste zadávať rôzne hodnoty a po chvíli zistite, že podmienka je splnená len vtedy, ak je jedna z hodnôt 1 a druhá 0.
Skúste ju nahradiť nasledujúcou podmienkou a uvidíte, že dosiahnutý výsledok nie je rovnaký.

```
10 INPUT a
20 INPUT b
30 IF a<>b THEN PRINT "podmienka je splnena"
40 GO TO 10
```

Aké rozličné podmienky použijete, je samozrejme len na Vašej fantázii. Jedna z drastickejších podmienok je v dalšom príklade:

```
 1 INPUT "Zadaj heslo: ";a$: IF a$<>"heslo" THEN NEW
```

Ak si predošlý riadok dáte na začiatok svojeho programu, každý okrem Vás, ktorý poznáte "heslo" s ním zažije krušné chvíle. Ak dotyčný nezadá správne heslo, program sa jednoducho vymaže z pamäti počítača. Ale pozor, aby ste sa nepomýlili aj Vy! Počítač totiž v tomto prípade rozlišuje veľké a malé písmená.

Pozrime sa teraz na iný príklad. V nasledujúcom programe pevne definujeme, že 1 je "pravda" a 0 "nepravda".

```
10 INPUT a
20 INPUT b
30 LET x=(a=b)
40 IF x=1 THEN PRINT "pravda"
50 IF x=0 THEN PRINT "nepravda"
60 GO TO 10
```

Zátvorky v riadku 30 nie sú nevyhnutne nutne, ale chceme ukázať prehľadnosť riešenia. Vyraz v zátvorkach sa bude vždy rovnať buď 0 alebo 1. Teda vyraz sa bude rovnať jednej len vtedy, ak sa a bude rovnať b.

Ak to potrebujeme pre našu programovú aplikáciu, nie je problém aby "pravda" bolo iné číslo ako 1. Pre "nepravdu" ale vždy ostane hodnota 0. V nasledujucej ukažke bude "pravda" nadobudať hodnotu 2.

```
10 INPUT a
20 INPUT b
30 LET x=(a=b)*2
40 IF x=2 THEN PRINT "pravda"
50 IF x=0 THEN PRINT "nepravda"
60 GO TO 10
```

Na programe je praktické, že výsledok výrazu môžete použiť v ďalších podmienkach. Ako tento príklad aplikujeme do praxe, ukazuje další program.

```
10 REM PROGRAM NA POSTREH
20 LET s=0
30 FOR j=1 TO 9
40 PRINT AT 10,2*j; INK j/2;j
50 NEXT j
60 FOR g=30 TO 1 STEP -1
70 PRINT AT 5,3; PAPER 2; INK 6;"CAS: ";g;" ";AT 5,13; PAPER 6;
   INK 2;"BODY: ";s;" "
80 LET a=INT (RND*9)+1
90 PRINT AT 11,2*a; FLASH 1; INK 2;"˙"
100 FOR h=1 TO 24
110 BEEP .01,50/h
120 LET a$=INKEY$
130 IF a$<>"" THEN GO TO 170
140 NEXT h
150 LET g=g-1
170 LET s=s+(a$=STR$ a)*a
180 PRINT AT 11,2*a;" "
190 NEXT g
200 PRINT AT 5,3; PAPER 2; INK 6;"CAS: ";g;" ";AT 5,13; PAPER 6;
   INK 2;"BODY: ";s;" "
```

Program najskôr nakresli čísla 1 až 9. Potom náhodne vypočíta jedno z týchto čísiel a zobrazí pod ním šípku. Vtedy musíte pohotovo zareagovať a rýchlo stlačiť príslušné číslo na klávesnici. Podlá rýchlosti Vašej reakcie Vám počítač prideli body.

Riadok 170 nás zaujíma v tomto programe obzvlášť. Tu bude práve porovnaná náhodne vybraná klavesa s tou, ktorú ste stlačili. Ak budú zhodné, výsledok výrazu bude 1. Ďalej bude tento výsledok násobený hodnotou a. Ak bude výsledkom 0, nemeni sa Vaše bodové číslo, ak 1, budú body rovné 1*a. Sumačné bodové číslo je vyčíslené premennou s, Váš čas je ohraničený premennou g.

Pozrite sa teraz na podmienky s funkciami AND, OR, alebo NOT. Tieto podmienky sa často používajú na presuvanie objektu doľava/doprava a hore/dole po obrazovke v rôznych hrách. Používajú sa taktiež v systémových programoch na ošetrenie a presun pozície kurzora. Skúste tento podprogram:

```
10 LET x=15
20 IF INKEY$="5" AND x>1  THEN PRINT AT 21,x;" ": LET x=x-1
30 IF INKEY$="8" AND x<30 THEN PRINT AT 21,x;" ": LET x=x+1
40 PRINT AT 21,x; INK 2;"=": GO TO 20
```

Po spustení tohoto programu môžete pomocou tlačítiek 5(<) a 8(>) pohybovať červeným kurzorom v spodnom riadku. Rovnaký výsledok dosiahneme pomocou tejto upravy:

```
10 LET x=15
20 LET x=x-(INKEY$="5" AND x>1)+(INKEY$="8" AND x<30)
30 IF INKEY$<>"" THEN PRINT AT 21,x-1;"   "
40 PRINT AT 21,x; INK 2;"="
50 GO TO 20
```

Pri tomto riešení sme dve podmienky zlúčili do jedného riadku, čím sme jednak ušetrili pamäť, ale hlavne sme zrýchlili priebeh programu.

Ukážme si ešte raz použitie funkcie OR. Tentokrát si ju predvedieme na príklade vodiča diaľkového autobusu. Tento vodič má pevne stanovené, že osobám, ktoré majú 14 a menej rokov má predávať lístky za polovičné cestovné. Nasledujúci krátky program mu to spoľahlivo zabezpečí.

```
10 INPUT "Zadaj plne cestovne: ";cena
20 INPUT "Zadaj vek cestujuceho: ";vek
30 LET cena=cena*(.5 OR vek>14)
40 PRINT "Cestovne je ";cena;" Kcs"
```

Takto môžete do jedného riadku zaviesť aj omnoho viac cenových kategórií.

Pre rozličné podmienky v programoch sa dá výborne používať funkcia NOT. I to sa dá najlepšie ukázať na príklade:

```
10 INPUT a
20 INPUT b
30 PRINT a;TAB 4;b;TAB 8;NOT a=b
40 GO TO 10
```

Čo uvidíte na obrazovke po spustení programu, sú obe čísla, ktoré sú zadané na riadkoch 10 a 20, s doprovodom čísla 0 alebo 1, podľa toho, či sú zadané čísla rovnaké alebo nie. Preskúmajte aký výsledok prinesú rôzne hodnoty premenných a, b. Preskúmajte tiež, aký efekt prinesie výmena znaku rovnosti "=" za iný znak.

Na záver ešte niečo zvláštne. Pozrite sa na nasledujúci riadok:

```
10 IF a=1 THEN IF b=2 THEN PRINT "pravda"
```

Je zaujímavé, že znamená to isté ako:

```
10 IF a=1 AND b=2 THEN PRINT "pravda"
```

Zapamätajte si to, môže sa stať, že to niekedy budete potrebovať.

Druhá zvláštnosť nie je vlastne nič zvláštneho, ale nie je to veľmi známa aplikácia, preto ju radšej prinášame. Vyskúšajte si tento program

```
10 INPUT a
20 IF a THEN PRINT a
```

a potom zmeňte program takto:

```
10 INPUT a
20 IF NOT a THEN PRINT a
```

Zdanlivo očakávate, že program nepobeží, pretože a sa s ničím nezrovnáva. Ale mýlite sa. Pri prvej aplikácii sa zisťuje, či je premenná a naplnená nejakou ľubovoľnou hodnotou a pri druhej podmienka reaguje len na nulovú hodnotu premennej a.

Pomocou IF-THEN sa dá veľmi ľahko vyrobiť počítadlo a kontrolovať (napríklad v hre), kedy dosiahne určitý počet bodov. Potom môže byť vypísaná správa, ukončený program a podobne. Skúste si:

```
10 LET x=0
20 LET x=x+1
30 IF x>25 THEN PRINT "Vyhral si!"
40 GO TO 20
```

Po THEN nemusí bezpodmienečne nasledovať výkonný príkaz (napr. PRINT, LET) ale môže po ňom nasledovať skokový príkaz GO TO, takže náš príklad bude potom vyzerať povedzme takto:

```
10 LET x=0
20 LET x=x+1
30 IF x>25 THEN GO TO 60
40 GO TO 20
50 PRINT "Prehral si!"
60 PRINT "Vyhral si!"
```

I F - T H E N (E L S E)

Veľa dialektov jazyka BASIC na iných počítačoch môže používať s IF-THEN podmienkou ešte dodatok ELSE. V príklade to vyzerá takto:

```
10 IF a=3 THEN PRINT "tri"
20 ELSE PRINT "nie su to tri"
```

V praxi to znamená, že ak premenná a bude mať hodnotu 3, vypíše sa číslo slovom: "tri". V opačnom prípade (teda ak a<>3) sa zobrazí správa "nie su to tri".

V BASICu Spectra žiaľ tento praktický príkaz chýba, ale napriek tomu môžeme logiku IF-THEN-ELSE nepriamo používať.

Praktický príklad použitia simulovaného príkazu IF-THEN-ELSE si môžeme predviesť na nasledujúcom programe GRAFICKÝ ZAPISOVAČ.

V riadku 60 je v tomto programe zadaná funkcia, ktorú náš "grafický zapisovač" bude zapisovať takto: Zakaždým keď funkcia k dosiahne hodnotu

väčšiu alebo rovnakú ako .5 nakresli znak "o" a ked je vysledok funkcie hodnota menšia ako .5 nakresli znak ".".

```
 10 REM GRAFICKY ZAPISOVAC
 20 FOR y=10 TO -10 STEP-1
 30 IF y<>10 AND y<>-10 AND y>=1 THEN PRINT " ";
 40 PRINT y;TAB 4;
 50 FOR x=-10 TO 10
 60 LET k=y-x*x/2+7
 70 PRINT ("o" AND k>=.5)+("." AND k<=.5);
 80 NEXT x
 90 PRINT
100 NEXT y
110 PRINT TAB 4;".9.7.5.3.1.1.3.5.7.9."
```

Výsledok po spusteni programu bude vyzerat takto:

```
 10  .....oooooooooo.....
  9  .....oooooooooo.....
  8  .....oooooooooo.....
  7  .....oooooooooo.....
  6  .....oooooooooo.....
  5  ......oooooooo......
  4  ......oooooooo......
  3  ......oooooooo......
  2  ......oooooooo......
  1  .......oooooo.......
  0  .......oooooo.......
 -1  .......oooooo.......
 -2  .......oooooo.......
 -3  ........ooooo........
 -4  ........ooooo........
 -5  .........ooo.........
 -6  .........ooo.........
 -7  ....................
 -8  ....................
 -9  ....................
-10  ....................
     .9.7.5.3.1.1.3.5.7.9.
```

Teraz zmeňte riadok 60 tak, aby program vyzeral takto:

```
 10 REM GRAFICKY ZAPISOVAC
 20 FOR y=10 TO-10 STEP -1
 30 IF y<>10 AND y<>-10 AND y=1 THEN PRINT " ";
 40 PRINT y;TAB 4
 50 FOR x=-10 TO 10
 60 LET k-3*ABS (y)-x*x
 70 PRINT ("o" AND k>=.5)+("." AND k<=.5);
 80 NEXT x
 90 PRINT
100 NEXT y
110 PRINT TAB 4;".9.7.5.3.1.1.3.5.7.9."
```

A opat si prezrite vysledok:

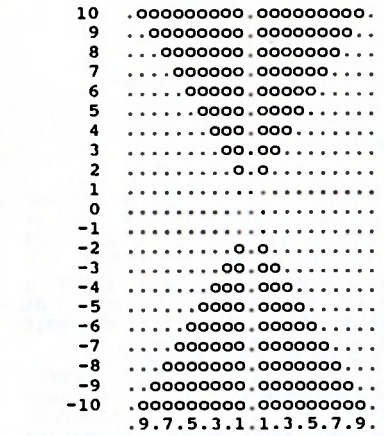
```
 10  .....oooooooooo.....
  9  .....oooooooooo.....
  8  ......oooooooo......
  7  ......oooooooo......
  6  ......oooooooo......
  5  .......ooooooo.......
  4  .......ooooooo.......
  3  ........ooooo........
  2  ........ooooo........
  1  .........ooo.........
  0  ...................
 -1  .........ooo.........
 -2  ........ooooo........
 -3  ........ooooo........
 -4  .......ooooooo.......
 -5  .......ooooooo.......
 -6  ......oooooooo......
 -7  ......oooooooo......
 -8  ......oooooooo......
 -9  .....oooooooooo.....
-10  .....oooooooooo.....
     .9.7.5.3.1.1.3.5.7.9.
```

Teraz si ukážeme ešte tri zaujimavé funkcie. Zmeňte v programe vždy len riadok 60 a program znova spustite.

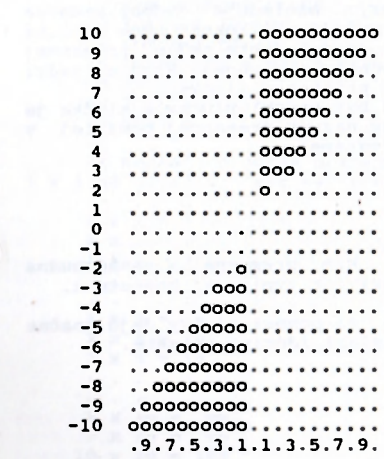
```
60 LET k=SQR (ABS (y*x*2))-x
```

```
 10  oooooooooo.oooooooooo
  9  oooooooooo.oooooooooo
  8  oooooooooo.oooooooooo
  7  oooooooooo.oooooooooo
  6  oooooooooo.oooooooooo
  5  oooooooooo.oooooooo..
  4  oooooooooo.oooooo....
  3  oooooooooo.oooo......
  2  oooooooooo.oo........
  1  oooooooooo...........
  0  oooooooooo...........
 -1  oooooooooo...........
 -2  oooooooooo.oo........
 -3  oooooooooo.oooo......
 -4  oooooooooo.oooooo....
 -5  oooooooooo.oooooooo..
 -6  oooooooooo.oooooooooo
 -7  oooooooooo.oooooooooo
 -8  oooooooooo.oooooooooo
 -9  oooooooooo.oooooooooo
-10  oooooooooo.oooooooooo
     .9.7.5.3.1.1.3.5.7.9.
```

```
60 LET k=ABS (y*x)-x*x
```




```
60 LET k=y*x-x*x/1.1
```



Ak sa Vám program zapáčil a chcete si ho uschovať na magnetofónovú

pasku, odporúčame doplniť riadok 10 a 120 a zmeniť riadok 60. Program sa tak stane ovela komfortnejší a bude vyzerať takto:

```
 5 REM GRAFICKY ZAPISOVAC
10 INPUT "funkcia: ";k$
20 FOR y=10 TO-10 STEP -1
30 IF y<>10 AND y<>-10 AND y=1 THEN PRINT " ";
40 PRINT y;TAB 4
50 FOR x=-10 TO 10
60 LET k=VAL k$
70 PRINT ("o" AND k>=.5)+("." AND k<=.5);
80 NEXT x
90 PRINT
100 NEXT y
110 PRINT TAB 4;".9.7.5.3.1.1.3.5.7.9."
120 PAUSE 0: RUN
```

Teraz už nemáte zadanú funkciu napevno, ale pomocou príkazu INPUT a prevodu pomocou funkcie VAL môžete zadávať a skúšať ľubovoľné hodnoty. Ak budete chcieť program ešte vylepšiť, môžete znaky "o" a "." nahradiť vhodnými grafickými znakmi z UDG (užívateľom definovanej grafiky).

F O R - T O - N E X T S L U Č K Y

S FOR-TO-NEXT slučkami sa stretávame, ak pracujeme na našom počítači v BASICu. Každá takáto slučka sa skladá z troch príkazov, ktoré možno použiť iba súčasne.

Slučka začína príkazom FOR, za ktorým nasleduje jednopísmenová premenná, ktorá slúži ako počítadlo, teda určuje, koľkokrát bude slučka prebiehať. Potom nasleduje priradenie počiatočnej hodnoty tejto premennej a za príkazom TO priradenie konečnej hodnoty premennej. Rozdiel medzi konečnou a počiatočnou hodnotou určuje počet priebehov slučky.

Potom nasleduje časť programu, ktorá má byť vnútri slučky a slučka je zakončená príkazom NEXT, ktorý je nasledovaný názvom premennej použitej v cykle. Najlepšie si to ukážeme na krátkom programe:

```
100 FOR x=1 TO 10
150 PRINT "x = ";x
200 NEXT x
```

Táto slučka bude prebiehať dovtedy, kým premenná x nenadobudne hodnotu 10 a zároveň bude pre lepšiu predstavivosť vypisovať hodnotu x.

Nasledujúci program ukazuje, ako sa za pomoci slučky dajú značne urýchliť matematické operácie s viacerými číslami idúcimi po sebe.

```
10 LET a=2
20 LET b=10
30 FOR c=a TO b
40 PRINT TAB 6;c;TAB 12;c*a;TAB 18;c/a
50 NEXT c
```

Po spustení programu sa zobrazí táto postupnosť čísiel:

```
 2     4     1
 3     6     1.5
 4     8     2
 5    10     2.5
 6    12     3
 7    14     3.5
 8    16     4
 9    18     4.5
10    20     5
```

Ako teda vidíte, v slučkách nemusia byť len fixné hodnoty, ale možno do nich dosadiť aj premenne. V mnohých programoch sa dajú využiť duchaplne slučky, ktoré obsahujú počiatočnú i konečnú hodnotu tvorenú premennou, ktorá je výsledkom matematickej operácie v inej časti programu a ktorá sa vytvorila až v jeho priebehu.

V N O R E N É S L U Č K Y

Slučky FOR-TO-NEXT môžeme používať aj viackrát za sebou s tou istou premennou. Ale nastanú situácie, keď bude potrebné, aby sa jeden proces opakoval y-krát v rámci procesu, ktorý sa bude opakovať x-krát. Nie je nič jednoduchšie. Slučky môžeme totiž ľubovoľne vnárať do seba, pričom samozrejme musíme zachovávať prísne ich hierarchiu.

Skúste si nasledujúci program, kde je slučka s premennou b úplne vnorená do slučky s premennou a.

```
10 REM VNORENE SLUCKY
20 FOR a=1 TO 10
30 FOR b=1 TO 10
40 PRINT TAB 8;b;" * ";a;" = ";b*a
50 NEXT b
60 POKE 23962,-1
70 NEXT a
```

Ak necháte program prebehnúť, dostanete kompletnú tabuľku pre násobky 1 x 1 až 10 x 10. Časť výsledku uvidíte takto:

```
1 x 1 = 1
2 x 1 = 2
3 x 1 = 3
. . . . .
. . . . .
1 x 5 =  5
2 x 5 = 10
3 x 5 = 15
. . . . . .
. . . . . .
8 x 10 =  80
9 x 10 =  90
10 x 10 = 100
```

Teraz skúste spraviť zámernú chybu. Porušte hierarchiu vnorených

slučiek, vymeňte medzi sebou riadky 50 a 70 a spustite program znova.

Tu je časť výsledku:

```
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
. . . . .
. . . . .
1 x 10 = 10
2 x 11 = 22
3 x 12 = 36
. . . . . .
. . . . . .
8 x 17 = 136
9 x 18 = 162
10 x 19 = 190
```

Ako vidíte na príklade, je veľmi nebezpečné prehodiť poradie slučiek ponorených do seba. Taktiež neoporúčame, pokiaľ si to nevyžaduje zvláštna aplikácia, používať premenné slučiek na iných miestach programu. Takéto zdvojené použitie premenných vedie hlavne v dlhých programoch k chybám, ktoré sa len ťažko hľadajú.

S T E P

Na začiatku tejto kapitoly si ešte raz prezrite program raketa, ktorý sme si ukázali na začiatku knihy.

Najdôležitejší je pre nás riadok 30 a ďalej riadky 40 a 70. Pomocou slova STEP bude určené, že v riadku 30 hodnota v premennej j klesá postupne z 10 na 1. Zameňte teraz hodnotu -1 v príkaze STEP za -2 a pozorujte, ako sa priebeh slučky zrýchli.

AK nie je STEP (krok) v slučke nijako špecifikovaný, počítač zvolí automaticky veľkosť kroku 1.

Príkaz STEP slúži teda k tomu, že dáva programátorovi možnosť určovať veľkosť kroku v slučke, prípadne zvoliť záporný krok.

Hodnota za príkazom STEP nemusí byť celé číslo, ale je dôležité, aby hodnota za príkazom TO v tomto prípade bola kladné číslo. Skúste si:

```
10 FOR a=100 TO 1 STEP -12.5
20 PRINT TAB 8;a
30 NEXT a
```

```
100
87.5
75
62.5
50
37.5
25
12.5
```

```
10 FOR a=10 TO 1 STEP -.175
20 PRINT TAB 8;a
```

```
30 NEXT a

 10
 9.825
 9.65
 9.475
 9.3
 9.125
 8.95
 8.775
 8.6
 8.425
 8.25
 8.075
 .
 .
```

Číslo za príkazom STEP môže byť zlomok, desatinné číslo, alebo tiež výsledok nejakého výpočtu (vo forme premennej alebo čísla), nemusíme ho voliť, ak je poslednou konečnou hodnotou v poslednej slučke. Slučka beží tak dlho, pokiaľ nie je premenná hodnotou menšia alebo rovnaká ako limit, ktorý je umiestnený za TO. Je však sotva možné meniť hodnotu kroku v priebehu slučky. Skúste si tento príklad:

```
10 FOR f=1 TO 0
20 PRINT "X"
30 NEXT f
```

Nasledujúci príklad možno niekedy vhodne využijete. Zabráni vykonaniu slučky, keď x=6. Skúste si program napísať a po jeho spustení zadajte najskôr číslo 5 a potom 6.

```
10 INPUT "Zadaj x: ";x
20 FOR f=(x=6)*33 TO 31
30 PRINT CHR$ 143;
40 NEXT f
```

Kontrola, či nie je limitná hodnota prekročená je na riadku 20.

Vyjsť zo slučky nie je žiadny problém, ale nemôžete skočiť dovnútra, pokiaľ je definovaná kontrolná premenná. Niektoré verzie BASICu dovolia možnosť, že za NEXT nenasleduje žiadna premenná. V takomto prípade je automaticky použitá premenná najbližšej slučky, ktorá je vyššie v programe. BASIC ZX Spectra však takúto možnosť nepripúšťa. Tu musí byť za NEXT vždy uvedená premenná.

G O S U B A R E T U R N

Podprogram je programový blok, uložený zväčša na konci hlavného programu v podobe špeciálne spracovávanej úlohy. Podprogram je volaný príkazom GO SUB, za ktorým nasleduje číslo riadku, na ktorom začína podprogram. Pri vykonávaní inštrukcie GO SUB počítač skočí na začiatok podprogramu a vykonáva ho až kým nenarazí na príkaz RETURN. V takomto prípade skočí späť do hlavného programu na prvý riadok pod riadkom s

prikazom GO SUB. Podprogram je užitočný najma vtedy, ak ho budeme potrebovať v programe viackrát na rôznych miestach.

Príklad: Máme finančný program, v ktorom sú uložené rôzne výstupné a vstupné faktury. Teraz musíme každú jednotlivú fakturu (ktorá je uložená na rôznych miestach v programe) prečítať a vytlačiť. O toto sa najlepšie postará podprogram.

Takýto podprogram sa napíše práve tak, ako iný program. Zabuduje sa do hlavného programu alebo sa zaradí za neho. Kedy bude vyvolaný a kedy bude končiť je pevne zadané príkazmi GO SUB a RETURN.

Tu máte jeden jednoduchý príklad na takýto podprogram: Program beží po riadok 40 normálne a potom skočí priamo na podprogram na riadku 100. Podprogram bude riadok po riadku spracovávaný, až počítač narazí na príkaz RETURN. Vtedy sa vráti na riadok, ktorý je bezprostredne pod riadkom 40, v našom prípade na riadok 50.

```
 10 REM GO SUB - RETURN
 20 POKE 23609,100: REM pipnutie klavesy
 30 INPUT "cislo: ";x
 40 GO SUB 100
 50 GO TO 30
100 REM tu zacina podprogram
110 PRINT x;" na druhu je ";x^2
120 RETURN
```

Na riadku 30 zadáme číslo do premennej x. Riadok 40 volá podprogram na riadku 100. Tam sa vypíše hodnota v premennej x a druhá mocnina z x. Riadok 120 vracia riadenie programu na riadok 50 (riadok za GO SUB), odtiaľ sa skáče na riadok 30 a celý cyklus začína znova.

To, že môžete s podprogramom previesť tiež niečo zaujímavého, uvidíte na ďalšom malom príklade hry:

```
 10 REM ZAVODENIE
 15 PAPER 5: BORDER 5: CLS
 20 FOR f=0 TO 21: PRINT TAB 2;"|": NEXT f
 25 LET a$="<<"
 30 LET pocitac=28
 40 LET clovek=28
 50 LET x=5
 55 BEEP .01,RND*50
 60 GO SUB 100
 70 LET x=0
 75 BEEP .01,RND*50
 80 GO SUB 100
 90 GO TO 50
100 IF x=5 THEN LET pocitac=pocitac-RND: PRINT AT x,pocitac; INK 6;A$:
    IF pocitac<2 THEN PRINT AT 0,7; PAPER 6; FLASH 1; BRIGHT 1;
    "Vyhral pocitac !": STOP
110 IF x-10 THEN LET clovek=clovek-RND: PRINT AT x,clovek; INK 2;A$:
    IF clovek<2 THEN PRINT AT 0,8; PAPER 6; FLASH 1;"Vyhral clovek!":
    STOP
120 RETURN
```

Program predvádza závody dvoch lodí. Vrchnú riadi počítač, v spodnej

je človek. Závody vyhrava ten, kto sa skorej dostane k cieľovej čiare. To kto vyhraje, samozrejme v tomto prípade nie je dôležité a riadi to náhoda. Skôr nám tu išlo o ukážku praktického využitia podprogramu.

Z V U K A H U D B A

Pre programovací jazyk ZX Spectra je príjemným obohatením príkaz BEEP. Môžeme ho používať na akustické zvýraznenie dôležitého hlásenia v programe ale najväčšie uplatnenie nájde v hrách. Môžeme pomocou neho simulovať zvuk odrážanej lopty, výbuch budovy po zásahu bombou či búracanie motora rakety pri štarte.

Na prvý pohľad sú možnosti tohoto príkazu dosť obmedzené, ale pri dôkladnom oboznámení s ním môžeme dosiahnuť prekvapivé efekty. Najlepšie efekty sa samozrejme dajú dosiahnuť ak sa zvuk ZX Spectra naučíme ovládať zo strojového kódu. Vtedy sa dajú aj v dosť obmedzených možnostiach zvuku ZX Spectra vytvoriť priam fantastické hudobné programy.

Reproduktor zabudovaný v našom počítači nie je veľmi hlasný. Preto ak chceme dosiahnuť kvalitnejšiu reprodukciu hudby, je vhodné pripojiť k výstupu EAR zosilňovač a zvuk vyviesť do veľkých reproduktorov.

Príkaz BEEP má dva parametre oddelené čiarkou. Prvý udáva dĺžku tónu a druhý jeho výšku.

Aké rozmanité zvuky môžete vytvoriť, budete počuť, ak do počítača zapíšete nasledujúci program a spustíte ho.

```
 5 REM NAHODNE ZVUKY
10 BEEP RND/RND/3,RND*60-35
15 BORDER RND*7: CLS
20 BEEP RND/RND/2,RND*80-45
25 BORDER RND*7: CLS
30 BEEP RND/RND/3,RND*130-65
35 PAPER RND*7: CLS
40 BEEP RND/RND/2,RND*40-5
45 GO TO 10
```

Môžete prirodzene zadať BEEP do slučky alebo dokonca použiť viac do seba vnorených slučiek a tým docieliť krajšie efekty.

```
10 REM SLUCKOVE ZVUKY
20 FOR a=-60 TO 60
30 FOR b=.01 TO .03 STEP .01
40 BEEP b,a: BEEP b,a/10*b: BEEP b, ABS (a)
50 NEXT b
60 NEXT a
```

Funkcia RND produkuje tóny, ktoré sú ohraničené limitom a môžu vytvoriť celkom osobitý ráz hudby.

```
10 LET dlzka=(INT (RND*8)+1)/30
20 LET vyska=INT (RND*24)-12
30 BEEP dlzka,vyska
40 IF RND >=.7 THEN GO TO 30
50 GO TO 10
```

Ako ste sa už možno dozvedeli v príručke k ZX Spectru, produkujú čísla výšok celkom určité tóny, napríklad číslo 0 je stredné C, 12 je o oktávu vyšší ton atď. Ďalší program mení Spectrum na malý klavír. Použijeme k tomu spodnú radu klávesnice. Pod klávesou "Z" sa bude ukrývať C, pod klávesou "X" bude D atd. Vyskúšajte si to.

```
 5 REM KLAVIR
10 POKE 23658,8: REM zmena na velke pismena
20 LET a=CODE INKEY$
30 IF a<66 OR á>90 THEN GO TO 20
40 LET a=12*(a=75)+2*(a=88)+4*(a=67)+5*(a=86)+7*(a=66)+9*(a=78)+11*
   (a=77)
50 BEEP .04,b
60 IF INKEY$<>"" THEN GO TO 50
80 GO TO 20
```

Ak sa Vám zdá vibrafónový spôsob hry namáhavý, môžete prispôsobiť klavír na automatiku. Ako vidíte v DATA riadku, potrebujete C-dur stupnicu, aby ste získali plne temperovaný klavír J.S.Bacha.

```
10 REM TEMPEROVANY KLAVIR
20 DIM a(8)
30 FOR b=1 TO 8
40 READ a(b)
50 NEXT b
60 LET b=INT (RND*8)+1
70 LET m=(INT (RND*4)+1)/10
80 BEEP m,a(b)
90 IF RND >.9 THEN GO SUB 120
100 GO TO 60
110 DATA 0,2.039,3.86,4.98,7.02.8.84,10.88,12
120 LET z=RND
130 LET m=a(1)*(z>=.5)+a(8)*(z<.5)
140 BEEP 1,m
150 PAUSE 25
160 RETURN
```

V poslednom programe tejto kapitoly hrajú tóny dôležitú úlohu v hre na postreh. Volá sa "Šimon" a je pre Vašu pamäť ťažkou skúškou. Počítač volí náhodne zvuky medzi hodnotami 1 a 4 a zobrazí na obrazovku farebný symbol a zodpovedajúci tón. Vtedy musíte stlačiť príslušnú klávesu. Ak odpoviete správne, počítač pridá jeden tón a teraz musíte už stlačiť dve klávesy v príslušnom poradí. Vyhrávate vtedy, ak si zapamätáte kombináciu sedem tónov. Ak skončíte skorej, počítač Vám zobrazí bodové vyhodnotenie.

```
 5 REM SIMON
10 LET a$=""
15 BORDER 7: PAPER 7: INK 0: CLS
20 FOR a=1 TO 7
30 PRINT AT 10,8;"Prosim cakajte..."
35 BORDER RND*7
40 LET a$=a$+STR$ (INT (RND*4)+1)
45 PAUSE 5
50 NEXT a
55 BORDER 7: CLS
```

```
 60 LET x=1
 70 FOR q=1 TO x
 72 LET l=4*(CODE a$(q)-48)
 73 LET t=VAL a$(q)
 75 BEEP .05,10*t
 80 PRINT AT 1,7; INK t;"==";AT 1+1,7;"==";AT 1-1,7;"=="
 85 BORDER RND*7
 90 PAUSE 20-x
100 PRINT AT 1,7; INK 6;"**";AT 1+1,7;"**";AT 1-1,7;"**"
102 PAUSE 4
105 CLS
110 NEXT q
120 FOR b=1 TO x
122 IF INKEY$<>"" THEN GO TO 122
123 LET t$=INKEY$
124 IF CODE t$=0 THEN GO TO 123
125 CLS
126 LET y=4*(CODE t$-48)
130 PRINT AT y,7; INK y/4;"##";AT y-1,7;"##";AT y+1,7;"##"
145 BEEP .04,2.5*y
146 IF CODE t$<>CODE (a$(b)) THEN GO TO 300
147 PAUSE 7
148 CLS
150 NEXT b
155 BORDER 7: PAPER 7: CLS
156 IF x=7 THEN PRINT AT 10,9;"Zvitazil si !": BORDER RND*7:
    INK RND*6: PAUSE 20: GO TO 156
160 LET x=x+1
165 PAUSE 50
170 GO TO 70
300 BORDER 7: PAPER 7: CLS
305 PRINT AT 6,9;"Prehral si !"
310 PRINT AT 10,8;"Tvoje body: ";x-1
315 BORDER RND*7
320 BEEP .02,RND*30
330 GO TO 315
```

D E F F N

Pomocou prikazu DEF FN si môžeme nadefinovať ľubovolnú komplikovanú funciu, ktorú nepozná BASIC ZX Spectra. Takúto funciu potom môžeme pomocou mena, ktoré sme si pre ňu zadefinovali vyvolať kedykoľvek počas programu. Takýto postup nám ušetrí pamäť, čas a celú radu zbytočných výpočtov.

Definovanie funkcie pomocou prikazu DEF FN má takýto tvar:

DEF FN jednopismenový_názov (jednopismenový_argument_funkcie)=vzorec

Vyzerá to priliš komplikovane a preto si to radšej ukážeme v praxi:

```
10 REM DEFINUJEME Y VO FUNKCII
20 DEF FN y(x)=x*x
30 INPUT "Zadaj cislo: ";x
40 PRINT x;TAB 6;FN y(x)
```

```
50 GO TO 30
```

Na riadku 20 definujeme funkciu y pomocou argumentu x, kde bude vypočítaná druhá mocnina x. Pokiaľ použijeme kdekoľvek v programe postupnosť FN y(x), výsledkom bude druhá mocnina x.

V ďalšom programe bude definovaná funkcia na riadku 70. Funkcia dáva odmocninu z rozdielu druhých mocnín dvoch premenných a v podprograme medzi riadkami 120 a 210 bude hodnota h (vid riadok 130), použitá ako PRINT pozícia pre body, ktorými sa vykreslí celý motýľ.

```
 10 REM MOTYL
 30 LET l=0
 40 LET p=10
 50 LET q=17
 60 LET b=0
 70 DEF FN a(b)=SQR (l*l-b*b)
 80 PAPER 8: CLS
 90 BORDR 2: INK 0
110 LET l=l+1
120 FOR b=0 TO l
130 LET h=FN a(b)
140 PRINT AT p+b,q+h;"."
160 PRINT AT p+b,q-h;"."
180 PRINT AT p-b,q-h;"."
190 PRINT AT p-b,q+h;"."
200 BEEP .02,50+h
210 NEXT b
220 IF l<11 THEN GO TO 110
230 PRINT INK 2;AT 2,17;"OO"
240 PRINT INK 2;AT 1,17;"oo"
250 PRINT INK 2;AT 0,17;")("
```

D I M A P O L I A

Príkaz DIM sa používa na vymedzenie (dimenzovanie) oblasti pamäti, ktorá bude slúžiť pre uloženie číselných alebo reťazcových premenných. Takáto oblasť pamäti sa nazýva ARRAY (pole).

Pri používaní za príkazom DIM nasleduje jedno písmeno, ktoré označuje názov poľa a v zátvorkách číslo, ktoré určuje počet prvkov poľa. Tak napríklad DIM x(5) vymedzuje číselné pole x, ktoré obsahuje 5 prvkov; DIM y$(100) vymedzuje reťazcové pole y, ktoré obsahuje 100 prvkov.

Zadajte nasledujúci program, ktorý Vám všetko objasní:

```
10 REM CISELNE POLE
20 DIM b(4)
30 FOR a=1 TO 4
40 LET b(a)=INT (RND*9)+1
50 NEXT a
60 FOR a=1 TO 4
70 PRINT TAB 6;"b(";a;") je ";b(a)
80 NEXT a
```

Ako vidíte na riadku 20, potrebuje počítač dimenziu poľa v tvare DIM skôr, ako ho bude používať.

Polia, o ktorých sme teraz hovorili, boli jednorozmerne ako je napr. rada čísiel. Môžeme ale použiť aj viacrozmerne pole. Pri dimenzovaní treba potom odpovedajúce čísla jednotlivých rozmerov v zátvorkách oddeliť čiarkami. Pozrime sa v nasledujúcom programe na dvojrozmerne pole:

```
 10 REM DVOJROZMERNE POLE
 30 DIM a(4,4)
 40 FOR b=1 TO 4
 50 FOR c=1 TO 4
 60 LET a(b,c)=INT (RND*9)+1
 70 PRINT "a(";b;",";c;") je ";a(b,c)
 80 NEXT c
 90 NEXT b
100 PRINT AT 16,16;"1  2  3  4"
110 FOR b=1 TO 4
120 PRINT TAB 13;b;TAB 16;a(b,1);"  ";a(b,2);"  ";a(b,3);"  ";a(b,4)
130 NEXT b
```

Ak program spustíte, dostanete výsledok, ktorý vyzerá asi takto:

```
a(1,1) je 4
a(1,2) je 9
a(1,3) je 5
a(1,4) je 5
a(2,1) je 1
a(2,2) je 3
a(2,3) je 2
...... .. .
...... .. .
...... .. .
a(4,4) je 8
```

	1	2	3	4
1	4	9	5	5
2	1	3	2	7
3	1	5	1	2
4	7	6	6	8

Program Vám vypíše všetky prvky dvojrozmerného poľa. Ak si pole ešte stále neviete dosť dobre predstaviť, pomôže nám časť programu na riadkoch 100 až 130. Tento úsek programu zobrazí malú tabuľku ktorá prehľadne ukazuje, ako sú jednotlivé prvky usporiadané. Každý jednotlivý rez poľom je určený svojimi súradnicami, napr. prvok 2,2 nám dáva číslo 3. A skutočne, aj v zobrazenom výpise vidíme, že prvok a(2,2) je 3.

Dimenzácia poľa potrebuje určitý pamäťový priestor, ktorý sa vymedzí aj keď niektoré prvky poľa vôbec nenaplníme. Je dobré si preto vymedziť pole presne také aké v skutočnosti potrebujeme, inak nám zbytočne zaberá pamäť, ktorú už nemôžeme na nič iné použiť. Počet prvkov sa dá ľahko vypočítať tak, že prvé číslo v zátvorkách vynásobíme druhým.

Dvojrozmerné pole samozrejme nie je hranicou. Môžeme používať aj troj a viac rozmerné polia. Na nasledujúcom programe si ukážeme príklad trojrozmerného poľa.

```
 10 REM TROJROZMERNE POLE
 30 DIM a(3,3,3)
```

```
 40 FOR b=1 TO 3
 50 FOR c=1 TO 3
 55 FOR d=1 TO 3
 60 LET a(b,c,d)=INT (RND*9)+1
 70 PRINT "a(";b;",";c;",";d;") je ";a(b,c,d)
 75 NEXT d
 80 NEXT c
 90 NEXT b
```

V ďalšom príklade skúsime zvačšiť počet rozmerov na 5. Ak sú jednotlivé dimenzie zložené z dvoch elementov, obsahuje pole celkom 32 prvkov (2 x 2 x 2 x 2 x 2).

```
 10 REM PATROZMERNE POLE
 30 DIM a(2,2,2,2,2)
 40 FOR b=1 TO 2
 50 FOR c=1 TO 2
 60 FOR d=1 TO 2
 70 FOR e=1 TO 2
 80 FOR f=1 TO 2
 90 LET A(b,c,d,e,f)=INT (RND*9)+1
100 PRINT "a(";b;",";c;",";d;",";e;",";f;") je ";a(b,c,d,e,f)
110 NEXT f
120 NEXT e
130 NEXT d
140 NEXT c
150 NEXT b
```

H Á D A N I E Č Í S L A

V tejto logickej hre uvidíme, ako je použité v praxi jednorozmerné pole. Hra funguje jednoducho: Počítač zvolí kombináciu štyroch čísiel a my ich máme uhádnut na desať pokusov. Správne uhádnuté čísla na zlom mieste odmeňuje počítač bielym bodom, za správne čísla na správnom mieste dáva čierny bod. V kombinácii sa nesmie používať nula !

```
 10 REM HADANIE CISIEL
 20 DIM c(4)
 30 DIM g(4)
 40 PRINT 'PAPER 2; INK 6; TAB 2;"Myslim si stvormiestne cislo, "
 50 PRINT PAPER 2; INK 6; TAB 2;"mas 10 pokusov na uhadnutie.  "
 70 PRINT AT 21,2;"Ak mozme zacat stlac klavesu"
 90 PAUSE 4e4
100 CLS
110 LET c(1)-INT (RND*9)+1
120 FOR z=2 TO 4
130 LET c(z)=INT (RND*9)+1: BEEP .5,c(z)
140 FOR j=1 TO z-1
150 IF c(j)=c(z) THEN GO TO 130
160 NEXT j
170 NEXT z
180 FOR g=1 TO 10: BEEP 1,2* g
200 PRINT AT 1,8; PAPER 1; INK 7;"Pokus cislo: ";g
210 INPUT "Vloz stvormiestne cislo: ";a
```

```
220 LET a1=a
230 FOR z=1 TO 4
240 LET g(z)=a-10*INT (a/10)
250 LET a=INT (a/10)
260 NEXT z
270 LET b=0
280 LET w=0
290 FOR z=1 TO 4
300 IF c(z)<>g(z) THEN GO TO 330
310 LET b=b+1: BEEP .02,10*b
320 LET g(z)=0
330 NEXT z
340 FOR z=1 TO 4
350 IF g(z)=0 THEN GO TO 400
360 FOR j=1 TO 4
370 IF c(z)<>g(j) THEN GO TO 390
380 LET w=w+1: BEEP .02,7*w
390 NEXT j
400 NEXT z
420 PRINT AT g+3,1; INK 2;a1;TAB 10; INK 0;b;" x cierna";"  ";
440 PRINT INK 2;w;" x biela"
460 PRINT
470 IF b=4 THEN PRINT " Kombinaciu si uhadol na ";g;". krat"
480 IF b<>4 THEN NEXT g
490 PRINT '" Cislo bolo: "; INK 2;c(4);c(3);c(2);c(1)
```

R E Ť A Z C O V É P O L I A

Reťazcové polia fungujú práve tak, ako polia numerické. Uvidíte to v nasledujúcom programe. Spustite ho a zadajte pomocou príkazu INPUT štyri ľubovoľne slová.

```
10 REM RETAZCOVE POLIA
20 DIM a$(4,10)
30 FOR b=1 TO 4
40 INPUT "Zadaj slovo: ";a$(b)
50 NEXT b
60 FOR b=1 TO 4
70 PRINT "a$(";b;") je ";a$(b)
80 NEXT b
```

Ako vidíte, jediný rozdiel pri dimenzovaní reťazcového poľa je v tom, že za písmenom označujúcim názov poľa nasleduje znak $ (dolár). Druhé číslo v tomto prípade určuje koľko znakov maximálne môžu mať jednotlivé reťazce.

Nasledujúci program môžete často využiť aj v praxi. Triedi reťazce podľa abecedného poradia. Ak chcete triediť dlhšie reťazce alebo väčší počet, stačí zmeniť hodnoty na riadkoch 20, 40 a 50.

```
10 REM TRIEDENIE RETAZCOV
20 DIM w$(5,10)
30 LET b=0
40 LET g=5
```

```
 50 FOR a=1 TO 5
 60 INPUT w$(a)
 70 PRINT w$(a)
 80 NEXT a: PRINT ''
 90 LET z=1
100 LET b=z+1
110 IF b>g THEN GO TO 190
120 IF w$(g)>w$(z) THEN GO TO 150
130 LET z=z+1
140 GO TO 100
150 LET q$=w$(z)
160 LET w$(z)=w$(b)
170 LET w$(b)=q$
180 GO TO 130
190 PRINT w$(g)
200 LET g=g-1
210 IF g>0 THEN GO TO 90
```

P R Á C A S R E Ť A Z C A M I

Reťazce, s ktorými sme sa stretli, sa skladajú z rady alfanumerických znakov (písmená a čísla), ale môžu obsahovať tiež matematické znaky, prázdny priestor a grafické symboly. Napevno sa reťazec definuje tak, že názvu reťazca priradíme hodnotu uzavretú v zátvorkách v tvare: LET a$="slovo". Výnimkou v ukladaní do reťazca sú práve úvodzovky, ktoré ohraničujú jeho veľkosť. Možno ich do reťazca zadávať, ale zdvojené. Skúste si napísať a spustiť nasledujúci riadok:

```
10 LET a$="""""""": PRINT a$
```

V BASICu Spectra existuje celý rad užitočných funkcií na prácu a manipuláciu s reťazcami a ich časťami. Tu je ich prehľad:

**CODE x$** — Dáva ASCII kód prvého znaku z x$. Napríklad ak x$="Micro" dá PRINT CODE x$ číslo 77, pretože "M" má ASCII kód 77.

CHR$ n — Je opakom funkcie CODE. Zadajte napr. PRINT CHR$ 77 a dostanete znak "M", ktorý zodpovedá CODE 77.

x$(n TO m) — Dáva časť reťazca od n do m. Napr. x$(2 TO 4) dá "icr".

x$(3 TO) — Dáva časť reťazca X$ od tretieho znaku po koniec. V našom prípade x$(3 TO) je "cro".

x$(TO 3) — Dáva časť reťazca x$ od začiatku po tretí znak. Ak x$="Micro", potom x$(TO 3) je "Mic".

LEN x$ — Udá dĺžku reťazca, v našom prípade je to číslo 5.

STR$ x — Zmení numerickú premennú x na reťazec. Ak napr. x=234, potom dáva STR$ x reťazec "234".

**VAL x$** — Je funkcia opačná k STR$. Mení numerický obsah reťazca späť na číslo. Ak reťazec x$="22+34" dáva VAL x$ číslo 56

Pomocou nasledujuceho programu si ukážeme prehlad týchto funkcii v praktickom využiti.

```
10 PRINT "LET x$=""Micro"""
20 LET x$="Micro"
30 PRINT "CODE x$= ";CODE x$
40 PRINT "CHR$ 77= ";CHR$ 77
50 PRINT "x$(3 TO)= ";x$(3 TO)
60 PRINT "x$(TO 3)= ";x$(TO 3)
70 PRINT "x$(2 TO 4)= ";x$(2 TO 4)
80 PRINT "LEN x$= ";LEN x$
90 PRINT "LET x$= ""22+34"""
100 LET x$="22+34"
110 PRINT "VAL x$= ";VAL x$
120 PRINT "LET x=56"
130 LET x=56
140 PRINT "LET x$=STR$ x"
150 LET x$=STR$ x
160 PRINT "x$= ";x$
```

P O U Ž I T I E F U N K C I E L E N

Funkciu LEN môžete velmi prakticky využiť napriklad v nasledujúcom príklade: Vypisujete na obrazovku reťazce rozličnej dĺžky a všetky potrebujete podčiarknuť. Pomocou funcie LEN sa to dá dosiahnuť velmi jednoducho.

```
10 INPUT "Zadaj slovo: ";a$: PRINT a$
20 FOR a=1 TO LEN a$
30 PRINT "-";
40 NEXT a
50 PRINT '': GO TO 10
```

Nasledujúca metóda je elegantnejšia a prebieha oveľa rýchlejšie.

```
10 INPUT "Zadaj slovo: ";a$: PRINT a$
20 PRINT "--------------------------------" (TO LEN a$)
30 PRINT ': GO TO 10
```

Pri oboch metódach samozrejme môžeme zadávať text dlhý maximálne 32 znakov, čo je dĺžka riadku.

P O U Ž I T I E F U N K C I E S T R $

Nemenej dôležitá je funkcia STR$. Pripomeňme si, že dokaže zmeniť číslo na reťazec.

```
10 PRINT 2
20 PRINT STR$ 2
30 PRINT 1/3
40 PRINT STR$ (1/3)
50 PRINT 9E15
60 PRINT STR$ 9E15
```

Ako vidíte, do funkcie STR$ možno zapísať i celý vzorec, alebo číslo vo vedeckej notácii.

Možno Vás teraz zamestnáva otázka: "No dobre, ale ako to využijem ?". Tu sú príklady pre praktickú aplikáciu funkcie STR$:

1) Zarovnávanie čísiel podľa desatinnej bodky.

```
10 LET a=RND*100
20 PRINT a
30 LET a=a*10
40 GO TO 20
```

Po spustení tohoto programu sa Vám vypíšu výsledky. Sú správne, ale asi by Vám viac vyhovovalo, keby boli zarovnané prehľadnejšie, podľa desatinnej bodky. O to sa postará tento program:

```
10 LET a=RND*100
20 PRINT TAB 15-LEN STR$ INT a;a
30 LET a=a*10
40 GO TO 20
```

Pozrime sa, ako to funguje. Prvá hodnota v premennej a je povedzme 69.433594. Program vypočíta celočíselnú hodnotu (Integer) z a (v našom prípade INT a=69), toto číslo premení na reťazec a zistí funkciou LEN ako je dlhý. Dĺžka je v našom prípade 2, pretože číslo 69 sa skladá z dvoch znakov. Táto hodnota sa zoberie do funkcie TAB, čím je pevne stanovené, kde musí byť prvá printová pozícia.

2) Výpis desatinného čísla s konkrétnym počtom desatinných miest za desatinnou bodkou.

```
10 LET a=RND
20 LET a$=STR$ a
30 IF a$(1)="." THEN LET a$="0"+a$
40 LET b=LEN STR$ INT VAL a$
50 PRINT TAB 15-b;(a$+("." AND b=LEN a$)+"000")(TO b+4)
60 LET a=a*10
70 GO TO 20
```

Takto sa Vám budú všetky čísla vypisovať na tri desatinné miesta. Ak chcete zaviesť iný počet miest za desatinnou bodkou, potom musíte previesť na riadku 50 nasledujúce zmeny:

- zadať počet desatinných miest (pomocou núl v reťazci)
- v poslednom výraze (TO b+4) namiesto čísla 4 dosadiť Vaše číslo

Prejdime si ešte raz všetky riadky:

```
10  generuje náhodné číslo a vkladá ho do premennej a
20  prevedie premennú a do reťazca a$
30  ak je číslo menšie ako nula a nula vpredu chýba, doplní ju
40  vypočíta hodnotu b, ktorá je rovná počtu celých desatinných miest
50  pomocou TAB mení PRINT pozíciu a dosadzuje desatinné miesta
60  vynásobí premennú a číslom 10
```

70 skok na riadok 20

3) Šetrenie miesta v pamati.

Ak čísla v numerických premennych uschovate do reťazca a neskôr pred použitím ich vrátite pomocou funkcie VAL do pôvodného tvaru, ušetríte pamäť.

```
LET a$=STR$ (1024)
```

potom napíšte

```
PRINT VAL a$
```

Okrem toho je celkom praktické, ak v programe do reťazcov ukladáte jednotlivé funkcie či vzorce, napríklad:

```
LET a$="RND*6"
```

a potom si už len vyberiete číslo do potrebnej premennej

```
LET r=VAL a$
```

I N K E Y $

Ako Vám ukáže nasledujúci program, môžete sa zaobísť aj bez tlačenia klávesy ENTER, ak zavediete do programu funkciu INKEY$.

Ak ste program napísali do počítača a spustili, stláčajte čísla medzi 1 a 9. Na obrazovke sa okamžite objaví zodpovedajúce hlásenie. Ak si prajete ukončiť program, stlačte číslo 0 a potom dostanete poslednú správu a konečne príkaz STOP uzavrie program.

```
10 REM INKEY$
20 PAUSE 0
40 LET a$=INKEY$
50 PRINT "Stlačili ste číslo ";a$
60 IF a$="0" THEN STOP
70 GO TO 20
```

Nasledujúci program "Jasnovidec" potrebuje rovnako príkaz INKEY$. Snažíte sa v ňom uhádnuť aké číslo bude generovať počítač (medzi 1-9). Čím menší máte počet pokusov, tym ste lepší jasnovidec.

```
10 REM JASNOVIDEC
20 LET e=9
30 LET q=0
40 PAUSE 0
50 LET a$=INKEY$
60 PRINT AT 8,5; INK RND*7; PAPER 9;"Tvoje číslo je: ";a$
70 LET q=q+1: BEEP .02,q
80 LET w$=STR$(INT (RND*9)+1)
90 PRINT AT 10,5; INK RND*7; PAPER 9;"Moje číslo je:  ";w$
100 PRINT AT 14,5; INK RND*7; PAPER 9; BRIGHT 1;"Pokus číslo: ";q
```

```
110 IF w$=a$ THEN BEEP .01,RND*30: GO TO 100
120 GO TO 40
```

Ďalší program nám nakreslí jednoduchý labyrint. Našou úlohou je dostať sa z neho von. Naše postavenie označuje červený krúžok (na začiatku v pravom dolnom rohu). Východ je v ľavom hornom rohu a je označený blikajúcou šípkou. Nášho hrdinu ovládame pomocou týchto kláves:

o - vľavo p - vpravo q - hore a - dole

Program je veľmi jednoduchý a ani neošetruje či švindľujete (teda či neprechádzate cez steny). Ak sa Vám zapáči, môžete si do neho túto sekciu (a iné vylepšenia) dorobiť a môžete z neho spraviť celkom peknú hru.

```
 10 REM LABYRINT
 70 BORDER 0: PAPER 7: CLS
 80 LET s=0
 90 FOR y=1 TO 704
100 LET t=INT (RND*3)
110 PRINT INK 0;("#" AND t=0)+(" " AND T<>0);
120 NEXT y
130 PRINT AT 0,0; INK 4; FLASH 1;"´"; FLASH 0;" ";AT 1,0;"   "
140 LET x=21: LET y=31
150 LET m=x
160 LET n=y
170 PRINT AT x,y; INK 2;"O"
180 LET s=s+1
190 LET a$=INKEY$
200 IF a$="" THEN GO TO 190
210 IF a$="q" AND x>0 THEN LET x=x-1
220 IF a$="a" AND x<21 THEN LET x=x+1
230 IF a$="o" AND y>0 THEN LET y=y-1
240 IF a$="p" AND y<31 THEN LET y=y+1
250 IF x=0 AND y=0 THEN GO TO 280
260 PRINT AT m,n;" "
270 GO TO 150
280 PRINT AT 10,3; PAPER 2; INK 6;"Vyborne, nasiel si vychod !"
290 PRINT AT 12,4; PAPER 2; INK 7;"Potreboval si ";s;" krokov."
300 FOR f=0 TO 10: BEEP .01+(f/100),f*3: NEXT f
310 FOR f=0 TO 1000: NEXT f: RUN
```

Ešte raz sa vrátime k riadku 110. Ak t=0, bude nakreslený znak pre stenu ("#"), v opačnom prípade (teda ak t<>0) bude nakreslená medzera. Takéto úlohy sa u iných počítačov riešia pomocou IF (THEN) ELSE, ale toto riešenie je tiež celkom elegantné.

Teraz Vám ukážeme ešte jeden krátky podprogram. Môžete ho použiť v tých Vašich programoch, kde je nutné rýchlo sa rozhodovať, či sa má vykonať istá operácia. (V našom prípade či sa má program spustiť odznova).

```
 10 FOR F=1 TO 100
 20 LET a$=INKEY$
 30 IF a$="r" THEN GO TO 60
 40 NEXT F
 50 STOP
```

```
60 PRINT "Start odznova"
70 FOR f=1 TO 300: NEXT f: RUN
```

Po spustení programu máte približne 2 sekundy na to, aby ste stačili klávesu "r". Ak to stihnete, program beží od začiatku, v opačnom prípade sa zastaví na riadku 50.

Keďže hovoríme o čase, povieme si ešte o jednej dôležitej veci, ktorá s ním súvisí. Je to časový takt počítača, ktorý limituje rýchlosť všetkých procesov, ktoré sa v ňom odohrávajú. Tento je uložený ako 16-bitové číslo v systémových premenných na adresách 23672 a 23673.

Tento "Timer" si môžeme nastaviť pomocou uloženia hodnoty 255 na tieto adresy: POKE 23672,255: POKE 23673,255.

Ak napíšete nasledujúci krátky program, dostanete pomerne presný sekundový čas.

```
10 POKE 23672,255: POKE 23673,255.
20 LET t=(256*PEEK 23673+PEEK 23672)/50
30 PRINT AT 10,10;t
40 GO TO 20
```

A tu tento nápad rozvinieme a dostaneme kompletné digitálne hodinky:

```
10 REM DIGITALNE HODINKY
20 BORDER 1: PAPER 1: INK 7: CLS
30 INPUT "Zadaj hodiny: ";h
40 INPUT "Zadaj minuty: ";m
50 LET s=0
60 POKE 23673,255
70 POKE 23672,255
80 LET h$=("0"+STR$ h AND h<10)+(STR$ h AND h>=10)
90 LET m$=("0"+STR$ m AND m<10)+(STR$ m AND m>=10)
100 LET s$=("0"+STR$ s AND s<10)+(STR$ s AND s>=10)
110 PRINT AT 1,11;h$+":"+m$+":"+s$
120 LET s=INT ((256*PEEK 23673+PEEK 23672)/50)
130 IF s>59.9999 THEN LET m=m+1: POKE 23673,255: POKE 23672,255
140 IF m=60 THEN LET h=h+1: LET m=0
150 IF h=13 THEN LET h=1
160 GO TO 80
```

Aby ste dobre pochopili čo sa deje na riadkoch 60, 70, 120 a 130, zalistujte si trochu pred touto kapitolou v kapitole, ktorá pojednáva o príkazoch PEEK a POKE.

R E A D (D A T A) R E S T O R E

READ a DATA sú veľmi užitočné príkazy pre prácu s dátovými poliami. Pre ich rýchle pochopenie bude najlepšie, ak zadáte do počítača nasledujúci program a pozriete sa naň.

```
10 REM READ DATA
20 REM -------------
30 REM CITANIE Z DAT
```

```
 40 REM -------------
 50 DIM b(5)
 60 FOR a=1 TO 5
 70 READ b(a)
 80 NEXT a
 90 REM ------------------
100 REM SPATNE VYPISOVANIE
110 REM ------------------
120 FOR c=5 TO 1 STEP-1
130 PRINT b(c)
140 NEXT c
150 DATA 13,35241,88,2,199999
```

Na riadku 70 sa nachádza príkaz READ. Keď na neho počítač narazí, ide k prvému výrazu, ktorý sa nachádza za príkazom DATA a prečíta ho. Hodnota, ktorú v tomto prípade nájde, určuje prvý prvok pola.

DATA môžete zaradiť kdekoľvek do programu. Ale pozor, ak je v programe viac príkazov DATA, príkaz READ bude čítať hodnoty z prvého príkazu DATA, ktorý nájde.

Pozrite sa ešte raz krátko na program "Raketa" na začiatku tejto knihy. V prvom programe tejto kapitoly tvoria DATA čísla, ktoré čítame pomocou READ ako numerické premenné. V programe "Raketa" tvoria DATA reťazce (riadok 220), ktoré sú načítané do reťazcovej premennej A$ a v riadku 120 budú zobrazované pomocou slučky na obrazovke.

Veľmi dôležitý pre prácu s dátami je aj príkaz RESTORE. Pomocou neho počítač vie, že sa musí vrátiť na začiatok dát. Prezrite si ďalší program, ktorý predvádza použitie príkazu RESTORE v praxi.

```
 10 REM READ DATA RESTORE
 20 REM -------------
 30 REM CITANIE Z DAT
 40 REM -------------
 50 DIM b$(21,5)
 60 FOR a=1 TO 21
 70 READ b$(a)
 80 IF 3*INT (a/3)=a THEN RESTORE
 90 NEXT a
100 REM ---------
110 REM VYPIS DAT
120 REM ---------
130 FOR c=21 TO 1 STEP-1
140 PRINT b$(c);",";
150 NEXT c
160 DATA "jeden","dva","tri"
```

V tomto programe je v DATA uložená len skupina troch reťazcov. Pomocou príkazu RESTORE dochádza k ich opakovanému načítaniu do reťazcovej premennej b$, ak sú už raz prečítané. Riadok 80 zabezpečuje, že je príkaz RESTORE vykonaný vždy v pravú chvíľu. Všimnite si, že reťazcové výrazy musia byť v DATA vždy v úvodzovkách a samozrejme oddelené čiarkou.

Ako sme si už povedali, DATA môžu byť uložené na ľubovoľnom mieste programu. Ak narazí počítač bez READ príkazu v priebehu spracovania programu na DATA, nespozorujete žiadnu reakciu, ako keby tu bol REM

riadok. Zmeňte v predchadzajúcom programe riadok 160 na riadok 25. Uvidíte, že DATA budu načítane uplne rovnako ako pri vychodzom postavení. Dokonca, aj ked su DATA v programe rozhádzane, poskladá ich počítač dohromady, ako ukazuje nasledujúci program:

```
 10 REM READ DATA RESTORE
 20 DATA "jeden",1
 30 DIM b$(21,5)
 40 DIM z(21)
 50 FOR a=1 TO 21
 60 READ b$(a),z(a)
 70 IF 3*INT (a/3)=a THEN RESTORE
 80 DATA "dva"
 90 NEXT a
100 DATA 2,"tri"
110 FOR c=1 TO 21
120 PRINT b$(c),z(c)
130 DATA 3
140 NEXT c
```

Mimoriadne dôležite je množstvo výrazov uložených v DATA, pretože pri každom príkaze READ je hladaný nový výraz. Vymažte teraz v poslednom programe riadok 130 a odštartujte ho ešte raz. Dostanete chybové hlásenie "E Out of DATA, 60:1" v postavení, ked počítač prečital dva dátové výrazy a treti nemôže nájsť a súčasne nenašiel žiadny príkaz RESTORE, ktorý by ho vrátil späť na začiatok DATA.

Na našom príklade vidíte, že pre chod programu nie je dôležité, aby boli DATA v blízkosti príkazu READ. Ale program je prehľadnejší ak si ich uložíte na jedno miesto napríklad na koniec programu.

U Ž I V A T E Ľ O M D E F I N O V A N Á G R A F I K A

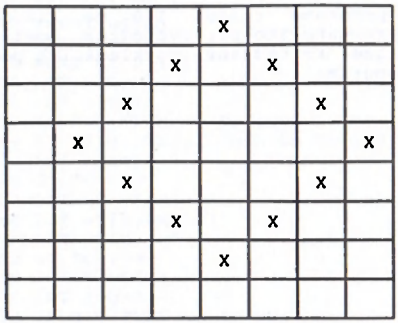
Jednou z fascinujúcich vlastnosti ZX Spectra je možnosť vytvárania špeciálnych užívateľom definovaných znakov. Špeciálne znaky sa dajú vytvárať veľmi jednoducho a umožňuju Vám graficky vylepšiť vizuálnu časť Vášho programu. V tejto kapitole chceme ukázať v silne zjednodušenej počítačovej hre použitie užívateľom definovanej grafiky v akcii. Naša hra sa volá "Dotman". Dotman má možnosť získať mnoho bodov, ale musí sa mať na pozore pred duchom, ktorý sa mu snaží zbieranie bodov prekaziť.

Než sa pustíme do tvorby hry "Dotman", povieme si o užívateľom definovanej grafike (UDG) niečo všeobecne.

Na miesto každého písmena od A po U môžeme uložiť jeden UDG znak. Ak zapneme grafický mód a stlačíme príslušné písmeno, tento znak sa nám zobrazí. Prirodzene sadu UDG znakov stratíme, ak vypneme počítač, ale UDG znaky zostanú zachované pri príkaze NEW.

Aby sme mohli uložiť na miesto písmena grafický znak, použijeme krátku slučku s príkazom POKE. Ako náš prvý príklad skúsme napríklad toto: stlačením tlačítka "A" chceme zobraziť malý diamant.

Každý znak Spectra sa skladá z 8 bytov. Každý z týchto 8 bytov sa ďalej skladá z 8 bitov. Takto teda dostaneme šachovnicu o stranách 8 x 8. Každé políčko tejto šachovnice môže byť bud obsadené (príslušný bit má hodnotu 1), alebo neobsadené (príslušný bit má hodnotu 0). Najjednoduchšie teda bude nakresliť si žiadaný znak najskôr na štvorčekový papier.



Ak sme si ho nakreslili, nie je už nič ľahšie, ako pomocou príkazu POKE znak uložiť pod príslušné písmeno. Pre zjednodušenie využijeme funkciu BIN, ktorá nám umožňuje zadávať čísla v binárnom tvare. Vyskúšajte si nasledujúci podprogram a potom si porovnajte binárne DATA s kresbou diamantu.

```
10 REM DIAMANT
20 FOR j=0 TO 7
30 READ q
40 POKE USR "A"+j,q: REM pismeno "A" napiste v mode GRAPH !!!
50 NEXT j
60 PRINT "A": REM pismeno "A" napiste v mode GRAPH !!!
70 DATA BIN 00001000,BIN 00010100,BIN 00100010,BIN 01000001,
   BIN 00100010,BIN 00010100,BIN 00001000,BIN 00000000
```

Zapnite teraz mód GRAPH a podržte klávesu "A". Na obrazovke sa Vám objaví celé diamantové pole. Skúste teraz riadok 70 vymeniť za nasledovný:

```
70 DATA BIN 00110000,BIN 01110000,BIN 01111110,BIN 01011111,
   BIN 10011111,BIN 00010010,BIN 00010010,BIN 00000000
```

Ak teraz podržíte klávesu "A" v mode GRAPH, na obrazovke sa zjaví rada malých slonov.

Ale teraz sa vráťme späť k našej hre "Dotman". Chceme dať hre určitú grafickú formu a tak najskôr napíšeme podprogram, v ktorom definujeme UDG, ktoré v programe potrebujeme. Naša hra je skutočne jednoduchá: Na hracej ploche 14 x 14 polí je Dotman, ktorým pohybuje hráč a duch, ktorým pohybuje počítač. Ďalej je určené, kolko má Dotman pozbierať bodov. Steny hracej plochy nie sú prechodné ani pre Dotmana, ani pre ducha.

D O T M A N

Štruktúra programu vyzerá asi takto:

```
    1-999: odskoky do rôznych podprogramov
1000-1999: ovladanie Dotmana
2000-3999: posuvanie ducha
4000-4999: duch pristihne Dotmana alebo Dotman dosiahne 1000 bodov
5000-5999: definovanie grafiky Dotmana
6000-6999: definovanie grafiky ducha
7000-7999: zobrazenie hracieho poľa
8000-8999: zadanie premenných
```

Ak program prebehne celý, vracia sa zase na riadky 1000 až 3999 pokiaľ duch nepristihne Dotmana. Vtedy sa program ukončí na riadku 4000.

Premenná ga určuje horizontálnu pozíciu ducha. gd jeho priečne súradnice. ega a egd sú premenné, v ktorých je stará pozícia ducha, vždy vymazávaná, keď sa má duch pohybovať ďalej. Podobne su určené súradnice dd a da pre Dotmana, edd a eda sú vždy posledné Dotmanové pozície, ktoré budu vymazané, ak Dotman vykročí o krok ďalej. Dotmana ovládame pomocou kláves:

o - vľavo p - vpravo q - hore a - dole

Napíšme teraz podprogramy, ktorými sa duch a Dotman pohybujú.

```
  10 GO SUB 8000
  20 GO SUB 7000
  30 GO SUB 5000
  40 GO SUB 6000
1000 REM pohyby DOTMANA
1010 REM grafika DOTMANA (pismena B,C,D,E v grafickom mode !!!)
1020 PRINT AT edd,eda;" "
1030 PRINT AT dd,da;a$
1035 LET edd=dd:LET eda=da
1040 IF INKEY$="p" THEN IF da<13 THEN IF SCREEN$ (dd,da+1)<>"#" THEN
     LET da=da+1: LET a$="B"
1050 IF INKEY$="o" THEN IF da>0 THEN IF SCREEN$ (dd,da-1)<>"#" THEN
     LET da=da-1: LET a$="C"
1060 IF INKEY$="q" THEN IF dd>0 THEN IF SCREEN$ (dd-1,da)<>"#" THEN
     LET dd=dd-1: LET a$="E"
1070 IF INKEY$="a" THEN IF dd<13 THEN IF SCREEN$ (dd+1,da)<>"#" THEN
     LET dd=dd+1: LET a$="D"
1980 IF RND>=.2 THEN GO TO 1000
1990 REM pohyb a grafika DUCHA (pismeno G v grafickom mode !!!)
2000 PRINT AT egd,ega;" "
2010 PRINT AT gd,ga;"G"
2015 LET egd=gd: LET ega=ga
2020 IF dd<gd THEN IF SCREEN$ (gd-1,ga)<>"=" THEN LET gd=gd-1
2030 IF dd>gd THEN IF SCREEN$ (gd+1,ga)<>"=" THEN LET gd=gd+1
2040 IF da>ga THEN IF SCREEN$ (gd,ga+1)<>"=" THEN LET ga=ga+1
2050 IF da<ga THEN IF SCREEN$ (gd,ga-1)<>"=" THEN LET ga=ga-1
2999 GO TO 1000
4999 STOP
5999 RETURN
6999 RETURN
```

```
7000 REM kresba hracej plochy
7010 FOR g=1 TO 14
7020 FOR h=1 TO 14
7030 PRINT ".";
7040 NEXT h
7050 PRINT
7060 NEXT g
7070 PRINT AT 2,7;"######"
7080 PRINT AT 3,3;"=";AT 3,11;"="
7090 PRINT AT 4,3;"#";AT 4,11;"#"
7100 PRINT AT 5,3;"=###";AT 5,9;"###"
7110 PRINT AT 6,6;"#";AT 6,9;"#"
7120 PRINT AT 7,6;"#";AT 7,9;"#"
7130 PRINT AT 8,3:"#";AT 8,6;"#";AT 8,9;"#"
7140 PRINT AT 10,3;"#"
7150 PRINT AT 11,3;"####"
7160 PRINT AT 12,9;"###"
7170 RETURN
8000 REM definicia premennych
8010 LET da=0
8020 LET dd=0
8030 LET eda-0
8040 LET edd=0
8050 LET ga=13
8060 LET gd=13
8070 LET ega=13
8080 LET egd=13
8090 LET body=0
8100 LET a$="B"
8999 RETURN
```

Ak program správne opíšete do počítača a spustite ho, hra by už mala fungovať. Pravdepodobne budete sklamaný jej grafikou, ale nenechajte sa zmýliť. Toto je zatiaľ len hrubá kostra programu, na ktorú budeme postupne nabalovať ďalšie podprogramy a hra bude postupne dostávať celkom inú tvár. Ako prvé pridajte tieto dva riadky:

```
7065 INVERSE 1
7165 INVERSE 0
```

Ak program teraz znova spustíte, zbadáte, že stena na hracej ploche sa vykresľuje inverzne.

Ale teraz spravíme podstatnejšiu úpravu grafiky. Nadefinujeme štyri znaky UDG pre Dotmana, aby sa jeho ústa obracali vždy smerom k bodu, ktorý bude požierať. Na tomto základe bude potom zmenená premenná a$, ktorá predstavuje Dotmana v riadkoch 1040 až 1070.

Pre definovanie Dotmana teda doplníme program o tieto riadky:

```
5000 REM definovanie DOTMANA (pismena B,C,D,E v grafickom mode !!!)
5010 FOR j=0 TO 7
5020 READ q
5030 POKE USR "B"+j,q
5040 NEXT j
5050 DATA BIN 00111100,BIN 01111111,BIN 11111100,BIN 11110000,
```

```
      BIN 11111000,BIN 11111100,BIN 01111111,BIN 00111100
 5060 FOR j=0 TO 7
 5070 READ q
 5080 POKE USR "C"+j,q
 5090 NEXT j
 5100 DATA BIN 00111100,BIN 11111110,BIN 00111111,BIN 00011111,
      BIN 00001111,BIN 00111111,BIN 11111110,BIN 00111100
 5110 FOR j=0 TO 7
 5120 READ q
 5130 POKE USR "D"+j,q
 5140 NEXT j
 5150 DATA BIN 00111100,BIN 01111110,BIN 11111111,BIN 11111111,
      BIN 11110111,BIN 11100111,BIN 01000010,BIN 01000010
 5160 FOR j=0 TO 7
 5170 READ q
 5180 POKE USR "E"+j,q
 5190 NEXT j
 5200 DATA BIN 01000010,BIN 01000010,BIN 11100111,BIN 11101111,
      BIN 11111111,BIN 11111111,BIN 01111110,BIN 00111100
```

Teraz pridáme do našej hry trochu farieb a to tým, že vymenime riadky 1030 a 2010 za nasledujúce:

```
 1030 PRINT INK 2;AT dd,da;a$
 2010 PRINT INK 1;AT gd,ga;"G"
```

Dotman bude po tejto malej úprave červený a duch modrý.

Teraz doplnime ešte poslednú časť programu, ktorá obsahuje grafiku ducha a podprogramy pre zobrazovanie počtu bodov a konečné vyhodnotenie:

```
 1025 IF SCREEN$ (dd,da)="." THEN LET body=body+10
 1032 IF (ga=da AND gd=dd) OR body=1000 THEN GO TO 4000
 1985 PRINT AT 0,20;"BODY: ";body
 2012 IF (ga=da AND gd=dd) OR body=1000 THEN GO TO 4000
 4000 PRINT AT 0,20;"BODY: ";body
 4005 PRINT AT 16,6; INK 6; PAPER 2;"K O N I E C   H R Y"
 4010 FOR f=1 TO 30: BEEP 01,RND*60: NEXT f
 4020 IF body=1000 THEN PRINT AT 19,4; INK 2;"BLAHOZELAM, ZVITAZIL SI !"
 4030 IF body<1000 THEN PRINT AT 19,5; INK 1;"SMOLA, DUCH TA CHYTIL !"
 4040 PRINT AT 21,5;"CHCES HRU ZNOVA (A/N) ?"
 4050 IF INKEY$="" THEN GO TO 4050
 4060 IF INKEY$="a" THEN RUN
 6000 REM definovanie DUCHA (pismeno G v grafickom mode !!!)
 6010 FOR j=0 TO 7
 6020 READ q
 6030 POKE USR "G"+j,q
 6040 NEXT j
 6050 DATA BIN 00111000,BIN 01111100,BIN 11010110,BIN 11010110,
      BIN 11111110,BIN 11111110,BIN 10101010,BIN 10101010
```

Teraz si celú hru môžete nahrať na magnetofónovú pásku. Ak sa Vám zapáčila, môžete si ju rôzne upraviť, napríklad zväčšiť a zmeniť bludisko, doplniť viacej duchov, či iných strašidiel a podobne.

V Y M A Z A N I E Č A S T I O B R A Z O V K Y

Toto je veľmi prakticky podprogram, pomocou ktorého môžeme odmazať určitý počet riadkov obrazovky bez toho, aby sme museli použit prikaz CLS.

```
8010 INPUT "Kolko chcete mazat riadkov ?: ";c
8020 IF c=0 OR c>21 THEN GO TO 8010
8030 FOR f=21 TO 21-c STEP -1
8040 PRINT AT f,0;"                                "
8050 NEXT f
8060 PRINT AT f+1,0;
8070 RETURN
```

Na riadku 8010 si program vypýta počet riadkov, ktore majú byť zmazane. Mazanie sa prevádza smerom zdola hore, pričom sa začína na spodnom riadku.

Po zmazani nepotrebných riadkov program na riadku 8060 nastaví novú PRINT poziciu.

S C R O L L O V A N I E N A O B R A Z O V K E

V strojovom kode existujú praktické podprogramy, pomocou ktorých môžeme veľmi rýchlo posúvať celý obraz alebo jeho časť niektorým smerom.

Naše BASIC podprogramy sú sice oveľa pomalšie, ale inak s nimi môžeme dosiahnuť rovnaký efekt. Táto metóda je založená na tom, že celé obrazové pole sa uloži do reťazcového poľa (string array) a potom sa realizuje takzvaný scroll efekt. Tu sú štyri podprogramy s ktorými môžete presúvať kompletný obraz na obrazovke hore, dole, doľava a doprava.

Scrollovanie hore:

```
 5 REM scrollovanie hore
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 LET a$=a$(33 TO)+"                                "
50 GO TO 30
```

Scrollovanie dole:

```
 5 REM scrollovanie dole
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 LET a$="                                "+a$(TO 672)
50 GO TO 30
```

Scrollovanie doľava:

```
 5 REM scrollovanie dolava
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 FOR f=1 TO 673 STEP 32
```

```
50 LET a$(f TO f+31)=a$(f+1 TO f-31)+
   "                                "
60 NEXT f
70 GO TO 30
```

Scrollovanie doprava:

```
 5 REM scrollovanie doprava
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 FOR f=1 TO 673 STEP 32
50 LET a$(f TO f+31)=" "+a$(f TO f+30)
60 NEXT f
70 GO TO 30
```

P O N E C H A N I E R I A D K O V N A O B R A Z O V K E

Niekedy sa stane, že potrebujeme nechať odscrollovať iba určitý počet riadkov a niekoľko riadkov chceme nechať na obrazovke. Toto nám jednoducho zabezpeči nasledujúci podprogram. Do reťazca a$ zadajte aspoň 5 riadkov ľubovolného textu a do premennej 1 zadajte počet riadkov, ktoré majú ostať zachované na obrazovke od hora.

```
10 DIM a$(704)
20 INPUT a$
25 INPUT "Kolko riadkov nechat ?: ";1
30 PRINT AT 0,0;a$
40 LET a$(1*32+1 TO)=a$((1+1)*32+1 TO)
50 GO TO 30
```

A teraz skúsime scrollovanie opačným smerom. Do premennej 1 zadajte počet riadkov, ktoré majú ostať zachovane na obrazovke zdola.

```
10 DIM a$(704)
20 INPUT a$
25 INPUT 1
30 PRINT AT 0,0;a$
40 LET a$(TO 704-1*32)="                                "+
   a$-(TO 704-1*32)
50 GO TO 30
```

R O L O V A N I E P O O B R A Z O V K E

Ďalšou zaujímavou technikou je rolovanie dokola, pri ktorom všetko, čo na obrazovke zmizne na jednej strane, objavi sa ihneď na druhej strane.

Rolovanie hore:

```
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 LET a$=a$(33 TO)+a$(TO 32)
```

```
50 GO TO 30
```

Rolovanie dole:

```
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 LET a$=a$(673 TO)+a$(TO 672)
50 GO TO 30
```

Rolovanie doľava:

```
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 FOR f=1 TO 673 STEP 32
50 LET a$(f TO f+31)=a$(f+1 TO f+31)+a$(f)
60 NEXT f
70 GO TO 30
```

Rolovanie doprava:

```
10 DIM a$(704)
20 INPUT a$
30 PRINT AT 0,0;a$
40 FOR f=1 TO 673 STEP 32
50 LET a$(f TO f+31)=a$(f+31)+a$(f TO f+30)
60 NEXT f
70 GO TO 30
```

P O H Y B L I V Á G R A F I K A

Aby sa nám ľubovoľná grafika "pohybovala" po obrazovke, nepotrebujeme urobiť nič iné, iba nakresliť znak na určité miesto na obrazovke, potom ho vymazať a vzápati ho nakresliť na iné miesto.

Pozrime sa na jednoduchý príklad, ako to funguje:

```
10 LET x=0
20 PRINT AT 5,x;"#"
30 PRINT AT 5,x;" "
40 LET x=x+1
50 GO TO 20
```

Nie je to samozrejme žiadny úchvatný program. Znak "#" sa preženie po obrazovke z ľava do prava a keď premenná x presiahne pravý okraj obrazovky, program sa zastaví s chybovým hlásením, pretože jeden riadok má iba 32 PRINT pozícií (0-31).

Skúsime teda program trochu upraviť tak, aby sme dostali uspokojivejší výsledok. V prvom rade musíme zabrániť skončeniu programu pre presiahnutie pravého okraja obrazovky. Preto doplníme riadok:

```
45 IF x=31 THEN GO TO 10
```

Nasledujúcim krokom bude zamedzenie blikania kocky. Dosiahneme to predĺžením doby, po ktorú kocka zostava na obrazovke. Doplníme teda riadok:

```
25 FOR f=1 TO 10: NEXT f
```

A dostaneme tento výsledok:

```
10 LET x=0
20 PRINT AT 5,x;"#"
25 FOR f=1 TO 10: NEXT f
30 PRINT AT 5,x;" "
40 LET x=x+1
45 IF x=31 THEN GO TO 10
50 GO TO 20
```

Teraz to už vyzerá o niečo lepšie, ale s výsledkom ešte nemôžeme byť spokojný. Pretože v predošlej koncepcii sa už len ťažko dá niečo vylepšiť, skúsime napísať program trochu inak:

```
10 LET x=0
20 LET p=x
30 LET x=x+1
40 IF x>31 THEN LET x=0
50 PRINT AT 5,p;" ";AT 5,x;"#"
60 GO TO 20
```

Pozrime sa teraz trochu na štruktúru programu. Je celkom prehľadný:

Riadok 10 vynuluje premennú x, určujúcu horizontálnu pozíciu znaku.
Riadok 20 vloží hodnotu uloženú v premennej x do premennej p.
Riadok 30 zväčšuje premennú x o jednu, čím sa posúva pozícia znaku.
Riadok 40 zastaví úplne bezpečne znak na pravom okraji obrazovky.
Riadok 50 znak zobrazí a vymaže starú pozíciu.
Riadok 60 zabezpečí nekonečné uzavretie cyklu.

Môžeme však program skrátiť o jeden riadok tak, že zmeníme riadok 30:

```
30 LET x=x+1 AND x<31
```

Toto je celkom šikovný a obľúbený trik. Tento riadok nehovorí počítaču nič iné než: Ak je x menšie než 31, potom sa pripočíta k x jedna, ale ak nie je menšie ako 31, potom dosať do hodnoty x nulu. Teraz vymažeme riadok 40, ktorý je už zbytočný. Prečíslujeme čísla riadkov v 10-kovom kroku a dostaneme tento výsledok:

```
10 LET x=0
20 LET p=x
30 LET x=x+1 AND x<31
40 PRINT AT 5,p;" ";AT 5,x;"#"
50 GO TO 20
```

Teraz sme sa už dostali ku konečnému riešeniu. K nášmu uspokojeniu je priebeh dostatočne rýchly a ušetrili sme miesto v RAM.

Vratme sa teraz ešte k prikazu INKEY$, velmi cennemu pre pohyblivú grafiku. Iste si spomínate, že funkcia INKEY$ obdrží vždy znak klávesy, ktorá bola naposledy stlačená.

Pomocou príkazu INKEY$ získame účinný prostriedok pre manipuláciu s pohyblivou grafikou. Pre ovladanie rôznych postavičiek z hier si môžeme zvolit vždy tie klávesy, ktore su pre nas najvhodnejšie umiestnené.

Pre pohyb v štyroch smeroch môžeme napríklad používať klávesy 5-8 a to preto, lebo sú na nich zobrazene kurzorove šípky.

Skúsime si teraz napísať krátky program na pohyb jedneho znaku, ktorý budeme ovládať pomocou kurzorových kláves.

```
100 IF INKEY$="5" THEN LET x=x-1
110 IF INKEY$="8" THEN LET x=x+1
120 IF INKEY$="7" THEN LET y=y-1
130 IF INKEY$="6" THEN LET y=y+1
140 PRINT AT y,x;"*"
```

Skôr než program odštartujeme, musíme definovať premenné x a y, inak program nebude fungovať. Doplníme teda riadky:

```
10 LET x=0
20 LET y=0
30 LET a=x
40 LET b=y
```

Keď program spustíme, objaví sa v ľavom hornom rohu hviezda. Program bude vykonaný a zastaví sa na riadku 140. Tomu zabránime tak, že doplníme riadok 200.

```
200 GO TO 30
```

Nesmieme tiež zabudnúť zmazať starú pozíciu znaku. Najelegantnejšie to spravíme tak, že zmeníme riadok 140 tak, aby vyzeral takto:

```
140 IF a<>x OR b<>y THEN PRINT AT b,a;" ";AT y,x;"*"
```

Celý program bude teda po všetkých úpravách vyzerať takto:

```
 10 LET x=0
 20 LET y=0
 30 LET a=x
 40 LET b=y
100 IF INKEY$="5" THEN LET x=x-1
110 IF INKEY$="8" THEN LET x=x+1
120 IF INKEY$="7" THEN LET y=y-1
130 IF INKEY$="6" THEN LET y=y+1
140 IF a<>x OR b<>y THEN PRINT AT b,a;" ";AT y,x;'*"
200 GO TO 30
```

P R E T E K Á R

Teraz si v praktickom príklade ukážeme použitie scrollingu. Tentokrát to bude v hre "Pretekár". Scroll efekt tu dosiahneme jednoducho pomocou

príkazu POKE 23692,-1.

Program "Pretekár" simuluje jazdu závodným automobilom. Cesta je ohraničena znakom "|". Pretekárske auto môžete ovládať pomocou kláves:

z - vľavo m - vpravo

Skúste si program napísať do počítača a spustiť ho:

```
 10 REM PRETEKAR
 20 LET c=0
 30 LET t=0
 40 GO SUB 250
 50 LET a=10
 60 LET x=13
 70 LET y=12
 80 LET k=INT (RND*2)
 90 LET a=a-(k=1 AND a>1)+(k=0 AND a<24)
100 REM C na riadkoch 110, 200 a 270 je v grafickom mode !!!
110 PRINT AT y,x-1; INK 1;"C"
115 FOR f=1 TO 20: NEXT f: PRINT AT y,x-1;" "
120 PRINT AT 20,a;"|"; TAB a+6;"|"
130 PRINT
140 POKE 23692,-1: PRINT
150 PRINT INK 6; PAPER 2; AT 0,9;"STAV BODOV: ";t
160 IF SCREEN$ (y,x-1)="|" THEN GO TO 200
170 LET x=x-(INKEY$="z")+(INKEY$="m")
180 LET t=t+1
190 GO TO 80
200 PRINT AT y,x-1; FLASH 1; INK 2;"C"
205 FOR f=1 TO 20: BEEP .01,-f+10: NEXT f
210 PRINT AT 1,8; FLASH 1; BRIGHT 1;"HAVAROVAL SI !!!"
220 PRINT AT 21,5;"CHCES HRU ZNOVA (A/N) ?"
230 IF INKEY$="" THEN GO TO 230
240 IF INKEY$="a" THEN RUN
245 STOP
250 FOR j=0 TO 7
260 READ z
270 POKE USR "C"+j,z
280 NEXT j
290 RETURN
300 DATA BIN 10011001,BIN 11111111,BIN 10100101,BIN 00100100,
    BIN 00111100,BIN 10011001,BIN 11111111,BIN 10011001
```

Ak sa s programom vyhráte do sýtosti, skúste si ďalší prog am, ktorý taktiež používa pohyblivú grafiku.

H O R I A C I Š Í P

Táto hra Vás privedie do stredoveku. Ste lukostrelcom na obliehanom hrade, ktorý sa nepriatelia snažia zapáliť pomocou horiacich šípov. Jedine Vy tomu môžete zabrániť a to tak, že horiace šípy zostrelíte. Ale pozor, stačí aby Vám unikli 3 zápalné šípy a celý hrad je v plameňoch!

Skúste si teda napísať nasledujúci program a vyskúšať ho. Horiaci šíp môžete zostreliť len vtedy, keď letí priamo nad Vašim stanoviskom na

cimburi hradu. Strielať môžete ľubovolnou klavesou.

```
  1 REM HORIACI SIP
  5 REM Pismena A,B,C,D su v grafickom mode !!!
  8 GO SUB 200
 10 PRINT AT 15,14; INK 1;"C";AT 4,0; INK 2;"**********";
    " ZASAHOV:0  ";"**********";AT 16,0; INK 3;"BBBBBBBBBBBBBBBBBBBBB
    BBBBBBBBBB"
 15 FOR f=17 TO 21: PRINT AT f,0; INK 3;"AAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    AAA":NEXT f
 20 LET h=0
 30 LET m=0
 40 LET y=RND*6+7
 50 FOR x=0 TO 19
 60 IF m=3 THEN GO TO 1000
 70 REM Pismeno D je v grafickom mode !!!
 80 PRINT AT y,x; INK 2;"D": FOR f=1 TO 20: NEXT f: PRINT AT y,x;" "
 90 IF INKEY$<>"" AND x=14 THEN GO TO 140
100 NEXT x
110 LET m=m+1
130 GO TO 40
140 LET h=h+1
150 FOR f=1 TO 30
160 PRINT AT 6,12; INK 6; PAPER 2; FLASH 1;"ZASAH !": BEEP .01,39-3*f
170 NEXT f: PRINT AT 6,12;"       "
180 PRINT AT 4,19; INK 2;h
190 GO TO 40
200 FOR j=0 TO 7
210 READ q
220 POKE USR "A"+j,q
230 NEXT j
240 FOR j=0 TO 7
250 READ q
260 POKE USR "B"+j,q
270 NEXT j
280 FOR j=0 TO 7
290 READ q
300 POKE USR "C"+j,q
310 NEXT j
320 FOR j=0 TO 7
330 READ q
340 POKE USR "D"+j,q
350 NEXT j
400 RETURN
500 DATA BIN 00100010,BIN 11111111,BIN 10001000,BIN 10001000,
    BIN 10001000,BIN 11111111,BIN 00100010,BIN 00100010
510 DATA BIN 00000000,BIN 11000011,BIN 11000011,BIN 11000011,
    BIN 11111111,BIN 11111111,BIN 00100010,BIN 00100010
520 DATA BIN 11100000,BIN 11100000,BIN 00110000,BIN 00011000,
    BIN 00001100,BIN 00000110,BIN 00000011,BIN 11111111
530 DATA BIN 00001000,BIN 10000100,BIN 01000010,BIN 00111111,
    BIN 01000010,BIN 10000100,BIN 00001000,BIN 00000000
1000 PRINT AT 6,10; PAPER 2; INK 6; FLASH 1;"HRAD HORI !!"
1005 FOR g=1 TO 70
1010 BEEP .01,g/2: BEEP .01,40-g
```

```
1020 NEXT g
1030 RUN
```

A teraz si o programe niečo povieme:

Na riadku 10 a 15 sa vykreslí cimburie hradu a jeho hradby. Premenná h počíta počet zostrelených šípov, premenna m počet zapalných šípov, ktore sme nezasiahli. Na začiatku su obe premenne nastavené na hodnotu 0.

Na riadku 40 sa pomocou premennej y vypočíta náhodná výška preletu šípu medzi hodnotou 7 a 13.

Na riadku 50 sa začína hlavná slučka programu, v ktorej sa zobrazuje prelietavajúci šíp a zároveň sa zisťuje či bol zasiahnutý.

Na riadku 60 sa zisti, či už hrad zasiahli tri šípy. Ak áno, skoči sa na riadok 1000.

Na riadku 80 sa zobrazi šíp a zároveň sa pomocou medzery vymaže stará pozícia.

Na riadku 90 program zisťuje, či bol šíp zasiahnutý. Ak áno, skoči sa na podprogram na riadku 140. Tu sa pričíta jedna k premennej h, kontrolujúcej počet zásahov a z riadku 130 sa skáče znova na riadok 40.

H A N D R Á R

V ďalšej hre sa stávate pouličným zberačom handier. Mali ste už handier plný vozík, keď tu zrazu zafúkal vietor a všetky Vám ich rozfúkal po ulici. Musíte handry ponapichovať na špeciálnu palicu, ale musíte sa poponáhľať, inak Vám ich vietor odfúkne nenávratne preč.

```
 10 REM HANDRAR
 20 GO SUB 1000
 30 FOR g=1 TO 50
 35 LET m=RND*16+3: LET n=RND*26+4
 40 IF ATTR (m,n)=49 THEN GO TO 35
 45 REM Pismeno A je v grafickom mode !!!
 50 PRINT AT m,n; INK 1;"A"
 60 NEXT g
 70 LET x=0
 80 LET y=0
 85 FOR g=1 TO 500
 90 LET a=x
100 LET b=y
110 LET y=y-(INKEY$="q" AND y>1)+(INKEY$="a" AND y<20)
120 LET x=x-(INKEY$="o" AND x>1)+(INKEY$="p" AND x<31)
130 IF ATTR (y,x)=49 THEN LET body=body+1: PRINT AT 0,5; INK 1;
    "BODY: ";body;AT 0,20; INK 2;"CAS: ";500-g;"  "
140 IF ATTR (y,x)=49 THEN BEEP .01,40
150 REM Pismeno B je v grafickom mode !!!
160 PRINT AT b,a;" ";AT y,x; INK 2;"B"
165 IF body=50 THEN PRINT AT 10,10; INK 1; FLASH 1;"STIHOL SI TO !":
    FOR f=1 TO 40: BEEP .01,f: NEXT f: RUN
170 NEXT g
180 PRINT AT 0,5; INK 1;"BODY: ";body;AT 0,20; INK 2;"CAS: ";"0  "
190 PRINT AT 10,10; INK 2; FLASH 1;"NESTIHOL SI !"
200 FOR f=1 TO 20: BEEP .1,RND*50: NEXT f
210 RUN
```

```
1000 FOR a=0 TO 7
1010 READ q
1020 POKE USR "A"+a,q
1030 NEXT a
1040 DATA BIN 10100101,BIN 01011010,BIN 10100101,BIN 01011010,
     BIN 01011010,BIN 10100101,BIN 01011010,BIN 10100101
1050 FOR a=0 TO 7
1060 READ q
1070 POKE USR "B"+a,q
1080 NEXT a
1090 DATA BIN 10000001,BIN 01000010,BIN 00100100,BIN 00011000,
     BIN 00011000,BIN 00100100,BIN 01000010,BIN 10000001
1100 BORDER 2: PAPER 6: CLS
1110 LET body=0
1120 RETURN
```

Teraz sme Vám ešte dlžní vysvetlenie, ako program rozoznáva, či je handra napichnutá, alebo nie. Dá sa to zistiť celkom jednoducho pomocou funkcie ATTR (y,x). Výsledkom tejto funkcie je atribút štvorca na suradniciach y,x - teda kombinácia prvkov podielajúcich sa na farbe: PAPER, INK, BRIGHT a FLASH. Takýchto kombinácii môže byť presne 256, teda čísla 0-255. Pomocou funkcie ATTR (y,x) zistíme teda veľmi ľahko, či je na niektorej pozícii "handra", teda červený znak na žltom pozadí, majúci atribútove číslo 49.

Tento šikovný trik si dobre zapamätajte, isto sa Vám bude hodiť aj v iných programoch.

M A N I P U L Á C I A S R E Ť A Z C A M I

Aj keď nie je často používaná, existuje ešte jedna možnosť, ako prevádzať prácu s pohyblivou grafikou. V takomto prípade nepracujeme s jedným znakom, ale s celým reťazcom. Potom môžeme využívať všetky funkcie pracujúce s časťami reťazca.

V praxi to fuguje tak, že si zadefinujeme reťazce dostatočne veľké, aby mohli vyplniť celú jednu časť obrazovky. Potom už len tieto reťazce striedame, čím vyvolávame dojem animácie.

Teraz si ukážeme dva spôsoby animácie manipuláciou s reťazcami.

1. Zobrazovanie rozmanitých častí reťazcov v určenom poradí a slede.

```
10 DIM a$(32)
20 LET a$(1)="+"
30 FOR a=32 TO 1 STEP -1
40 PRINT AT 20,0;a$(a TO)+a$(TO a-1)
50 NEXT a
60 GO TO 30
```

Ak ste spustili predošlý program, máte možnosť vidieť animáciu pomocou zobrazovania rôznych častí reťazca. Napíšte teraz do počítača nasledujúci program, spustite ho a uvidíte niečo zaujímavejšie.

```
20 INPUT "Zadajte text: ";a$
30 IF a$="" THEN GO TO 20
```

```
 40 LET s=50
 50 FOR b=1 TO LEN a$+33
 70 PRINT AT 5,0;("                                    "+a$+
    "                                  ")(b TO b+31)
 80 LET s=s-(5 AND INKEY$="f" AND s>1)+(5 AND INKEY$="d")
 90 IF INKEY$ ="a" THEN GO SUB 140
100 FOR a= 1 TO s
110 NEXT a
120 NEXT b
130 GO TO 50
140 FOR a=1 TO 200
150 NEXT a
160 RETURN
```

Pri vkladaní programu do počítača zapisujte pozorne najmä riadok 70. Obidva reťazce naplnené medzerami musia mať presne 32 znakov. Ak spustite program, počítač si od Vás vypýta text. Zadajte ho a odošlite klávesou ENTER. Ak ste program prepísali správne, text, ktorý ste zadali sa objaví na pravej strane a začne sa pomaly scrollovať doľava. Komfortnosť tohoto programu však ešte len oceníte. Pomocou klávesy "f" môžete totiž plynule zvyšovať rýchlosť posúvaného textu a klávesou "d" ju znižovať. Ba čo viac, keď sa Vám zdá, že sa text pohybuje na Váš vkus aj tak príliš rýchlo, pomocou klávesy "a" ho môžete na chvíľu úplne zastaviť.

Program sa stane úplne dokonalým, keď doplníte nasledujúci riadok:

```
85 IF INKEY$="r" THEN RUN
```

Teraz môžete počas chodu programu kedykoľvek stlačiť klávesu "r", čím sa program znova spustí od začiatku a Vy môžete zadať nový text.

2. Posúvanie prvkov poľa.

```
10 DIM a$(32)
20 FOR a=1 TO 32
30 LET a$(a)="+"
40 PRINT AT 20,0;a$
50 LET a$(a)=" "
60 NEXT a
70 GO TO 20
```

Toto je druhý spôsob simulovania animácie pomocou manipulácie s reťazcami.

Teraz si ukážeme animáciu pomocou manipulácie s reťazcami na BASICovej verzii známej počítačovej hry "Space Invaders" (Invázia z vesmíru). Táto hra známa najmä z herných automatov je síce v BASICu trochu menej kvalitná ako v strojovom kóde, ale na naše demonštračné účely je viac než vyhovujúca.

Tak a teraz sa už sústreďme na našu hru: Zem je ohrozovaná inváziou votrelcov z kozmu. Ochrana zeme je len vo Vašich rukách. Musíte zneškodniť všetkých votrelcov pomocou laserového dela, ktoré máte k dispozícii.

Aby ste sa stali absolútnym víťazom, musíte odraziť sedem útočných vĺn votrelcov. Laserové delo ovládate pomocou kláves:

o - vľavo p - vpravo z - streľba

```
  1 REM SPACE INVADERS
  4 BORDER 0: PAPER 0: CLS
  5 GO SUB 290
 10 DIM a$(32)
 20 DIM b$(32)
 30 FOR d=1 TO 7
 35 REM Pismeno A je v grafickom mode !!!
 40 LET a$="   A   A   A   A   A   A   A   "
 50 LET x=15
 60 LET c=x
 70 PRINT AT 5,12; INK d; PAPER 9;d;". UTOK"
 80 LET c=0
 90 FOR b=d+9 TO 19 STEP 2
100 FOR A=0 TO 31
110 LET x=x+(INKEY$="p" AND x<31)-(INKEY$="o" AND x>0)
115 REM Pismeno B je v grafickom mode !!!
120 PRINT AT b,0; INK RND*7;a$;AT 20,c;" " AND c<>x;AT 20,x; INK 5;"B"
130 IF a$=b$ THEN GO TO 200+(40 AND d=7)
140 LET c=x
150 IF INKEY$="z" THEN BEEP.01,20: IF a$(x+1)="A" THEN LET a$(x+1)=" "
    : PRINT AT b,0;INK 6;a$: BEEP .01,50: LET sc=sc+1: PRINT AT 0,20;
    INK 6;"SKORE: ";sc*10
160 NEXT a
165 LET a$=a$(4 TO)+a$(TO 3)
170 PRINT AT b,0;b$
180 NEXT b
190 GO TO 240
200 CLS: NEXT d
240 IF d<>7 OR sc<>49 THEN PRINT PAPER 2; INK 6; FLASH 1; AT 10,10;
    "PREHRAL SI !";AT 12,5; FLASH 0;"VOTRELCI OBSADILI ZEM.":
    GO TO 260
245 PRINT PAPER 6; INK 2; FLASH 1;AT 10,10;"VYHRAL SI !!"
250 PRINT PAPER 6; INK 2;AT 12,3;"VSETCI VOTRELCI SU ZNICENI"
260 FOR f=1 TO 40: BEEP .01,f: BEEP .01,-f: NEXT f
270 IF INKEY$="" THEN GO TO 270
275 RUN
280 REM Grafika votrelcov
290 FOR j=0 TO 7
300 READ q
310 POKE USR "A"+j,q
320 NEXT j
330 DATA BIN 00111110,BIN 00101010,BIN 00111110,BIN 00011100,
    BIN 00001000,BIN 01110111,BIN 01000001,BIN 01000001
340 REM Grafika laseroveho dela
350 FOR j=0 TO 7
360 READ q
370 POKE USR "B"+j,q
380 NEXT J
390 DATA BIN 00011000,BIN 00011000,BIN 00011000,BIN 00111100,
    BIN 01100110,BIN 11100111,BIN 11111111,BIN 10100101
400 LET sc=0
410 RETURN
```

Pre našu ukážku sú najdôležitejšie riadky 10 a 20, kde su dimenzované reťazce a$ a b$. Do reťazca a$ vložíme 7 x pismeno "A" v grafickom móde.

Toto pismeno bude neskôr predefinovane na grafiku votrelcov. Je dolezite, aby retazec a$ mal presne 32 znakov. Medzi jednotlivých votrelcov doporučujeme vložiť 3 prazdne znaky.

Na riadku 130 sa kontroluje, či sa retazec a$ rovná retazcu b$ (ktorý obsahuje iba prazdne znaky). Ak je to pravda, program bezpecne zistil, že jedna útocná vlna votrelcov bola zničena.

Dalej je pre nás zaujimavý riadok 150. Tu sa simuluje streľba z laserového dela a zároveň sa zisti, či bol po vystrele votrelec zničeny (musi byť v zakryte s laserovym delom).

Ak si chceme bezpečne uschovať v pamati kompletný obsah celej obrazovky (22 x 32 znakov) pomocou retazca, potom existuju opať dve metody ako to previesť:

1. Ako dvojrozmerné pole, vymedzené pomocou prikazu DIM a$(22,32).

Jednotlivé prvky tohoto poľa (teda vlastne riadky) môžeme potom zobrazovať na obrazovku pomocou prikazu PRINT AT. Výhoda tohoto spôsobu spočiva v tom, že môžeme manipulovať samostatne s každým riadkom obrazovky, čo ocenime najma pri hrách, ktoré využivaju šachovnicu (dama, šach). Nevýhoda spočiva v tom, že je nutné každý prvok poľa zobrazovať na obrazovku zvlášť. Napriklad:

```
PRINT AT 0,0;a$(1)
PRINT AT 1,0;a$(2)
PRINT AT 2,0;a$(3)
..... .. .........
..... .. .........
..... .. .........
PRINT AT 19,0;a$(20)
PRINT AT 20,0;a$(21)
PRINT AT 21,0;a$(22)
```

2. Ako jednorozmerné pole, vymedzené pomocou prikazu DIM a$(704).

Takéto pole potom vytlačíme na obrazovku veľmi jednoducho pomocou prikazu: PRINT AT 0,0;a$. V takomto veľkom poli sa však budeme veľmi ťažko orientovať. Táto nevýhoda sa dá vykompenzovať tým, že si zhotovíme tabuľku, kde budú jednotlivým prvkom obrazovky priradené konkrétne prvky poľa. Potom už bude hráčkou s našim poľom manipulovať. Tu je časť takejto tabuľky:

X	Y	0	1	2	..	29	30	31
0		a$(1)	a$(2)	a$(3)	.....	a$(30)	a$(31)	a$(32)
1		a$(33)	a$(34)	a$(35)	.....	a$(62)	a$(63)	a$(64)
2		a$(65)	a$(66)	a$(67)	.....	a$(94)	a$(95)	a$(96)
3		a$(97)	a$(98)	a$(99)	.....	a$(126)	a$(127)	a$(128)

Táto časť tabuľky znázorňuje vrchnú časť obrazovky. Ak chceme na niektorej pozicii obrazovky manipulovať so znakom urobíme to jednoducho

tak, že do našej funkcie dosadíme premenné x,y, alebo a$(z). Napríklad:

```
PRINT AT 3,2;a$(99)
LET a$(99)="*"
.....................
```

Tento spôsob nám umožňuje pomerne jednoduchý pohyb znaku na obrazovke do ôsmich smerov. Prehľadne to uvidíte v nasledujúcom diagrame:

8	1	2
7	X	3
6	5	4

Smer pohybu	Zmena k pozícii X
1	-32
2	-31
3	+ 1
4	+33
5	+32
6	+31
7	- 1
8	-33

Napríklad ak chceme znak na pozícii x premiestniť na pozíciu 2, stačí zadať túto dvojicu príkazov:

```
LET a$(x-31)=a$(x): LET a$(x)=" "
```

I N K E Y $ A C A P S S H I F T

Doteraz sme si ukazovali simuláciu pohybu pomocou reťazca len v horizontálnom a vertikálnom smere. Ďalší krátky podprogram nám umožní pohyb aj v diagonálnom (šikmom) smere.

```
10 DIM a$(704)
20 LET a=INT(RND*704)+1
30 LET a$(a)=" "
40 LET a=a-(32 AND INKEY$="7")-(31 AND INKEY$=CHR$ 11)+
   (1 AND INKEY$="8")+(33 AND INKEY$=CHR$ 9)+
   (32 AND INKEY$="6")+(31 AND INKEY$=CHR$ 10)-
   (1 AND INKEY$="5")-(33 AND INKEY$=CHR$ 8)
50 LET a$(a)="#"
60 PRINT AT 0,0;a$
70 GO TO 30
```

Ak program prepíšete do počítača a spustíte, na obrazovke sa Vám zobrazí znak "♯". S týmto znakom môžete pohybovať pomocou kurzorových šípiek (klávesy 5,6,7,8). Ak stlačíte jednu z týchto kláves spolu s klávesou CAPS SHIFT, dostanete diagonálny pohyb odklonený od stlačenej kurzorovej šípky o 45 stupňov v smere hodinových ručičiek.

Funkcie CHR$ 8 až CHR$ 11 obsahujú kódy kurzorových kláves stlačených spolu s klávesou CAPS SHIFT.

Symboly, ktorými sa prevadzaju v BASICu matematicke operacie, su Vam už iste veľmi dobre zname: nasobenie (*), delenie (/), odčitanie (-), sčitanie (+) a umocňovanie (^). Pre tieto operacie je zaistený prisny prioritný poriadok pri ich vyhodnocovani v aritmetickom vyraze.

Táto priorita je zaistena poradím, v ktorom su jednotlive operacie zhodnotene. Závisí to aj od toho, v akom poradí su vo vyraze napisané. Napríklad výraz v zátvorkach da pri vypočte častokrat celkom iný výsledok než bez zatvoriek. Pri nepritomnosti niektorej časti zatvorky je hlásená chyba syntaxe. Cely systém priorit funguje jednoducho tak, že každá operacia má pevne určené prioritne číslo medzi 1-16. Čím je vyššie toto číslo, tým je vyššia priorita, t.j., tým skór sa tato operácia vo výraze realizuje.

Ak nastane prípad, že sú vo výraze dve operácie s rovnakou prioritou, bude ich počítač vyhodnocovať v poradí z ľava do prava.

Operátor	Význam	Priorita
()	zátvorky	16
	príznaky a slicing	12
ABS,INT,SGN...	všetky funkcie okrem NOT	11
^	mocnenie	10
-	záporné znamienko	9
*	násobenie	8
/	delenie	8
+	sčítanie	6
-	odčítanie	6
=,<,>,<=,>=,<>	relačné operátory	5
NOT	negácia	4
AND	logický súčin	3
OR	logický súčet	2

Všimnite si prosím, že počítač všetky čísla pokladá vždy za kladné pokiaľ nie sú prostredníctvom mínusového znamienka označené ako záporné. Rovnako tak pokladá číslo bez desatinej bodky za celé číslo. Pre veľmi malé a veľmi veľké čísla možno používať takzvaný vedecký zápis (s E), ako sme si o tom už povedali v kapitole o premenných.

BASIC Vášho počítača pracuje aj v dosť zložitých matematických výpočtoch pomerne rýchlo, ako to uvidíte v nasledujúcich programoch. Prvý program Vám spracuje aritmetickú radu. Zadáte vstupné číslo, potom krok a nakoniec počet krokov. Počítač vypíše kompletnú aritmetickú radu.

```
10 REM ARITMETICKA RADA
20 PRINT AT 0,1;"SPRACOVANIE ARITMETICKEJ RADY"
30 PRINT
40 INPUT "Zadajte prve cislo: ";prve
50 PRINT " Začiatok:  ";prve
60 INPUT "Zadajte krok rady: ";krok
70 PRINT " Krok rady: ";krok
```

```
 80 INPUT "Zadajte pocet krokov: ";pocet
 90 CLS
100 LET pocet=INT (pocet+.5)
110 POKE 23692,-1
120 LET q=prve: LET citac=0: LET r=0
130 FOR f=1 TO pocet STEP krok
140 LET r=r+1
150 IF r>1 THEN LET q=q*2
160 LET citac=citac+q
170 PRINT TAB 2;r;TAB 10; INK 1;q
180 NEXT f
190 PRINT
200 PRINT TAB 2; INK 2;"Sucet je: ";citac
```

P R V O Č Í S L A

Tento krátky program Vám vypočíta prvočísla:

```
10 REM PRVOCISLA
20 INPUT "Po ktore cislo: ";b
30 FOR a=1 TO b: LET i=0
40 FOR f=2 TO SQR b: IF b/f=INT (b/f) THEN LET i=i+1: NEXT f
50 IF i=0 THEN PRINT TAB (8-LEN (STR$ a));a
60 POKE 23692,-1: NEXT a
```

Š T A T I S T I K A

Nasledujúci program v sebe zahŕňa radu štatistických podprogramov. Jednotlivé podprogramy začínajú na riadkoch 1000, 2000, 3000 a 4000.

Tieto štyri podprogramy nám umožnia tieto výpočty:

1. Aritmetický priemer
2. Geometrický priemer
3. Harmonický priemer
4. Faktoriál

Podprogram na riadku 9000 prezentuje menu možnosti. Pozrite sa na riadok 9990 na príkaz skoku GO TO a * 1000. Je to prakticky zjednodušenie vyjadrenia nasledujúcich štyroch riadkov:

```
9990 IF a=1 THEN GO TO 1000
9991 IF a=2 THEN GO TO 2000
9992 IF a=3 THEN GO TO 3000
9993 IF a=4 THEN GO TO 4000
```

Túto techniku môžete využiť vždy v tom prípade, keď má Váš program viac možností výberu na vytvorenie menu.

```
  10 GO TO 9000
 900 REM ------------------------------------------------------------
1000 REM ARITMETICKY PRIEMER
1010 PRINT "Zadajte hodnoty (koniec = ENTER)": PRINT
```

```
1020 INPUT q$: IF q$="" THEN GO TO 1070
1030 POKE 23692,-1: PRINT q$
1040 LET total=total+VAL q$
1050 LET citac=citac+1
1060 GO TO 1020
1070 PRINT '"Aritmeticky priemer: ";total/citac
1080 PAUSE 0: GO TO 9000
1090 REM ------------------------------------------------------
2000 REM GEOMETRICKY PRIEMER
2010 PRINT "Zadajte hodnoty (koniec = ENTER)": PRINT: LET total=1
2020 INPUT q$: IF q$="" THEN GO TO 2070
2030 LET citac=citac+1
2040 LET total=total+VAL q$
2050 POKE 23692,-1: PRINT q$
2060 GO TO 2020
2070 PRINT '"Geometricky priemer: ";total^(1/citac)
2080 PAUSE 0: GO TO 9000
2090 REM ------------------------------------------------------
3000 REM HARMONICKY PRIEMER
3010 PRINT "Zadajte hodnoty (koniec = ENTER)": PRINT
3020 INPUT q$: IF q$="" THEN GO TO 3070
3030 LET citac=citac+1
3040 LET total=total+(1/VAL q$)
3050 POKE 23692,-1: PRINT q$
3060 GO TO 3020
3070 PRINT '"Harmonicky priemer: ";total/citac
3080 PAUSE 0: GO TO 9000
3090 REM ------------------------------------------------------
4000 REM FAKTORIAL
4010 PRINT "Zadajte hodnoty, len cele kladnecisla do 33
     (koniec = ENTER)": PRINT
4020 INPUT num: IF num>=34 THEN GO TO 4020
4030 LET num=INT (num): LET a=1
4040 FOR b=1 TO num
4050 LET a=a*b
4060 NEXT b
4070 PRINT '"Faktorial z ";num;" je: ";a
4080 PAUSE 0: GO TO 9000
4090 REM ------------------------------------------------------
9000 CLS: FOR g=1 TO 5
9010 PRINT
9020 NEXT g
9030 PRINT INK 2;"VYBERTE SI Z TOHOTO MENU:"
9040 PRINT
9050 PRINT '"1 - Aritmeticky priemer"
9060 PRINT '"2 - Geometricky priemer"
9070 PRINT '"3 - Harmonicky priemer"
9080 PRINT '"4 - Faktorial"
9090 PRINT '"5 - Koniec programu"
9100 LET a$=INKEY$
9110 IF a$<"1" OR a$>"5" THEN GO TO 9100
9120 LET a=VAL a$
9130 IF a=5 THEN STOP
9140 LET total=0
9150 LET citac=0
```

```
9160 REM ------------------------------------------------------------
9990 CLS: GO TO a*1000
```

Tymto nie su všetky matematicke možnosti využitia Spectra vyčerpane. Nasledujuci program vypočita strednu hodnotu, štandartnu odchylku a štandartnu chybu a odchylku z maximalne 20 hodnôt. Ďalej vypočita špecialny T-TEST.

```
  10 REM T - TEST
  20 DIM x(240)
  22 DIM n(20)
  24 DIM m(20)
  26 DIM s(20)
  28 DIM e(20)
  30 DIM v(20)
  32 DIM g(20)
 110 PRINT "Tento program pocita strednu"'"odchylku, štandartnu
     odchylku"'"a štandartnu chybu."
 120 PRINT '"Potom sa vypocita T-TEST, ktory"'"urci, ci medzi dvoma
     hodnotami"'"je znacitelna diferencia."
 130 PRINT
 140 PRINT "Vystup zobrazit na monitor alebotlaciaren (m/t) ?"
 149 INPUT w$
 150 CLS
 160 LET g=0
 161 LET q=0
 162 LET s-0
 170 PRINT "Kolko jednotiek ?"
 180 INPUT m
 200 LET g=g+1
 205 CLS
 220 PRINT "Postupna jednotka cislo ";g
 245 PRINT '"Kolko hodnot ? (min.2)"
 250 INPUT n
 252 PRINT
 255 LET n(g)=n
 260 LET s1=0
 262 LET s2=0
 270 FOR i=1 TO n
 271 CLS
 280 PRINT "Zadajte hodnotu cislo ";i'
 290 INPUT x
 295 CLS
 300 LET s1=s1+x
 310 LET s2=s2+x*x
 320 LET x(i)=x
 330 NEXT i
 340 LET m(g)=s1/n
 345 LET v(g)=((s2-s1*s1/n)/(n-1))
 350 LET s(g)=SQR v(g)
 360 LET e(g)=s(g)/SQR (n)
 380 PRINT "Skontrolovat hodnoty ? (a/n)"
 390 INPUT q$
 395 CLS
 400 IF q$="n" THEN GO TO 550
```

```
405 PRINT "Postupna jednotka cislo ";q
410 LET q=q+1
415 LET r=2
417 LET k=0
420 FOR i=1 TO n
425 PRINT AT r,k;i;".hodnota: ";x(i)
430 IF i/20=INT (i/20) THEN LET k=k+10
431 IF k>21 AND w$="t" THEN COPY
432 IF k>21 THEN GO SUB 3000
435 IF k>21 THEN LET k=k-30
460 IF i/20=INT (i/20) THEN LET r=r-20
465 LET r=r+1
480 NEXT i
490 IF w$="t" THEN COPY
500 GO SUB 3000
550 IF q<m THEN GO TO 200
740 FOR i=1 TO m
750 PRINT "Postupna jednotka cislo ";i
752 PRINT
755 PRINT "Pocet hodnot: ";n(i)
758 PRINT
760 PRINT "Stredna odchylka: ";m(i)
762 PRINT
765 PRINT "Standartna odchylka: ";s(i)
767 PRINT
770 PRINT "Standartna chyba: ";e(i)
772 PRINT
775 PRINT "Priemer: ";v(i)
776 IF w$="t" THEN COPY
777 GO SUB 3000
780 NEXT i
790 PRINT
800 PRINT "Prajete si T-TEST ? (a/n)"
810 INPUT q$
815 CLS
820 IF q$="n" THEN GO TO 1200
830 PRINT "T-TEST medzi dvoma jednotkami"
835 PRINT
840 PRINT "Zadajte cislo 1. jednotky"
845 INPUT s1
847 PRINT '"Jednotka 1. = ";s1
849 PRINT
850 PRINT "Zadajte cislo 2. jednotky:"
852 INPUT s2
854 PRINT '"Jednotka 2. = ";s2
890 LET a=n(s1)+n(s2)
900 LET b=n(s1)*n(s2)
910 LET d=a-2
920 LET t=ABS (m(s1)-m(s2))/SQR (((n(s1)-1)*s/s(1)^2+(n(s2)-1)*
    s(s2)^2)*a/(b*d))
930 GO SUB 2000
935 CLS
940 PRINT "T-TEST pre jednotky c.";s1;" a c.";s2
945 PRINT ''
950 PRINT "T  = ";t
```

```
 955 PRINT
 960 PRINT "df = ";d
 965 PRINT
 970 PRINT "p  = ;p
 975 PRINT
 980 IF p>.001 THEN GO TO 990
 982 PRINT "p  < 0.001"
 984 GO TO 1050
 990 IF p>.01 THEN GO TO 1000
 992 PRINT "p  < 0.01"
 994 GO TO 1050
1000 IF p>.05 THEN GO TO 1010
1002 PRINT "p  < 0.05"
1004 GO TO 1050
1010 PRINT "p  > .05 NS"
1050 IF w$="t" THEN COPY
1060 PRINT '''"T-TEST u dalsich jednotiek (a/n)"
1070 INPUT q$
1075 CLS
1080 IF q$="n" THEN GO TO 1200
1090 GO TO 840
1200 PRINT "Analyzovat dalsie jednotky (a/n)"
1210 INPUT q$
1215 CLS
1220 IF q$="a" THEN GO TO 110
1240 PRINT "------------ KONIEC -------------"
1245 GO TO 9999
2000 REM vypocet hodnot T,p,df
2020 LET p=1
2025 LET m=1
2030 IF t=0 THEN GO TO 2210
2040 LET u=t*t
2050 IF u<1 THEN GO TO 2100
2060 LET i=m
2070 LET j=d
2080 LET t1=u
2090 GO TO 2130
2100 LET i=d
2110 LET j=m
2120 LET t1=1/u
2130 LET a1=2/9/i
2140 LET b1-2/9/j
2150 LET z=ABS ((1-b1)*t1^(1/3)-1+a1)/SQR (b1*t1^(2/3)+a1)
2160 IF j>3 THEN GO TO 2180
2170 LET z=z*(1+.08*z^4/j^3)
2180 LET p=.5/(1+z*(.196854+z*(.115194+z*(.000344+z*.019527))))^4
2190 IF u>1 THEN GO TO 2210
2200 LET p=1-p
2210 RETURN
3000 PRINT AT 21,2;"Stlacte lubovolnu klavesu !"
3004 IF INKEY$ ="" THEN GO TO 3004
3007 CLS
3010 RETURN
9999 STOP
```

Z A J A C E A L I Š K Y

Posledný program tejto state simuluje životný cyklus dvoch na seba závislých živočíchov (napríklad líšiek a zajacov). Líšky sa živia zajacmi. Tým ich decimuju a keď počet zajacov klesne pod istú hranicu, začnú líšky vymierať, lebo sa nemajú čím živiť. Keď už je málo líšiek, zajace sa premnožia, pretože ich nikdo neohrozuje. Vtedy majú líšky opäť dostatok potravy, začnú sa premnožovať a hubiť čoraz viac zajacov.

Tento nekonečný prírodný cyklus simuluje nasledujúci program. Relácia medzi oboma živočíchmi je vyjadrená diferenciálnou rovnicou. Zadáme počiatočnú populáciu ako číslo 1 až 5 (možno zadávať aj desatinné čísla pretože počet sa rata v desaťtisícoch) a uvidíme celý fascinujúci vývoj prehľadne zobrazený na grafe.

```
 10 REM ZAJACE A LISKY
 20 PRINT "Kolko zije lisiek ? (1-5)"
 30 INPUT x: IF x<1 OR x>5 THEN GO TO 30
 40 PRINT "Kolko zije zajacov ? (1-5)"
 50 INPUT y: IF y<1 OR y>5 THEN GO TO 50
 55 LET x=x+RND: CLS
 60 FOR z=1 TO 20
 70 FOR t=1 TO 7 STEP .5
 80 PRINT AT 1,1; INK 2;"Lisky:  ";INT (x*10000);"    "
 90 PRINT AT 2,1; INK 1;"Zajace: ";INT (y*10000);"    "
100 LET x=x+(4*x-2*x*y)*.01
110 LET y=y+(x*y-3*y)*.01
120 PLOT 20*x,20*y
130 NEXT t
135 BEEP .05,z+t
140 NEXT z
150 LET x=RND*6
160 LET y=RND*6
180 GO TO 60
```

F U N K C I E

Spectrum verzia jazyka BASIC obsahuje, práve tak ako iné BASIC dialekty radu matematických funkcií, ktoré sú veľmi osožné a ktoré môžeme použiť ako v programovaní, tak i pre priamy výpočet.

Všeobecné funkcie:

ABS - Mení argument na kladné číslo.

```
ABS -3.2 = 3.2
ABS 1000 = 1000
```

INT - Vytvorí z čísla integer, teda celé číslo tak, že INT x je najväčšie celé číslo, ktoré je menšie alebo rovné ako zadané číslo. Pozor na záporné čísla!

```
INT 3.9 = 3
INT -3.9 = -4
```

Ak chceme používať klasické zaokrúhľovanie, teda do hodnoty 0.4 zaokrúhľovať dole a od 0.5 hore, stačí použiť vzorec:

```
INT (x+.5)
```

K funkcii INT tu máme jednoduchý príklad pre prácu s peniazmi:

```
10 INPUT "zadaj sumu: ";a
20 LET a$=STR$ (INT (a*100+.5)/100)
30 IF a$(1)="." THEN LET a$="0"+a$
40 LET b=LEN a$-LEN STR$ INT VAL a$
50 LET a$=a$+(".00" AND B=0)+("0" AND B=2)
60 PRINT a$;" Kcs"
70 GO TO 10
```

RND - Touto funkciou bude generované náhodné číslo medzi 0-1.

SGN - Táto funkcia skúša znamienko (signum) argumentu.
Jej výsledkom je: 1 keď je argument kladné číslo
0 keď je argument 0
-1 keď je argument záporné číslo

EXP - Exponenciálna hodnota EXP x vypočíta hodnotu e^x, kde e je základ prirodzených logaritmov (e = 2.7182818...).

LN - Funkcia LN x je prirodzený logaritmus x. Obecne platí, že logaritmus čísla x so základom z je výraz, na ktorý musíme umocniť základ z, aby sme získali číslo x.

SQR - Funkcia SQR x vypočíta druhú odmocninu čísla x.

T R I G O N O M E T R I C K É F U N K C I E

Trigonometrické funkcie si najlepšie vysvetlíme na pravouhlom trojuholníku. Keď si ho nakreslíme a jeden z jeho nepravých uhlov si označíme ako α, potom bude platiť:

SIN α = podiel protiľahlej odvesny a prepony

COS α = podiel priľahlej odvesny a prepony

TAN α = podiel protiľahlej odvesny a priľahlej odvesny

P S E U D O N Á H O D N É Č Í S L A

Počítač samozrejme nemôže generovať celkom náhodné čísla. Preto sa čísla, ktoré generuje volajú pseudonáhodné. Nie su uplne náhodné, ale k náhode sa velmi približujú. Len pri velmi velkom množstve takto vygenerovaných čísiel a pri pozornom štúdiu by sme možno našli v ich generovaní nejaký systém. Pre bežné aplikácie kde potrebujeme, aby nejaký proces prebiehal náhodne však tieto čísla môžeme smelo použiť.

Na generovanie náhodných čísiel nám slúži funkcia RND. Táto funkcia

generuje náhodné čísla medzi číslom 0 a 1, pričom hodnotu 1 nikdy nedosiahne. Skuste napísať tento riadok a uvidíte prácu funkcie RND:

```
10 PRINT RND: GO TO 10
```

Ďalšia funkcia, ktorá sa pri práci s náhodnými číslami môže používať je funkcia RANDOMIZE. Môže sa používať s parametrom, alebo bez parametru. Ak napríklad zadáte nasledujúci riadok

```
10 RANDOMIZE 1: PRINT RND: GO TO 10
```

potom budú náhodne čísla začínať vždy hodnotou 0.0022735596. Ako parameter za funkciou RANDOMIZE môžete samozrejme použiť ľubovoľné číslo medzi 1-65535. Ak zadáte príkaz RANDOMIZE bez parametru (RANDOMIZE 0 má rovnaký účinok), zbadáte, že postupnosť náhodných čísiel, ktorú dostanete využíva čas od zapnutia počítača.

```
10 RANDOMIZE: PRINT RND: GO TO 10
```

Celkovo možno zhrnúť, že RND a RANDOMIZE sú veľmi praktické a užitočné funcie, ktoré nájdu svoje uplatnenie rovnako v hrách ako v špeciálnych matematických aplikáciach.

P R E N O S C U D Z Í C H D I A L E K T O V B A S I C

Vo všetkých možných knižkách a počítačových časopisoch nájdete hojnosť výpisov rôznych BASIC programov. Akonáhle však chcete takýto BASIC prepísať do Vášho počítača, zistíte, že obsahuje množstvo príkazov, ktoré Spectrum neovláda. Cudzí dialekt BASICu musíte vlastne preložiť do reči Vášho Spectra. Takáto práca zväčša nie je jednoduchá. Koľko príkazov budete musieť zmeniť, závisí na štruktúre jednotlivých programov. Chceme Vám dať aspoň niekoľko základných rád, ktoré sa Vám budú hodiť, ak budete prepracovávať pre Váš počítač cudzí program.

CELÉ ČÍSLA

Principiálne by ste mali pred delením zaviesť vždy funkciu INT, ak pre počítač koncipujete celočíselnú aritmetiku. Pridajte do delenia zátvorky tak, aby INT platilo len pre toto delenie.

DIM

Niektoré dialekty dovoľujú viac DIM príkazom písať na jeden riadok, napr. 10 DIM a$(9), b$(8), c$(7). U Spectra musíte v takomto prípade použiť vždy nový riadok.

Ak má program, ktorý prepracovávate pre Spectrum pole, ktorého názov je dlhší ako jedno písmeno, musíte ho skrátiť len na jedno písmeno. Napr. NUM$(8) prerobíte na N$(8). Ak potrebujete viac polí ako máte písmen k dispozícii, je tu len jedna možnosť: prebývajúce pole uložiť do druhej časti programu. Vystríhajte sa nulových dimenzií poľa.

GET, GET$

Toto je funkcia, ktorá číta charakter klavesy, ktora je práve stlačena. Jednotlivé dialekty potrebuju rôzne tvary, ale principialne sa GET používa na ošetrenie stlačenej klavesy a dáva buď charakter stlačenej klávesy, alebo ASCII kod tohoto charakteru. Syntax takehoto príkazu vyzera zvačša takto: GET a$ alebo LET a$=GET$. U Spectra tento príkaz nahradíme pomocou tychto dvoch riadkov:

```
1000 LET a$=INKEY$
1010 IF a$="" THEN GO TO 1000
```

Tým dostaneme charakter stlačenej klavesy. Ak potrebujeme ASCII kod tejto klávesy, použijeme nasledujúci program:

```
1000 LET a$=INKEY$
1010 IF a$="" THEN GO TO 1000
1020 LET a=CODE a$
1030 PRINT a
```

Teraz si pozrime príklad, ked chceme čitať len konkrétne klávesy, v našom prípade klávesy 0-9. Ostatné klávesy majú byť ignorované a z prečítanej klávesy sa nakoniec ma vypočítať ASCII kod. Prevedenie môže vyzerať napríklad takto:

```
1000 LET a$=INKEY$
1010 IF a$<"0" OR a$>"9" THEN GO TO 1000
1020 LET a=VAL a$
1030 PRINT a
```

INKEY$ (X)

Tento príkaz je vlastne INKEY$ s časovým limitom pre odpoveď, teda do budúceho stlačenia klávesy, napríklad:

```
100 LET a$=INKEY$ (x)
```

kde je x hodnotou pre časový limit. Toto môžeme nahradiť dvojakým postupom. Jedna možnosť:

```
10 LET x=50
20 PAUSE x
30 LET a$=INKEY$
```

alebo týmto spôsobom, ktorým zároveň demonštrujeme jednoduchú hru:

```
 5 REM tlačenie kláves
10 LET b$=CHR$ (INT (RND*26)+CODE "A")
20 PRINT AT 10,0;"Stlac rychlo klavesu ";b$
30 FOR a=0 TO 100
40 LET a$=INKEY$
50 IF a$<>"" THEN GO TO 80
60 NEXT a
70 PRINT '"Cas je prekroceny !": STOP
```

```
80 IF a$=b$ THEN PRINT '"Vyborne, stihol si to !": STOP
90 PRINT "Stlacil si nespravnu klavesu !"
```

VAL

Ak argument u VAL nedáva žiadny platný numerický výraz, dostanete chybové hlásenie. Iné BASIC dialekty ponechávajú v tomto prípade argument rovný 0.

SET, RESET

Tento príkaz niektoré BASIC dialekty používajú vtedy, keď je potrebné jednotlivé body tlačiť na obrazovku bielou alebo čiernou farbou. Nahradíme ho pomocou príkazov Spectra PLOT a PLOT OVER.

ELSE

Toto je príkaz, ktorý je doplnkom príkazovej dvojice IF - THEN a umožňuje druhé odbočenie potom, ak nie je podmienka splnená. Na Vašom počítači musíte druhé odbočenie nahradiť napríklad takto:

cudzi dialekt:

```
20 IF x=1 THEN LET y=7 ELSE GO TO 80
```

Spectrum:

```
20 IF x=1 THEN LET y=7
21 IF x<>1 THEN GO TO 80
```

Týmto bude ELSE obstojne nahradené. Pri niektorých aplikáciach existuje však aj elegantnejšie riešenie:

cudzi dialekt:

```
50 IF x=1 THEN LET y=7 ELSE LET y=8
```

Spectrum:

```
50 LET y=(7 AND x=1)+(8 AND x<>1)
```

Nedá sa tu ukázať univerzálne riešenie, pretože môžu existovať najrôznejšie aplikácie tohoto príkazu. Naše príklady však ukazujú, ktorá cesta je pre optimálne riešenie najvhodnejšia.

Ukážme si jednu komplikovanejšiu aplikáciu príkazu ELSE:

```
10 IF x=1 THEN LET y=1 ELSE IF x=5 THEN GO TO 100
```

Môže byť nahradené postupnosťou:

```
10 IF x=1 THEN LET y=1
```

```
11 IF x<>1 THEN IF x=5 THEN GO TO 100
```

alebo:

```
10 IF x=1 THEN LET y=1
11 IF x<>1 AND x=5 THEN GO TO 100
```

REPEAT, UNTIL

Pomocou týchto príkazov dostaneme slučku, ktorá beží až do určenej podmienky, pokiaľ táto nie je splnena. Je ťažké ponuknuť generálnu metodu. Riešenie sa často ponúka pomocou IF - THEN príkazu ako uvadza tento príklad:

```
10 PRINT "Zadaj ano alebo nie"
20 REPEAT
30 INPUT a$
40 UNTIL a$="ano" OR a$="nie"
```

nahradíme pomocou:

```
10 INPUT "Zadaj ano alebo nie";a$
20 IF a$<>"ano" AND a$<>"nie" THEN GO TO 10
```

Štruktúry REPEAT, UNTIL sú častejšie vo väčších komplexoch než v našom príklade a nie vždy si môžeme pomôcť prostrednictvom IF - THEN.

NEDEFINOVANÉ PREMENNÉ

Ako Ste už iste zistili, premenné, ktoré u Spectra nie sú dopredu definované, sa pri prvom použití prejavia chybovým hlásením. Niektoré dialekty túto definíciu dopredu nepotrebujú a uložia automaticky začiatočnú hodnotu rovnú nule. Dávajte pozor pri prepisovaní programov pre Vaše Spectrum, aby boli všetky premenné definované skôr, ako budu prvýkrát použité.

MATICE

Niektoré dialekty majú maticové funkcie, prevádzajúce operácie v poliach. Spectrum túto možnosť nemá a preto je nutné operácie s prvkami poli prevádzať jednotlivo pomocou slučky.

cudzí dialekt:

```
10 DIM x(y)
20 DIM p(y)
30 MAT x=p
```

Spectrum:

```
10 LET n=0
20 DIM x(y)
```

```
30 DIM p(y)
40 LET n=n+1
50 LET x(n)=p(n)
60 IF n<>y THEN GO TO 40
```

```
DEF PROC, ENDPROC
```

Tato dvojica prikazov sa používa pre podprogramy, ked behom spracovania programu je prevedeny odskok do podprogramu, jeho čitanie a vlastne prevádzanie. Programy a ich vypisy su tak prehľadnejšie a lepšie čitatelné. Hlavný rozoznavaci znak pre skok do takehoto podprogramu je meno a nie číslo riadku, na ktorom je podprogram uložený. Môžeme si to ukazat na priklade:

```
 100 CALL bod
1000 DEF PROC bod
1010 PRINT "bod ";s
1020 ENDPROC
```

ENDPROC funguje podobne ako RETURN, to znamená, že po vykonani podprogramu príkaz ENDPROC vracia riadenie späť pod riadok, na ktorom bol podprogram vyvolaný. V našom prípade teda pod riadok 100. V BASICu Spectra nemôžeme vyvolávať podprogramy menom.

Pri preklade teda zavedieme GO SUB nasledované číslom riadku a môžeme pre väčšiu prehľadnosť označiť podprogram pomocou príkazu REM, čím zároveň bude jasné, čo sa v podprograme vykoná.

```
 100 GO SUB 1000
1000 REM podprogram bod
1010 PRINT "bod ";s
1020 RETURN
```

Ak sa nechcete vzdať uľahčenia práce, môžete nadefinovať na začiatku programu premennú, ktorej hodnota je rovná číslu riadku, na ktorom začina podprogram. Potom môžete vyvolať podprogram pomocou príkazu GO SUB meno.

Dobrá pomôcka pre sprehľadnenie programu je tiež prepisanie REM návestia, ktoré je na začiatku podprogramu, inou farbou PAPER a INK. Tým pri listovaní zreteľne vyniknú všetky podprogramy.

```
  50 LET bod=1000
 100 GO SUB bod
1000 REM podprogram bod
1010 PRINT "bod ";s
1020 RETURN
```

Tiež príkaz PROC je veľmi komplikovaná záležitosť. Často urobíte lepšie, ked tento výraz nahradíte v Spectre jednoduchým podprogramom pomocou príkazov GO SUB a RETURN.

```
INSTR (A$,B$)
```

Táto funkcia zisťuje či sa reťazec b$ nachádza v reťazci a$. Ak sa nachádza, výsledkom funkcie bude číslo určujúce poziciu retazca b$ v

reťazci a$. Napríklad ak b$="PUT" a a$="COMPUTER" tak dostaneme pomocou INSTR (a$,b$) hodnotu 4, pretože časť reťazca a$, ktorá je zhodná s "PUT" začína na štvrtom prvku. Ak sa reťazec b$ v reťazci a$ nenachádza, bude výsledná hodnota 0.

Ak chceme túto funkciu previesť do BASICu Spectra, musíme napísať tento podprogram:

```
   10 REM Y=INSTR (A$,B$)
   20 INPUT a$
   30 INPUT b$
   40 PRINT a$;" ";b$
   50 GO SUB 1000
   60 PRINT "INSTR (a$,b$)=";y
   70 GO TO 20
 1000 REM funkcia INSTR
 1010 LET y=0
 1020 IF LEN a$=0 OR LEN b$=0 OR LEN b$>LEN a$ THEN RETURN
 1030 FOR y=1 TO LEN a$-LEN b$+1
 1040 IF a$(y TO y+LEN b$-1)=b$ THEN RETURN
 1050 NEXT y
 1060 LET y=0
 1070 RETURN
```

DIV

Táto funkcia zaokrúhľuje výsledky delení na najbližšie celé číslo. Napríklad 17 DIV 5 dá výsledok 3. Na Spectre sa táto funkcia dá nahradiť jednoducho pomocou funkcie INT. Ukážeme si to na už spomenutom príklade:

```
17 DIV 5
```

teda napíšte ako:

```
INT(17/5)
```

a dostanete úplne zaručene rovnaký výsledok.

MOD

Funkcia MOD slúži k získaniu zvyšku delenia dvoch celých čísel. Napríklad 17 MOD 5 = 2, pretože 17/5 =3 a zvyšok je 2. Na Spectre môžete prepísať príkaz a MOD b takto:

```
a-INT((a/b)*b)
```

TAB

Niektoré počítače majú za TAB dva argumenty pre presun PRINT pozície. V takomto prípade má príkaz TAB v cudzom dialekte BASICu tú istú funkciu ako príkaz AT v BASICu Spectra. Je veľa dialektov BASICu, ktoré menia koordináty PRINT pozície tak, že TAB (x,y) je úplne rovnocenné s AT (y,x).

STUPNE A UHLOVA MIERA

Niektoré dialekty BASICu majú špeciálne funkcie na výpočty stupňov a uhlovej miery. Do BASICu Spectra ich zväčša prevedieme pomocou funkcie PI.

uhlová miera=(PI*GRAD)/180

Radiány sa prepočítajú do uhlových stupňov takto:

GRAD=(180*uhlová miera)/PI

LOGARITMY NA ZÁKLADE 10

Spectrum počíta s prirodzenými logaritmami. Ak potrebujete niekedy logaritmy na základe 10 dostanete ich nasledujúcim prepočtovým vzorocom:

LOG BASIC 10 (x)=(LN(x)/(LN(10))

Môžete vypočítať logaritmy na akomkoľvek základe, keď prevediete obecný vzorec (pre logaritmus x na základe b):

LOG BASIC (x)=(LN(x)/(LN/b))

PERCENTÁ (%)

Symbol percenta je v niektorých cudzích dialektoch BASICu obecne použiteľný, ak chceme špecifikovať celočíselné premenné. Môže sa tým ušetriť miesto v pamäti a takáto premenná tiež pracuje rýchlejšie ako konvenčné premenné. Principiálne tu nie je žiadny problém. V BASICu Spectrum v tomto prípade namiesto znaku % použijeme celkom normálnu premennú.

Budte však opatrný, ak sú napríklad tieto premenné použité v delení a Vy si nieste istý či vychádzajú celé čísla. V tomto prípade napíšte tento vzorec: LET a=INT (a/2) a zaistite si, že budete mať čo do činenia len s celými číslami.

OTÁZNIK (?)

Otáznik je u mnohých počítačov skratka pre prevedenie PRINT príkazu.

P E E K A P O K E

Tieto dva príkazy sú silnou zbraňou Vašeho počítača. Umožňujú Vám vykonať veci, ktoré nedosiahnete žiadnym iným programovacím spôsobom.

Začneme tým, že si niečo o príkazoch PEEK a POKE povieme.

PEEK m dá číslo, ktoré je uložené v pamäti počítača na adrese m

POKE m,n uloží číslo n do pamäte počítača na adresu m

Počítač má 65536 (0-65535) pamäťových adries. Na všetky tieto adresy (alebo pamäťové miesta) môžeme aplikovať príkazy PEEK a POKE. Ale s tým rozdielom, že pokiaľ funkcia PEEK nám dá vždy správnu hodnotu v celom rozsahu pamäte, funkcia POKE pracuje len na adresách 16384-65535. Je to tak preto, že celá pamäť je rozdelená vlastne na dve základne časti. Sú to ROM (0-16383) a RAM (16384-65535). Teraz si o nich povieme niečo bližšie:

ROM (Read Only Memory) pamäť len na čítanie

Už samotný názov tejto pamäte napovedá aká je jej funkcia. Do tohoto druhu pamäte sú napevno uložené programové rutiny, ktoré počítač potrebuje hneď po zapojení do siete. V našom prípade je to Spectrum BASIC, rôzne chybové hlásenia a rutiny na prácu s obrazovou pamäťou a periferiami. Túto časť pamäte nemožno premazať novými informáciami. Príkaz POKE tu síce akoby funguje, ale zmenu v pamäti samozrejme nemôže vykonať. Informácie v tejto časti pamäti sa nestrácajú ani po vypnutí počítača.

RAM alebo RWM (Read Write Memory) pamäť na čítanie i písanie

Táto časť pamäte je voľne prístupná programátorovi. Možno do nej čokoľvek zapísať, prečítať to alebo prepísať novou hodnotou. Jej nevýhoda ale spočíva v tom, že pri odpojení počítača zo siete, alebo pri výpadku elektrického prúdu sa nenávratne stratia všetky informácie do nej uložené. Preto sa používajú vonkajšie pamäťové média ako datarekordér, microdrive, alebo disketová jednotka. Na tieto média je možné hotový program z RAM preniesť a v prípade potreby jeho opätovného použitia ho znova nahrať do pamäti RAM počítača.

Pre obidve časti pamäte platí, že na každé pamäťové miesto je možné zapísať maximálne číslo v rozpätí 0-255. Je to dané tým, že každé pamäťové miesto tvorí pamäťová bunka nazvaná byte. Každý byte obsahuje osem základných pamäťových buniek - bitov. Každý z týchto bitov môže mať hodnotu 1 alebo 0, čiže môže byť nastavený, alebo nenastavený. Najjednoduchšie si takýto bit môžeme predstaviť ako malý vypínač. Podľa toho, či je niektorý z ôsmich bitov v byte nastavený alebo nie, vieme aké číslo je v byte zapísané. Najlepšie bude, keď si to ukážeme názorne.

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	1
128	64	32	16	8	4	2	1

Ako vidíte, je to veľmi jednoduché. Bity v byte sú očíslované z prava do ľava číslami 0-7 a každý z nich predstavuje nejakú hodnotu. Táto hodnota sa rátа iba vtedy ak je príslušný bit nastavený. Najmenšia hodnota 0 je v byte teda obsiahnutá vtedy, ak majú všetky bity hodnotu 0. Najväčšia hodnota 255 je v byte obsiahnutá naopak vtedy, ak majú všetky bity hodnotu 1. Veľmi ľahko si vypočítate, že 128+64+32+16+8+4+2+1=255.

V príklade na našom obrázku je v byte číslo 9, pretože sú nastavené bity 0 a 3 a teda 1+8=9. Ale vráťme sa k našim príkazom PEEK a POKE.

REM HLÁSENIE

Veľa programov používa informácie, ktoré sú obsiahnuté v REM hlásení väčšinou na prvom riadku BASIC programov, pretože ich možno ľahko nájsť. Pre náš príklad musíme vedieť, že adresa prvého znaku slova za REM na prvom riadku BASICu je 23760. Keď napíšeme program:

```
1 REM ABCDF
2 PRINT PEEK 23760
```

Dostaneme na obrazovke číslo 65, pretože ASCII kód znaku A uložený na adrese 23760 má hodnotu 65.
Tento znak môžeme ľahko zmeniť na Z ak vieme kód znaku Z čo je 90. Napíšme teda príkaz:

```
POKE 23760,90
```

Rovnaký efekt dosiahneme, keď zadáme príkaz: POKE 23760,CODE "Z". Nasledujúca adresa (23761) obsahuje ASCII kód písmena B, nasledujúca (23762) kód znaku C atď.

POUŽITIE ČASOVACEJ JEDNOTKY

Niektoré adresy v pamäti RAM obsahujú systémové premenné, obsahujúce zaujímavé informácie, ktoré môžeme využiť na rôzne aplikácie.
Časovacia jednotka je na adresách 23672, 23673 a 23674. Tieto adresy môžeme využiť pomocou PEEK a POKE na získanie pomerne presných hodín.

Aby sme nastavili časovaciu jednotku na 0 zadáme:

```
23674,255
23673,255
23672,255
```

a aby sme mohli čítať jej hodnotu, použijeme tento výraz:

```
65536*PEEK 23672+256*PEEK 23673
```

Pretože časovacia jednotka riadi zobrazovanie na obrazovke a pretože je na obrazovku zasielaných 50 obrazov za sekundu (v USA 60), potrebujeme výsledné číslo deliť hodnotou 50 pre získanie sekundového rytmu.

Napíšte si teraz tento program:

```
 5 POKE 23674,255
10 POKE 23673,255
20 POKE 23672,255
30 LET cas=(PEEK 23672+256*PEEK 23673)/50
40 PRINT AT 11,14;INT ((cas*10)/10);" "
50 GO TO 30
```

Príkaz INT na riadku 40 bude doplnený preto, aby ste sa vyhli tlači sekundových zlomkov menších ako 1/10. Tieto stopky su takmer presné. Môžete ich použiť v programe a môžete zobrazovať presný čas podľa toho ako

dlho program prebieha. Kapacita obrazoveho čitača použita v timingu trvá štyri dni. Ak chcete mat čas v minútach a sekundach, použite tento podprogram:

```
 5 POKE 23674,255
10 POKE 23673,255
20 POKE 23672,255
30 LET cas=(PEEK 23672+256*PEEK 23673)/50
40 PRINT AT 11,14;INT (cas/60);":";INT (cas-INT (cas/60)*60);" "
50 GO TO 30
```

O B C H O D N É P R O G R A M Y

V nasledujúcom krátkom priereze Vám chceme predviesť pár možností praktickeho nasadenia počítača na komerčné účely v obchode, účtovnicte a pri spracovaní textu či evidencii.

Ako prvý si ukážeme finančný program pre výpočet úrokov. Podrobnejšie vysvetlenie nie je nutné, pretože sprievodné texty v programe sú dostatočne zrozumiteľné.

```
 10 INPUT "Zakladny kapital: ";k: LET s=k
 20 INPUT "Urokova sadzba: ";u
 30 INPUT "Pocet rokov: ";r
 40 PRINT AT 0,0;"ROK";TAB 8;"UROK";TAB 16;"SPOLU"
 50 FOR f=1 TO r
 60 GO SUB 100
 70 PRINT f;TAB 8;ul;TAB 16;s
 80 NEXT f
 90 STOP
100 LET ul=(INT (((s/100)*u)*10)+.5)/10
110 LET s=s+ul
120 RETURN
```

S P R A C O V A N I E T E X T U

Nasledujúci program je jednoduchý textový editor. Slúži na napísanie textu na obrazovku, jeho opravu a následné vytlačenie na pripojenej tlačiarni. Profesionálny textový editor má samozrejme oveľa širšie možnosti ako náš, ale na demonštráciu tejto najrozšírenejšej aplikácie počítača v administratíve nám úplne postačí.

V našom editore môžeme napísať maximálne 17 riadkov textu, opraviť prípadné chyby a hotový text vytlačiť.

V prípade voľby "Korektúra" sa môžeme po texte pohybovať vo všetkých smeroch pomocou kurzorových tlačítiek (klávesy 5,6,7,8). Našu pozíciu označuje kurzor v tvare "≠". Ak stlačíme klávesu "o" môžeme opraviť znak, na ktorom práve stojí kurzor. Stlačenie klávesy "1" nás vráti do základného menu, kde si môžeme zvoliť tlač textu na tlačiarni, alebo písanie nového textu.

```
 5 REM SPRACOVANIE TEXTU
10 LET n=0
20 PRINT "Zadajte text !"
30 INPUT x$
```

```
  32 LET x$=x$+" "
  35 CLS
  40 GO SUB 1000
  45 GO SUB 1000
  50 CLS: PRINT AT 0,0;x$
  60 PRINT AT 17,8;"1 - Korektura"
  70 PRINT 'TAB 8; "2 - Vytlac text"
  80 PRINT 'TAB 8; "3 - Novy start  "
  90 INPUT q
 100 IF q=3 THEN RUN
 110 IF q=2 THEN GO TO 4000
 120 IF q-1 THEN GO TO 2000
 130 GO TO 80
1000 REM Ziadny zbytok slov na konci riadku
1010 LET n=n+1
1020 GO SUB 1180
1030 LET n=n+33
1040 IF n>=LEN x$ THEN RETURN
1045 REM Jednotlive prazdne miesta v nasledujucom riadku
1050 IF x$(n)=" " THEN GO TO 1160
1065 REM Jednotlive prazdne miesta v nasledujucom riadku
1070 IF x$(n)=" " THEN GO TO 1030
1080 LET j=0
1090 GO SUB 1180
1100 LET j=j+1
1105 REM jednotlive prazdne miesta na nasledujucom riadku
1110 IF n>0 THEN IF x$(n)>" " THEN GO TO 1090
1120 FOR n=n TO n+j-1
1125 REM Jednotlive prazdne miesta na nasledujucom riadku
1130 LET x$=x$(1 TO n)+" "+x$(n+1 TO)
1140 NEXT n
1150 GO TO 1030
1160 LET x$=x$(1 TO n-1)+x$(n+1 TO)
1170 GO TO 1020
1180 LET n=n-1
1190 RETURN
2000 REM Korektura
2010 CLS
2020 PRINT AT 21,8;"1 - Spat do menu"
2030 LET a=1: LET l=LEN x$
2032 LET z$=x$(1)
2035 PRINT AT 2,0;x$
2037 LET x$(a)=z$
2050 IF INKEY$="8" AND a<l THEN LET a=a+1
2055 IF INKEY$="6" AND a+32<=LEN x$ AND a<l+32 THEN LET a=a+32
2060 IF INKEY$="5" AND a>1 THEN LET a=a-1
2065 IF INKEY$="7" AND a>32 THEN LET a=a-32
2070 IF INKEY$="1" THEN GO TO 50
2080 IF INKEY$ ="o" THEN GO SUB 3000
2090 PRINT AT 0,0;a;" ";x$(a);" "
2100 LET z$=x$(a)
2110 LET x$(a)="#"
2120 GO TO 2035
3000 INPUT PAPER 2; INK 6;"Vloz pismeno:";h$
3010 LET x$(a)=h$
```

IN 31

```
3020 PRINT AT 2,0;x$
3030 RETURN
4000 LPRINT x$
4010 PRINT AT 17,8;"1 - Vytlac text"
4020 PRINT 'TAB 8; "2 - Novy start "
4030 PRINT 'TAB 8; "3 - Koniec programu"
4040 INPUT u
4050 IF u=1 THEN GO TO 4000
4060 IF u=2 THEN RUN
4070 IF u=3 THEN STOP
4080 GO TO 4040
```

K N I Ž N Ý I N D E X

Posledny program tejto rady sl ži k zjednodušeniu práce v knižnici. Možno pomocou neho vyrobiť pre každú knihu špeciálny kartotékový listok, na ktorom bude v zahlavi názov publikácie a jej autor. Pod týmito udajmi budú abecedne zoradené najdôležitejšie kapitoly z knihy nasledované číslom strany. Tieto udaje je možné do počítača zadať v ľubovolnom poradi, pretože program ich prehľadne zoradi podľá abecedneho poriadku.

Program pozostáva z troch časti. V prvej (po riadok 100) budú prevedené vstupy, v druhej (riadok 200 až 300) je rutina samotného triedenia a v tretej (riadok 310 až 530) bude prevedený kontrolný výpis na obrazovku, alebo tlač kartotékového listku na pripojenej tlačiarni.

```
 10 REM KNIZNY INDEX
 20 DIM a$(400,16)
 30 INPUT "Zadajte titul: ";t$
 40 INPUT "Zadajte autora: ";n$
 45 PRINT "Vlozte heslo a stranu max.16 zn.Koniec = samotny ENTER"''
 50 FOR g=1 TO 400
 60 INPUT a$(g)
 70 REM V nasledujucom riadku je 16 medzier
 80 IF a$(g)="                " THEN GO TO 200
 90 PRINT a$(g)
100 NEXT g
200 CLS
210 PRINT "Triedim udaje ... ";
220 FOR b=1 TO g-1
230 FOR c=b+1 TO g-1
240 LET b$=a$(b)
250 LET c$=a$(c)
260 IF b$=c$ THEN GO TO 280
270 IF b$>c$ THEN LET a$(b)=c$: LET a$(c)=b$
280 NEXT c
290 NEXT b
300 PRINT "Hotovo !"
310 PRINT ''"TEXT NA: 1-Tlaciaren 2-Obrazovku"
320 IF INKEY$ ="2" THEN CLS: GO TO 440
340 IF INKEY$ ="1" THEN GO TO 360
350 GO TO 320
360 LPRINT t$
370 LPRINT
380 LPRINT n$
```

```
390 LPRINT
400 FOR a=1 TO g-1
410 LPRINT a$(a)
420 NEXT a
430 STOP
440 PRINT t$
450 PRINT n$
460 PRINT "--------------------------------"
500 FOR a=1 TO g-1
520 PRINT a$(a)
530 NEXT a
```

V L A S T N É V Y L E P Š O V A N I E P R O G R A M O V

Iste s nemalou námahou ste prešli v priebehu učenia programovania v jazyku BASIC celou radou štádií. Už pri Vašom prvom stretnutí s BASICom, keď ste začali písať prvé krátke programy, vznikla potreba prehľadného a zároveň čo najefektívnejšieho zápisu programov. Veci okolo programovania sa niekedy vysvetľujú najlepšie názorne na ukážke a ak si Vaši priatelia pridú občas po radu, lepšie než rozsiahle vysvetľovanie je keď máte vždy po ruke nejaké tie výpisy programov.

Počas Vášho programátorského rastu tiež prídete, alebo ste už prišli bezpečne do štádia, keď Vám už nebude stačiť vedieť program napísať ľubovoľnou metódou, ale začnete hľadať rôzne nové a účinnejšie metódy spracovania rovnakého problému. Keďže Vám v tejto Vašej fáze programátorského dospievania chceme pomôcť, uvedieme aspoň niekoľko tipov, ktoré Vám môžu niektoré problémy uľahčiť.

Najskôr sa pozrieme na výpis programu. Niektoré, hlavne rozsiahle programy, su značne neprehľadné. Po dlhšej dobe, keď sa k programu vrátite aby ste do neho niečo dorobili, často už ani sami neviete, čo a ako funguje. Ak sa naučíte premyslene používať REM návestia, môžete program spraviť oveľa prehľadnejším. Táto metóda je veľmi účinná hlavne pre označovanie rôznych podprogramov. Pri nich môžete za prvým REM návestím uviesť názov podprogramu, prípadne krátku poznámku na čo slúži. Ďalšími dvoma REM návestiami za ktorými budú znamienka mínus, alebo iné vhodné znaky, môžete podprogram výrazne oddeliť od samotného programu. Prirodzene niekedy sa REM návestie stáva luxusom, najma vtedy, keď musíte bojovať o každý byt v pamäti. Ale ak sa ho naučíte používať efektívne a zároveň úsporne, stane sa iste Vaším dobrým pomocníkom.

Pozrite sa veľmi kriticky ako často používate príkaz GO TO. Veľký výskyt týchto príkazov urobí aj z toho najlepšieho programu neprehľadnú džungľu, v ktorej sa po chvíli prestane orientovať aj sám programátor. Pokiaľ je to možné, naučte sa programovať tak, aby program postupoval odzhora nadol, tak ako je napísaný. Pre často sa opakujúce sekvencie používajte podprogramy. Príkaz GO TO použite naozaj len v nevyhnutných prípadoch.

Tu je malá ukážka, ako by mal vyzerať prehľadný program:

```
10 REM Vypocet mocniny a odmocniny
20 REM Definovanie premennych
30 LET a=0: LET b=0: LET c=0
40 INPUT "Zadaj cislo: ";a
50 GO SUB 100
```

```
 60 PRINT "Zadane cislo:     ";a
 70 PRINT "Druha mocnina:    ";b
 80 PRINT "Druha odmocnina: ";c
 90 STOP
100 REM Vypocty
110 REM -------
120 LET b=a^2
130 LET c=SQR a
140 REM -------
150 RETURN
```

Prezrite si, či su vo všetkych častiach programu správne napísané a uzavreté slučky FOR-NEXT. Skontrolujte, či po každom volaní podprogramu príkazom GO SUB nasleduje návrat do hlavného programu príkazom RETURN.

Píšte program sektor po sektore, pričom vždy najprv vyskúšajte či jednotlivá časť funguje a až potom píšte ďalšiu. Ak budete takto v programe postupovať krok po kroku, nestratíte prehľad a ak sa vyskytne pri skúšaní eventuálne nejaké to hlásenie chyby, máte ho ihneď pod kontrolou.

Ak Vám nechýba pamäťový priestor, zaveďte riadky s návestím REM a s náplňou grafických znakov. Myslite na to, že odmenou za Vašu snahu bude program, ktory správne beží a naviac je prehľadný a jednotlivé podprogramy sú vo výpise opticky ľahko rozlíšiteľné.

Druhým dôvodom, prečo má byť výpis Vášho programu maximálne jasný a prehľadný, je realita, že s rastúcou knižnicou programov, ktoré budete vytvárať, po čase stratíte prehľad, na čo ktorý program slúži. Jasné komentáre na začiatkoch programov tento problém spoľahlivo odstráni.

Píšte každý program tak, aby ho mohol použiť bez problémov aj človek, ktorý nevie nič o jeho funkcii. Preto používajte často komentáre na obrazovke a využívajte možnosť hlásenia v príkaze INPUT. Ak svoj program vybavíte dostatočnými sprievodnými textami, nebude nutné písať k nemu podrobný manuál, ktorému by ste sa inak asi nevyhli.

A teraz niekoľko slov k vstupom. Aj keď Váš program bude napísaný bez najmenších chýb, nie je ešte stále úplne bezpečný. Najčastejšie chyby môžu vzniknúť pri vkladaní údajov. Človek, ktorý s počítačom často nepracuje, ľahko urobí chybu a vloží nesprávny údaj, čo má za následok chybný výpočet alebo prerušenie programu a vypísanie chybového hlásenia. Najčastejšie chyby vznikajú tak, že je zadané väčšie alebo menšie číslo ako počítač práve očakáva, alebo je zadaný reťazec namiesto čísla. Častou chybou u začiatočníkov je vkladanie písmena O namiesto číslice 0 (nula). Takýmto chybám môžeme ľahko predísť tak, že budeme dopredu predvídať všetky možné chybné kroky užívateľa programu a budeme sa ich snažiť dopredu ošetriť. Tento spôsob ochrany programu pred neskúseným alebo nepoučeným užívateľom sa nazýva slangovo 'blbuvzdornosť' alebo (nech nám to nežné pohlavie odpustí), v USA tiež 'odolnosť voči ženám'. Blbuvzdornosť programu je jednou zo základných podmienok jeho širšieho komerčného využitia.

Jednoduchý príklad zabránenia vloženiu nesprávnych údajov:

```
10 INPUT "Zadaj cislo (1-10): ";a
20 IF a<1 OR a>10 THEN BEEP 1,1: GO TO 10
```

Ako vidíte, užívateľ je najskôr v hlásení príkazu INPUT dopredu upozornený, že má vložiť len číslo ohraničené intervalom v zátvorkách. Ak je to taký ignorant, že napriek tomu vloží iné číslo, program to nevyvedie z miery. Zatrúbi na nepoučiteľného užívateľa a pýta si číslo znova

dovtedy, pokial nie je zadane spravne.

Prikazy GO TO okrem toho, že robia program neprehľadným, znižuju aj jeho rychlosť. Preto nimi skutocne šetrite a používajte ich ako vzacne korenie.

Premenné, ktoré používate v programe, definujte na začiatku programu. Pokial Vám to pamat umožňuje, davajte im dlhšie zrozumiteľne nazvy. Ak date prednosť šetreniu pamate, napíšte aspoň na začiatku za navestim REM, na čo ktorá premenna slúži.

A ešte jedna rada. Najlepšie odskúšate Váš hotovy program tak, že ho spustite a za počítač posadite nezainteresovanu osobu, ktorá o funkcii programu nema potuchy. Ak si dotyčny človek nebude vedieť s programom rady, alebo sa mu dokonca podarí program doviesť k chybovemu hlaseniu, je to pre Vás znamenie, že na programe ešte treba pracovať.

Záverom ešte krátka rekapitulácia:

- Používajte hojne REM návestia.
- Píšte program prehľadne a logicky.
- Snažte sa o jasnu štruktúru programov a podprogramov.
- Uvážte, či program obsahuje GO TO príkazy v unosnom počte.
- Skuste dať programu atraktivnu a prehľadnu obrazovu formu.
- Starajte sa o zrozumiteľnosť návesti a poznamok.
- Ochráňte program a hlavne jeho vstupy pred zlym zadanim.
- Napíšte do každého programu dôkladné vysvetlenie jeho funkcie.
- Modifikujte program so zreteľom na jeho vylepšenie a zrýchlenie.
- Testujte hotový program aj v najnepravdepodobnejších situáciach.

P R O G R A M, P R O G R A M, P R O G R A M

A na konci tu ešte pre Vás máme za priehrštie programov:

P S I E Z Á V O D Y

```
 10 REM PSIE ZAVODY
 20 RANDOMIZE
 25 GO SUB 200
 30 FOR x=1 TO 22
 35 PRINT INK 4;TAB 30;"|"
 40 NEXT  x
 45 PRINT AT 0,5;"Vsadili ste na cislo ";w
 50 DIM a(9): FOR f=1 TO 9: LET a(f)=1: NEXT f
 55 FOR x=1 TO 9
 60 PRINT AT 2*x,0; INK x/2;x;AT 2*x,1;">"
 70 LET a(x)=a(x)+INT (RND+.5)
 80 PRINT AT 2*x,a(x); INK x/2;">"
 90 BEEP .01,3*x
 95 IF a(x)=30 THEN GO TO 115
100 NEXT x
110 GO TO 55
115 FOR g=1 TO 70 STEP 2
120 PRINT AT 18,6; INK 2;"Pes cislo ";x;" je vitaz !": BEEP .02,g
123 IF w=x THEN PRINT AT 20,5; INK 3;"A Vy ste tiez vyhrali !"
130 FLASH RND
```

```
140 NEXT g
145 FLASH 0
150 RUN
200 BORDER 0
205 PRINT AT 2,3; INK 2;"Vitajte na zavodnej trati !"
210 PRINT AT 5,6; INK 4;"Na starte je 9 psov."
220 INPUT PAPER 6;INK 2;"Vsadte na vitaza:";w
230 IF W<1 OR w>9 THEN GO TO 220
235 BORDER 2
240 CLS: RETURN
```

B I L I A R D O V Á G U Ľ A

```
 10 REM BILIARDOVA GULA
 20 BORDER 0: PAPER 7: INK 1: CLS
 30 LET a=1: LET b=1: LET c=RND*20: LET d=RND*20
 40 PRINT AT c,d;"*": PAUSE 4: PRINT AT c,d;" "
 50 IF c+b>21 OR c+b<-1 THEN LET b=-b: BEEP .02,-RND*15
 60 IF d+a>31 OR d+a<0 OR RND>.96 THEN LET a=-a: BEEP .1,RND*25
 70 LET c=c+b: LET d=d+a
 80 GO TO 40
```

F A R E B N É P I Š K V O R K Y

Náš ďalší program je hra, ktorá je Vám iste veľmi dobre známa zo školských lavíc. Sú to piškvorky. Pravidlá hry sú jednoduché. Na hracej ploche 8 x 8 bodov musíte vytvárať piškvorky, ktoré pozostávajú z predom dohodnutých znakov, kladených vedľa seba. V našom prípade počítač používa krížik (x) a hráč kolieska (o). Cieľom je získať čo najväčší počet piškvoriek. Čím väčší počet znakov v piškvorke je, tým viac bodov za ňu získate. Najmenšia možná piškvorka, za ktorú získate jeden bod pozostáva z troch znakov. Za piškvorku zo štyroch znakov získate body 2, za päťznakovú piškvorku 3 body atď...

Dobrá stratégia hry spočíva samozrejme nielen v útoku (čiže vytváraní piškvoriek, ale aj v obrane (tým, že počítaču bránite vo vytváraní vlastných piškvoriek).

Miesto, na ktoré chcete umiestniť Váš znak, zadávate v podobe dvojciferného čísla. Teda ak chcete umiestniť znak na súradnice 5,8 napíšete jednoducho 58.

V našej verzii piškvoriek sa body ratajú len za piškvorky postavené v kolmých smeroch (teda zvislo a vodorovne). Je len na Vašej šikovnosti, aby ste doplnili aj možnosť vytvárania šikmých smerov, prípadne hru obohatili o iné vylepšenia.

```
  5 REM FAREBNE PIŠKVORKY
 10 FOR f=7 TO 1 STEP -1: PRINT AT 1,8; INK f;"FAREBNE PISKVORKY":
    PAUSE 10: NEXT f
 25 DIM a(8,8): FOR b=1 TO 8: FOR c=1 TO 8
 40 LET a(b,c)=CODE "."
 50 NEXT c: NEXT b:LET p=0: LET r=0
120 INPUT "Chces zacinat ? (a/n) ";g$
123 FOR f=1 TO 7: PRINT AT 1,8; INK f;"FAREBNE PISKVORKY": PAUSE 10:
    NEXT f
```

```
 125 CLS: GO SUB 3000
 127 FOR f=7 TO 1 STEP -1: PRINT AT 1,8; INK f;"FAREBNE PISKVORKY":
     PAUSE 10: NEXT f
 130 IF CODE g$<>CODE "n" AND CODE g$<>CODE "N" THEN GO TO 2000
1000 PRINT INK 2;AT 10,16;"Premyslam..."
1005 LET rand=INT (RND+.5)
1010 LET s=CODE "x": LET t=CODE "o": LET u=CODE "."
1020 FOR b=1 TO 8
1030 FOR d=1 TO 8
1035 IF rand=1 THEN GO TO 1140
1040 IF a(b,d)=s THEN IF d>1 THEN IF a(b,d-1)=s THEN IF d<8 THEN IF
     a(b,d+1)=u THEN LET f=b: LET g=d+1: GO TO 1340
1050 IF a(b,d)=s THEN IF d>2 THEN IF a(b,d-1)=s THEN IF a(b,d-2)=u
     THEN LET f=b: LET g=d-2: GO TO 1340
1060 IF a(b,d)=s THEN IF b>1 THEN IF a(b-1,d)=s THEN IF b<8 THEN IF
     a(b+1,d)=u THEN LET f=b+1: LET g=d: GO TO 1340
1070 IF a(b,d)=s THEN IF b>2 THEN IF a(b-1,d)=s THEN IF a(b-2,d)=u
     THEN LET f=b-2: LET g=d: GO TO 1340
1080 IF rand=1 THEN GO TO 1190
1140 IF a(b,d)=t THEN IF d>1 THEN IF a(b,d-1)=t THEN IF d<8 THEN IF
     a(b,d+1)=u THEN LET f=b: LET g=d+1: GO TO 1340
1150 IF a(b,d)=t THEN IF d>2 THEN IF a(b,d-1)=t THEN IF a(b,d-2)=u
     THEN LET f=b: LET g=d-2: GO TO 1340
1160 IF a(b,d)=t THEN IF b>1 THEN IF a(b-1,d)=t THEN IF b<8 THEN IF
     a(b+1,d)=u THEN LET f=b+1: LET g=d: GO TO 1340
1170 IF a(b,d)=t THEN IF b>2 THEN IF a(b-1,d)=t THEN IF a(b-2,d)=u
     THEN LET f=b-2: LET g=d: GO TO 1340
1180 IF rand=1 THEN GO TO 1040
1190 NEXT d
1200 NEXT b
1330 LET f=INT (RND*8+.5): LET g=INT (RND*8+.5)
1335 IF a(f,g)<>u THEN GO TO 1330
1340 LET a(f,g)=s
1360 GO SUB 3000
2000 PRINT INK 1;AT 10,16;"Si na tahu"
2010 LET s=CODE "x": LET t=CODE "o": INPUT ">";r
2050 LET r$=STR r: LET m=VAL r$(1 TO 1): LET n=VAL r$(2 TO 2)
2060 IF (m<1 OR m>8) OR (n<1 OR n>8) THEN PRINT #0;AT 1,0;
     "Chybny tah !": BEEP .5,1: GO TO 2010
2070 IF a(m,n)=s OR a(m,n)=t THEN PRINT #0;AT 1,0;"Chybny tah !":
     BEEP .5,1: GO TO 2010
2080 LET a(m,n)=t
2090 GO SUB 3000: GO TO 1000
3000 PRINT AT 5,0;: BEEP .25,RND*5
3010 LET c=0: LET h=0: LET bod=0
3020 LET s=CODE "x": LET t=CODE "o"
3040 FOR b=1 TO 8
3060 FOR d=1 TO 8
3070 IF a(b,d)=s THEN PRINT INK 2;"x";
3075 IF a(b,d)=t THEN PRINT INK 1;"o";
3077 IF a(b,d)=CODE "." THEN PRINT INK 5;".";: LET bod=bod+1
3080 IF a(b,d)=s THEN IF d>2 THEN IF a(b,d-2)=s THEN IF a(b,d-1)=s
     THEN LET c=c+1
3085 IF a(b,d)=t THEN IF d>2 THEN IF a(b,d-2)=t THEN IF a(b,d-1)=t
     THEN LET h=h+1
```

```
3090 IF a(b,d)=s THEN IF b>2 THEN IF a(b-2,d)=s THEN IF a(b-1,d)=s
     THEN LET c=c+1
3095 IF a(b,d)=t THEN IF b>2 THEN IF a(b-2,d)=t THEN IF a(b-1,d)=t
     THEN LET h=h+1
3100 NEXT d
3110 PRINT INK 4;b
3120 NEXT b
3130 PRINT INK 4;"12345678"
3150 PRINT '' INK 2;"mam: ";c;"  "; INK 1;"mas: ";h;" "
3160 IF bod=0 THEN GO TO 5000
3170 RETURN
5000 FOR x=1 TO 10: FOR y=1 TO 7
5010 IF c>h THEN PRINT AT 21,3; INK y;"Vyhral som za stavu  ";c;":";h
5020 IF h>c THEN PRINT AT 21,4; INK y;"Vyhral si za stavu  ";h;":";c
5025 IF h=c THEN PRINT AT 21,1; INK y;"Partia skoncila remizou  ";
     h;":";c
5030 NEXT y: NEXT x
5040 PRINT AT 21,4;"Stlac lubovolnu klávesu !"
5050 PAUSE 0: RUN
```

G E N E R Á C I A

Teraz nasledujú dve verzie programu "Hra života", ktoré znázorňujú zrodenie, rozširovanie a zánik jednej bunečnej kolónie. Život v bunečnej kolónii prebieha podľa nasledujúcich pravidiel:

- každá bunka v kolónii môže mať osem susedov
- každá bunka s dvomi alebo tromi susedmi prežije do budúcej generácie
- ak sú tri susedné bunky (a len tieto tri) potom zrodia novú bunku
- každá bunka so štyrmi alebo viac susedmi hynie následkom preľudnenia

```
   5 REM GENERACIA BUNIEK
  10 DIM a(145): DIM l(145): DIM e(8)
  15 LET g=0
  20 FOR t=1 TO 8: READ z: LET e(t)=z: NEXT t
  30 LET c=CODE "o": LET z=128
  35 BORDER 0: PAPER 0: CLS
  40 FOR b=1 TO 12
  50 FOR d=1 TO 12
  60 LET a(b+10*d)=z
  70 IF RND>.45 THEN LET a(b+10*d)=c
  80 LET l(b+10*d)=a(b+10*d)
  90 NEXT d: NEXT b
 100 LET g=g+1
 120 FOR u=1 TO 12
 130 FOR b=1 TO 12
 140 LET f=u+10*B
 180 IF g=1 THEN GO TO 250
 190 LET h=1
 200 FOR t=1 TO 8
 210 IF a(f+e(t)+1)=c THEN LET h=h+1
 220 NEXT t
 230 IF a(f)=c AND h<>3 AND h<>2 THEN LET l(f)=z
 235 IF a(f)=z AND h=3 THEN LET l(f)=c
```

```
240 NEXT b: BORDER RND*7: NEXT u
245 BORDER 0
250 FOR m=11 TO 144: LET a(m)=1(m): BEEP .005,m/3: NEXT m
255 PRINT AT 5,0;
260 FOR u-1 TO 12: PRINT TAB 4;
270 FOR b=1 TO 12: LET f=u+10*b
280 PRINT INK 6;CHR$ a(f);: NEXT b: PRINT;: NEXT u
285 PRINT AT 1,4; PAPER 2; INK 6;"GENERACIA ";g: BEEP .3,50
290 GO TO 100
300 DATA 11,10,9,1,-1,-9,-10,-11
```

K O L Ó N I A B U N I E K

```
 10 REM KOLONIA BUNIEK
 20 GO SUB 90
 30 LET zobrazenie=200
 40 LET rast=300
 50 GO SUB zobrazenie
 60 GO SUB rast
 70 GO TO 50
 80 REM Hlavny program
 90 REM -------------------------------------------------------
100 CLS
110 LET bunky=0
120 DIM a(11,11): DIM b(11,11)
130 FOR x=2 TO 10: FOR y=2 TO 10
135 BORDER RND*7
140 IF RND>.35 THEN LET a(x,y)=1: BEEP .02,x*y/2: LET bunky=bunky+1
150 LET b(x,y)=a(x,y)
160 NEXT y: NEXT x
170 LET rok=0
175 BORDER 7: CLS
180 RETURN
200 REM Zobrazenie kolonie
210 REM -------------------------------------------------------
220 LET rok=rok+1
230 BEEP .02,RND*20
250 PRINT AT 19,0; INK RND*6;"GENERACIA ";rok;AT 3,8;
255 LET bunky=0
260 FOR x=2 TO 10: FOR y=2 TO 10
270 LET a(x,y)=b(x,y)
280 IF a(x,y)=0 THEN PRINT " ";
285 IF a(x,y)=1 THEN PRINT INK RND*6;"O";: LET bunky=bunky+1
290 NEXT y: PRINT TAB 8;: NEXT x
292 PRINT AT 21,0; INK RND*6;"Pocet buniek: ";bunky;" "
293 IF bunky<6 THEN RUN
295 RETURN
300 REM Rast kolonie
310 REM -------------------------------------------------------
340 FOR x=2 TO 10: FOR y=2 TO 10
345 BORDER RND*6
350 LET c=0
360 IF a(x-1,y-1)=1 THEN LET c=c+1
370 IF a(x-1,y)=1 THEN LET c=c+1
```

```
380 IF a(x-1,y+1)=1 THEN LET c=c+1
390 IF a(x,y-1)=1 THEN LET c=c+1
400 IF a(x,y+1)=1 THEN LET c=c+1
410 IF a(x+1,y-1)=1 THEN LET c=c+1
420 IF a(x+1,y)=1 THEN LET c=c+1
430 IF a(x+1,y+1)=1 THEN LET c=c+1
440 IF a(x,y)=1 AND c<>2 AND c<>3 THEN LET b(x,y)=0
450 IF a(x,y)=0 AND c=3 THEN LET b(x,y)=1
460 NEXT y: NEXT x
470 RETURN
```

Z Á P A L K Y

Táto jednoduchá hra na trénovanie matematických schopností je Vám určite dobre známa. Dvaja hráči (v našom prípade jeden hráč a počítač) berú striedavo z kôpky zapaliek niekoľko zápaliek. Minimálne množstvo je jedna zápalka, maximálne množstvo sa stanoví predom (v našom prípade sa pri každej hre mení). Prehráva ten hráč, na ktorého ostane posledná zápalka. Keďže je počítač neomylný, asi by Vás to po chvíli hry prestalo baviť. Preto je program zostavený takým spôsobom, aby občas zvíťazil aj človek. Skúste tento program upraviť tak, aby bolo možné zvoliť si viacej stupňov obtiažnosti.

```
  5 REM ZAPALKY
 10 REM Biely text na modrom podklade
 15 BORDER 1: PAPER 1: INK 7: CLS
 20 LET e=0: LET z=16+INT (RND*9)
 30 IF 2*(z/2)-z THEN LET z=z+1
 40 LET h=INT (RND*4)+2
 50 PRINT INK 6;AT 0,0;"Zobrat mozes maximalne ";h;" zapalky"
 60 IF e>0 THEN PRINT AT 6,2;"Zobral si ";e;TAB 20;"Ja beriem ";q
 70 PRINT AT 3,0;: FOR K=1 TO z: BEEP .01,k
 80 PRINT "|";
 90 NEXT k
100 LET k=7: IF RND>.5 THEN LET k=4
110 INPUT "Kolko zapaliek beries? ";e
120 IF e>z OR e>h OR e<1 THEN BEEP .5,1: GO TO 110
130 CLS: LET z=z-e
140 IF z=0 THEN FOR f=1 TO 20: BORDER RND*7: PRINT PAPER RND*5;
    AT 10,10;"ZVITAZIL SOM !": BEEP .05,RND*30+30: NEXT f: RUN
150 LET q=z-1-INT ((z-1)/(h+1))*(h+1)+INT (RND*3)-1
160 IF>q>z OR q<1 OR q>h OR (z=5 AND q=z) OR (z=4 AND q=z) OR
    (z=3 AND q=z) OR (z=2 AND q=z) THEN GO TO 150
170 LET z=z-q
180 IF z=0 THEN FOR f=1 TO 20: BORDER RND*7: PRINT PAPER RND*5;
    AT 10,10;"ZVITAZIL SI !": BEEP .05,RND*30+30: NEXT f: RUN
190 GO TO 50
```

H R A C Í A U T O M A T

Nasledujúci program simuluje hrací automat "Fruit machine", nazývaný tiež jednoruký pirát. Obsluha hracieho automatu je jednoduchá. Stlačením ľubovoľnej klávesy spustíte otáčanie bubna so znakmi. Opätovným stlačením

klavesy bubon zastavite. Cieľom je dosiahnuť aspoň tri rovnaké znaky vedľa seba. Začínate s hotovosťou 100 $.

```
  20 POKE 23609,100
  30 GO SUB 9000
  40 POKE 23692,-1
  50 BORDER 0: PAPER 0: INK 7: CLS
  60 PRINT AT 10,2; INK 5;"Kolo cislo: ";kolo;AT 12,2; INK 4;"Tvoja
     hotovost: ";dol;" $";AT 21,3; INK 7;"Stlac lubovolnu klavesu !"
  70 IF INKEY$<>"" THEN GO TO 70
  80 IF INKEY$ ="" THEN GO TO 80
  90 FOR g=1 TO 50: BORDER RND*7: BEEP .01,50-g: NEXT g: BORDER 0
 100 FOR j=1 TO 4
 110 IF m(j)=j THEN GO TO 150
 120 LET a(j)=INT (RND*4)+1
 150 NEXT j
 155 LET kolo=kolo+1
 160 PRINT AT 21,1; INK 3;"Stlac klavesu pre zastavenie !"
 165 PRINT AT 1,25; INK 7;"=*$o";AT 1,25; INK 4;"$o**";AT 1,25; INK 6;
     "$$$#";AT 1,25; INK 5;"*o=$": IF INKEY$="" THEN GO TO 165
 170 GO SUB 5000
 175 GO SUB 4000
 180 IF dol>0 THEN GO TO 50
 190 PRINT AT 12,2; INK 4;"Tvoja hotovost: 0 $ "
 200 PRINT AT 15,2; INK 3;"+++ PREHRAL SI VSETKO +++"
 210 GO TO 6000
4000 REM Peniaze
4010 IF a(1)=a(2) AND a(2)=a(3) AND a(3)=a(4) THEN PRINT AT 15,2;
     INK 3;"*** PREMIA ***": GO SUB 7000: LET dol=dol+100: GO TO 4100
4020 IF a(1)=a(2) AND a(2)=a(3) OR a(2)=a(3) AND a(3)=a(4) THEN PRINT
     AT 15,2; INK 6;"*** VYHRAL SI ***": GO SUB 8000: LET dol=dol+50:
     GO TO 4100
4030 LET dol=dol-10
4100 DIM m(4): RETURN
5000 REM Valec
5020 PRINT AT 1,25;
5030 FOR j=1 TO 4
5040 IF a(j)=1 THEN PRINT INK 4;"$";: BEEP .1,10
5050 IF a(j)=2 THEN PRINT INK 5;"o";: BEEP .1,20
5060 IF a(j)=3 THEN PRINT INK 6;"*";: BEEP .1,30
5070 IF a(j)=4 THEN PRINT INK 7;"#";: BEEP .1,40
5080 PAUSE 70
5090 NEXT j
5100 RETURN
6000 FOR f=1 TO 20: FOR g=1 TO 7: BORDER g: BEEP .01,g: NEXT g: NEXT f
6010 BORDER 0: PRINT AT 21,0;"Chces tuto hru hrat znova? (a/n)"
6020 IF INKEY$="" THEN GO TO 6020
6030 IF INKEY$="A" OR INKEY$="a" THEN RUN
6040 STOP
7000 FOR f=1 TO 20: FOR g=1 TO 7: BORDER g: BEEP .01,20+g: NEXT g:
     NEXT f: RETURN
8000 FOR f=1 TO 20: FOR g=1 TO 7: BORDER g: BEEP .01,30+g: NEXT g:
     NEXT f: RETURN
9000 REM Nastavenie premennych
9010 DIM a(5): DIM m(5)
```

```
9020 LET dol=100
9030 LET kolo=1
9040 BORDER 0: PAPER 0: CLS
9050 RETURN
```

B Ú R A N I E S T E N Y

V tejto hre, ktora je jednoduchou verziou hry "The wall" (stena), sa snažite pomocou loptičky, ktorú odrážate palkou, rozbit všetky tehličky v stene. Palku ovladate pomocou klaves o - vlavo, p - vpravo. Na začiatku hry si môžete nastavit obtiažnosť v rozmedzi 1-20.

```
  5 REM BURANIE STENY
 10 POKE 23609,100: GO SUB 500
 15 BORDER 2: PAPER 6: CLS
 20 FOR f=0 TO 7
 30 PRINT AT f,8; INK RND*3;"##############"
 40 NEXT f
 55 LET s=0
 60 LET x=16
 70 LET b=INT (RND*10)+8
 80 LET q-1-INT (RND*3)
 90 IF q=0 THEN GO TO 80
100 FOR p=8 TO -8 STEP -1
105 IF x>23 OR x<8 THEN LET x=x
110 IF ABS b>23 THEN LET b=23
115 LET x=x+(INKEY$="p" AND x<>22)-(INKEY$="o" AND x<>8)
120 PRINT AT 9,x-2; INK 2;"  ==  ";AT p,b; INK 1;"o"
140 LET b1=b
150 IF b=8 OR b=23 THEN LET q=-q: BEEP .005,30
160 LET b=b+q
170 LET x=x+(INKEY$="p" AND x<>22)-(INKEY$="o" AND x<>8)
175 FOR g=1 TO z: NEXT g
180 PRINT AT p,b1;" "
190 NEXT p
195 LET x=x+(INKEY$="p" AND x<>22)-(INKEY$="o" AND x<>8)
200 IF ABS (b-x)>2 THEN GO TO 230
205 LET s=s+10
210 PRINT AT 16,2; INK 1;"SKORE: ";s;"  ": BEEP .25,1
220 GO TO 80
230 BEEP .5,-30: FOR g=1 TO 400: NEXT g
240 LET s=0
250 GO TO 15
500 INPUT "Zvol si obtiaznost (1-20) ";z
510 IF z <1 OR z>20 THEN GO TO 500
520 LET z=ABS (z-20): LET z=2*z
530 FOR s=1 TO z
540 BEEP .01,s
550 NEXT s
560 RETURN
```

G A L A X I A

A ako poslednú hru Vam ukážeme program galaxia. Je to namet ako robit zaujímavé hry so sci-fi tematikov. Raketova rampa sa ovláda pomocou klaves o - vľavo, p - vpravo, z - vystrelenie rakety.

```
  5 REM GALAXIA
 10 GO SUB 1000
 15 BORDER 0: PAPER 0: INK 7: CLS
 20 LET x=10: LET s=0
 30 LET p=0
 40 LET s=s+p
 50 CLS
 60 PRINT AT 0,20;"SKORE: ";s;"  "
 65 LET alien=INT (RND*3)+1
 70 LET p=0
 80 LET n=10
 90 LET y=13
100 LET c=0
110 PRINT AT 15,x;" "
120 IF INKEY$="o" AND x>0 THEN LET x=x-1
130 IF INKEY$="p" AND x<31 THEN LET x=x+1
140 PRINT AT 15,x; INK 7;a$
145 BEEP .004,10
150 PRINT AT p,n;" "
160 IF RND>.65 THEN LET n=n+INT (RND*3-1)
165 IF RND>.8 THEN LET p=p+1
170 REM Pismeno A v dalsom riadku je v mode GRAPH !
175 IF alien=1 THEN PRINT AT p,n; INK 4;"A"
180 REM Pismeno B v dalsom riadku je v mode GRAPH !
185 IF alien=2 THEN PRINT AT p,n; INK 5;"B"
190 REM Pismeno C v dalsom riadku je v mode GRAPH !
195 IF alien=3 THEN PRINT AT p,n; INK 6;"C"
200 IF p>14 THEN PRINT AT p,n;" ";AT y,c;" ":GO TO 60
210 IF c=0 AND INKEY$ ="z" THEN LET c=x
220 IF c=0 THEN GO TO 110
230 PRINT AT y,c;" "
240 LET y=y-1
250 IF y<0 THEN GO TO 90
255 REM Pismeno M v dalsom riadku je v mode GRAPH !
260 PRINT AT y,c; INK 3;"M": BEEP .009,50
265 REM Pismeno A v dalsom riadku je v mode GRAPH !
270 IF c=n AND y=p THEN FOR w=1 TO 10: PRINT AT y,c; INK 2; FLASH 1;
    "A": BEEP .01,-40: BEEP .03,-20: BEEP .08,-30: NEXT w: GO TO 40
280 GO TO 110
990 REM Pismeno E v dalsom riadku je v mode GRAPH !
1000 LET a$="E"
1005 FOR j=0 TO 7
1010 READ n
1020 POKE USR "A"+j,n
1030 NEXT j
1040 FOR j=0 TO 7
1050 READ n
1060 POKE USR "B"+j,n
1070 NEXT j
```

```
1080 FOR j=0 TO 7
1090 READ n
1100 POKE USR "C"+j,n
1110 NEXT j
1120 FOR j=0 TO 7
1130 READ n
1140 POKE USR "M"+j,n
1150 NEXT j
1160 FOR j=0 TO 7
1170 READ n
1180 POKE USR "E"+j,n
1190 NEXT j
2000 DATA BIN 11000011, BIN 00111100, BIN 11011011, BIN 11111111,
     BIN 00111100, BIN 00011000, BIN 00100100, BIN 11000011
2010 DATA BIN 01000010, BIN 00100100, BIN 10111101, BIN 10100101,
     BIN 11111111, BIN 00111100, BIN 00100100, BIN 11100111
2020 DATA BIN 01111110, BIN 01011010, BIN 01111110, BIN 00111100,
     BIN 00011000, BIN 11100111, BIN 10000001, BIN 11000011
2030 DATA BIN 00000000, BIN 00010000, BIN 00010000, BIN 00111000,
     BIN 00111000, BIN 00111000, BIN 01111100, BIN 00101000
2040 DATA BIN 01010100, BIN 01111100, BIN 01010100, BIN 01111100,
     BIN 11010110, BIN 10000010, BIN 10101010, BIN 11111110
5000 RETURN
```

P R E V O D

Ak radšej pracujete pri definovaní grafických znakov s dekadickými číslami ako s binárnymi, pomôže Vám nasledujúci program.

```
  10 FOR b=TO 225
  20 LET j=b
  30 LET a$=""
  40 FOR n=0 TO 7
  50 LET t=j-INT (j/2)*2
  60 IF t=0 THEN LET a$="0"+a$
  70 IF t<>0 THEN LET a$="1"+a$
  80 LET j=INT (j/2)
  90 NEXT n
 100 PRINT b,a$
 110 NEXT b
```

ZX SPECTRUM BEZ HRANÍC
(VIAC AKO 100 PROGRAMOV A PODPROGRAMOV)

Vydal: ULTRASOFT, spol. s r.o.
poštový priečinok, pošta 29
826 07 BRATISLAVA

V roku 1991 ako svoju 3. publikáciu
2. vydanie v roku 1992, náklad 1000 kusov

ISBN 80-85440-02-4