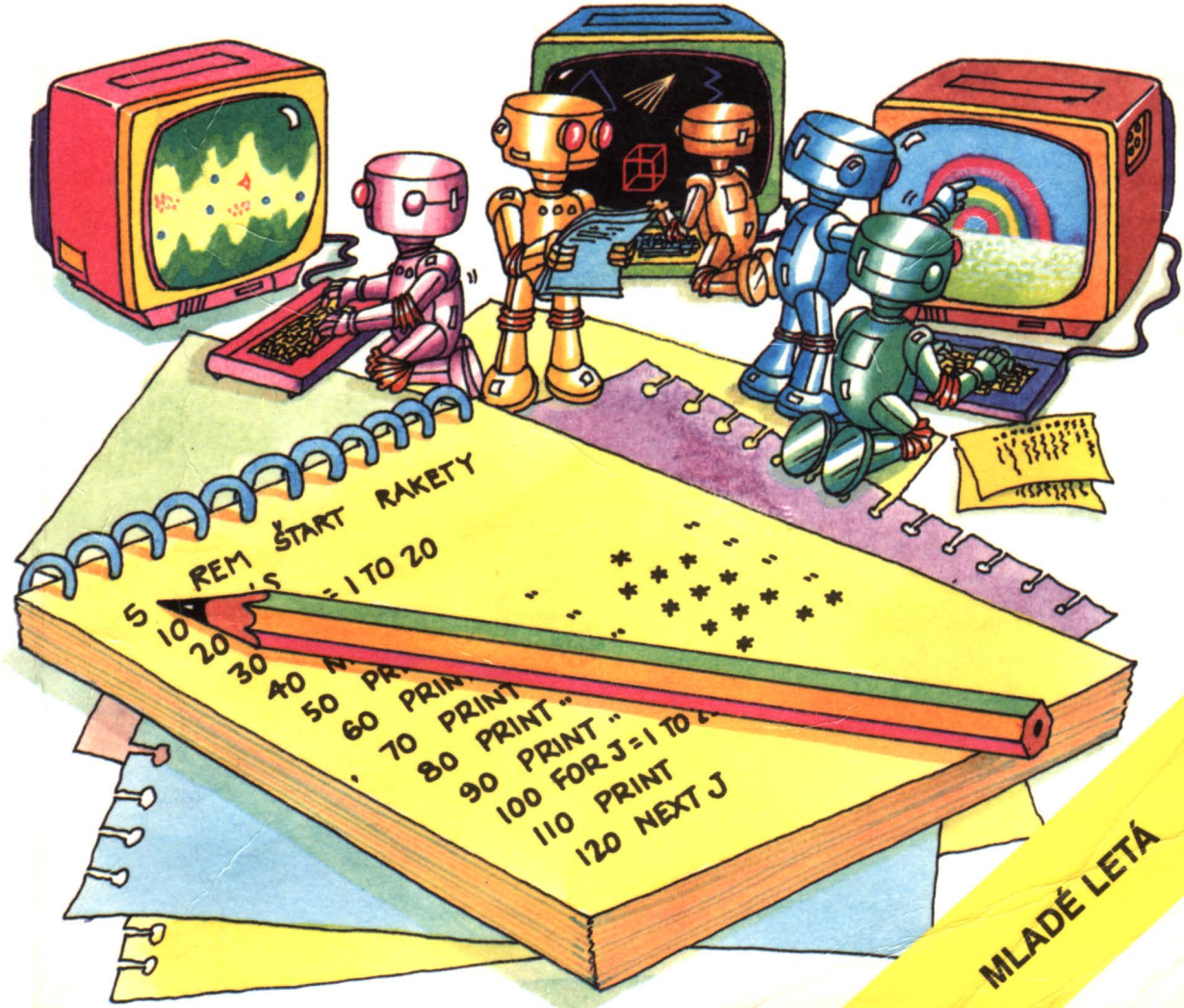
SKÚSIŠ PROGRAMOVAŤ?

BASIC A STROJOVÝ KÓD



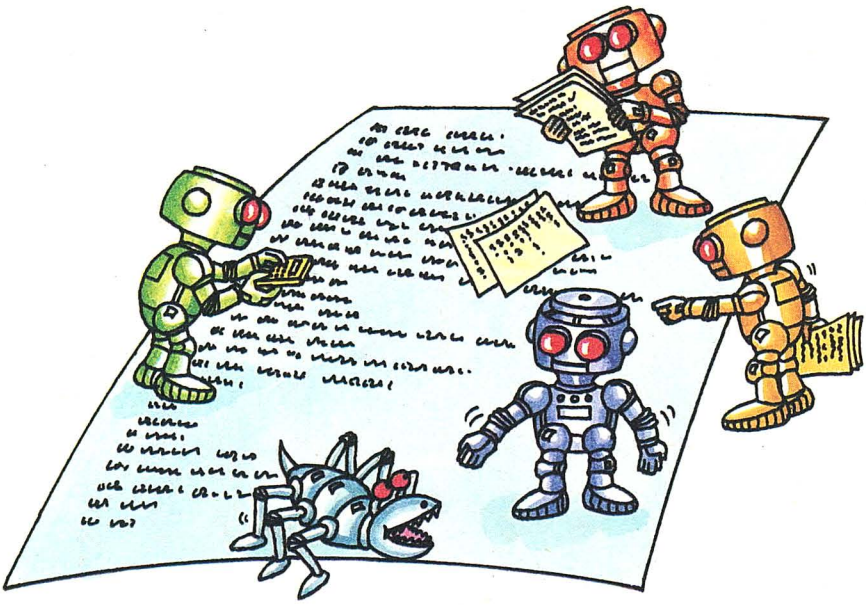
MLADÉ LETÁ

SKÚSIŠ PROGRAMOVAŤ?

BASIC A STROJOVÝ KÓD

Lisa Wattsová a Gaby Waters
Ilustrovali Graham Round a Naomi Reed

Odborní poradcovia pre basic B. Reffin Smith,
N. Cutler a M. Wharton
Odborní poradcovia pre strojový kód A. P. Stephenson
a C. Oxlade
Preložili Ivana Štefáneková a Peter Tomcsányi



MLADÉ LETÁ

Obsah

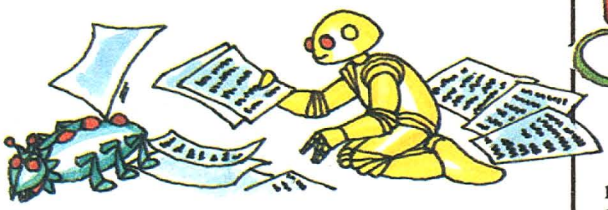
4 Niečo na úvod
5 Prehľad príkazov basicu
10 Aký basic používame
12 Píšeme jednoduché programy
14 Príkaz PRINT
16 Premenné
18 Opakovanie činností
20 Cykly
22 Príkaz IF/THEN
24 Náhodné čísla
26 Pracujeme so znakmi
29 Šifrujeme správu
30 Príkaz INKEY$
31 Ako napísať program Autoškola
32 Používame DATA
34 Používame polia
36 Podprogramy
37 Ovocná lotéria
38 Študujeme programy v basicu
40 Rady, ako študovať programy
42 Hra so slovami
44 Pristátie medziplanetárnej lode
46 Banka údajov o majstrovstvách sveta vo futbale
54 Jednoduchá grafika
58 Triedenie
64 Kreslenie grafov
66 Konverzačný program
74 Ako napísať program Poklad
82 Úvod do strojového kódu
84 Čo je strojový kód
86 Spoznaj svoj počítač
88 Pamäť počítača
91 Čísla v šestnástkovej sústave
92 Príkazy PEEK a POKE
94 Základná jednotka zvnútra
96 Programujeme základnú jednotku
98 Prekladáme program
100 Ako nájsť voľnú RAM
103 Ako uložiť a spustiť program
107 Vyberáme z pamäte a ukladáme do nej
108 Pracujeme s veľkými číslami
109 Podmienka prenosu
110 Sčítanie veľkých čísel
112 Píšeme na obrazovku
116 Skoky a vetvenie
120 Program na blikanie obrazovky
122 Ako ďalej
124 Tabuľky prevodov medzi desiatkovou a šestnástkovou sústavou
125 Skratky a kódy inštrukcií pre Z80
128 Skratky a kódy inštrukcií pre 6502
130 Slovník strojového kódu
132 Riešenia hlavolamov
145 Zhrnutie basicu
147 Tabuľka rozdielov basicu
150 Register

© Usborne Publishing 1984
Translation © Ivana Štefáneková a Peter Tomcsányi 1991

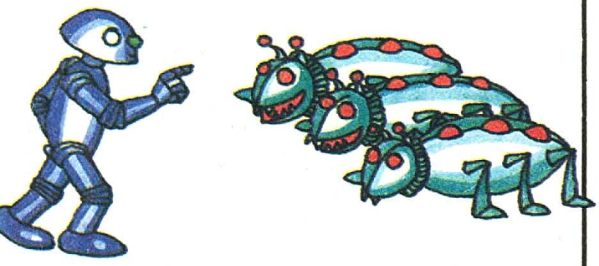
ISBN 80-06-00178-2

O knihe

Kniha Skúsiš programovať? ťa oboznámi so základmi basicu a strojového kódu. Má tri samostatné časti. V prvej nájdeš množstvo cvičení, hlavolamov a návrhov na vytváranie vlastných programov, ktoré ti pomôžu porozumieť príkazom basicu. V druhej časti nájdeš dlhšie programy, pomocou ktorých sa naučíš programovať zložitejšie úlohy, ako napríklad banky údajov, kreslenie grafov a obrázkov, triedenie údajov.

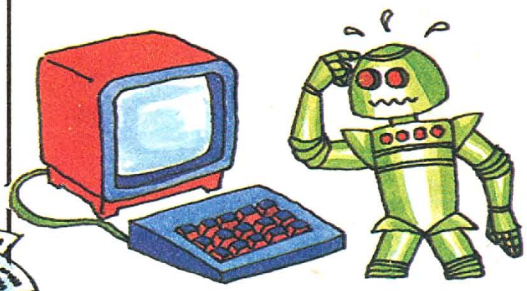


V tretej časti sa dozvieš základné informácie o strojovom kóde. Vysvetlíme ti, čo je to strojový kód a ako v ňom písať programy. Hoci je strojový kód zložitý a má mnoho pravidiel, v niektorých situáciách je oveľa vhodnejší než basic — vtedy, keď chceš vytvoriť rôzne efekty na obrazovke alebo zrýchliť hru.



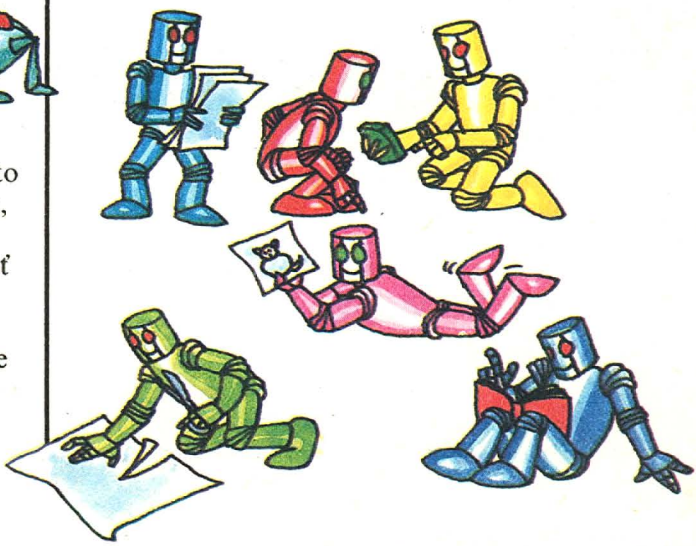
Všetky programy v basicu sú v tejto knihe napísané v bežnej verzii jazyka, a preto by mali pracovať na väčšine mikropočítačov. Možno budeš musieť zmeniť jeden alebo dva príkazy. Pomôže ti pri tom tabuľka na strane 147. Ak tvoj počítač nevie písať dĺžne a mäkčene, môžeš ich pokojne vynechať.

Programy v strojovom kóde sú písané pre počítače s mikroprocesorom Z80 alebo 6502. Mikroprocesor je integrovaný obvod, ktorý obsahuje základnú jednotku počítača. Počítače s rozdielnym mikroprocesorom



rozumejú rozdielnemu strojovému kódu. ZX Spectrum a ZX81 používajú mikroprocesor Z80, počítače Atari 800XL, Atari 130XE a Commodore používajú rôzne verzie mikroprocesora 6502. V československých počítačoch je mikroprocesor 8080, ktorý pozná veľa inštrukcií pre Z80.

Na konci knihy nájdeš riešenia cvičení a hlavolamov aj s krátkymi vysvetleniami. Nedaj sa znechutiť, ak niektorý z hlavolamov nevieš vyriešiť. Učíš sa aj tým, že si pozeráš riešenia. Ak je to možné, vyskúšaj si programy na počítači. Trochu ich zmeň a pozoruj, čo sa stane.

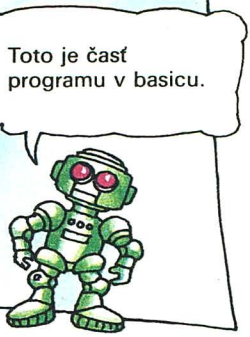


Niečo na úvod

Programovací jazyk basic sa skladá zo slov. Každé prikazuje počítaču vykonať nejakú činnosť. Ak chceš, aby počítač riešil zadanú úlohu, napíšeš mu postupnosť takýchto príkazov, nazývanú program. Program môže obsahovať iba slová v basicu a pri ich písaní musíš dodržiavať isté pravidlá.

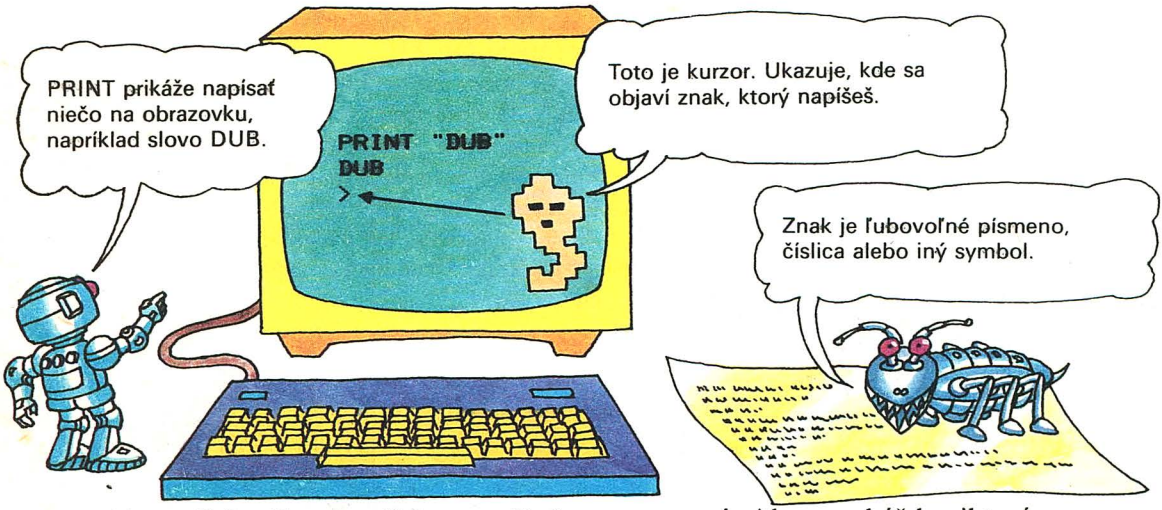
Každý riadok programu sa začína číslom. Čísla sa zvyknú zvyšovať o desať, a to preto, aby sa dali medzi riadky vsunúť ďalšie riadky a nemusel sa prečíslovať celý program.

```
10  GCLEAR
20  PRINT "ŠTART KOZMICKEJ LODE"
30  LET G=INT(RND(1)*20+1)
40  LET W=INT(RND(1)*40+1)
50  LET R=G*W
60  PRINT "GRAVITÁCIA=";G
70  PRINT "ZADAJ SILU"
80  FOR C=1 TO 10
90  INPUT F
100 IF F > R THEN PRINT "PRIVEĽA"
110 IF F < R THEN PRINT
120 IF F = R THEN GOTO
130 IF C <> 10 THEN PRINT
140 NEXT C
150 PRINT
160 PRINT "ŠTART
    SA NEVYDARIL"
170 PRINT "OHROZENIE
    ŽIVOTA"
180 STOP
190 PRINT "ÚSPEŠNÝ
    ŠTART"
```



Píšeme program

Pri písaní programu do počítača musíš na konci každého riadka stlačiť riadkovač (EOL alebo RETURN či ENTER — podľa typu počítača). Tým sa celý riadok uloží do pamäte počítača a počítač očakáva ďalší riadok. Keď dopíšeš všetky riadky, napíš RUN. Tým prikážeš počítaču, aby program vykonal.



Ak napíšeš príkaz bez čísla na začiatku riadka (hovoríme mu priamy príkaz), počítač ho vykoná hneď po stlačení riadkovača. Napríklad ak chceš, aby počítač na obrazovku vypísal tvoj program, napíšeš LIST ako priamy príkaz. Ak chceš, aby obrazovku vymazal, napíšeš GCLEAR alebo iný príkaz vhodný pre tvoj počítač.

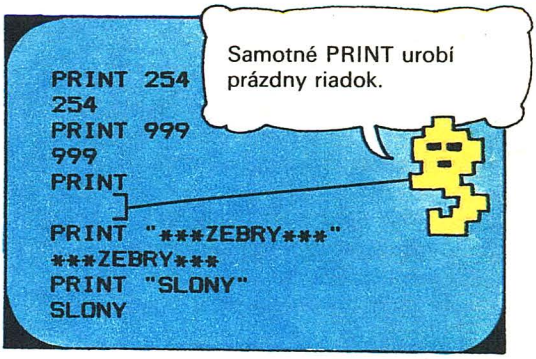
Pri písaní programu buď veľmi pozorný. Ak vynecháš hociktoré zo slov basicu alebo napíšeš nesprávne písmeno, číslicu či znamienko, počítač tento chybný príkaz nebude vedieť vykonať. Programátori chybám nelichotivo vravia chrobáky alebo ploštice (po anglicky bugs). Chyby však niekedy náhodou prinesú lepší alebo zaujímavejší výsledok, než sme pôvodne očakávali.

Prehľad príkazov basicu

Na nasledujúcich stranách je v krátkosti vysvetlený význam najdôležitejších príkazov basicu. Ak máš počítač, prečítaj si aj v jeho príručke, ako sa príkazy používajú, lebo na rôznych počítačoch sa môžu niektoré príkazy trochu odlišovať.

PRINT

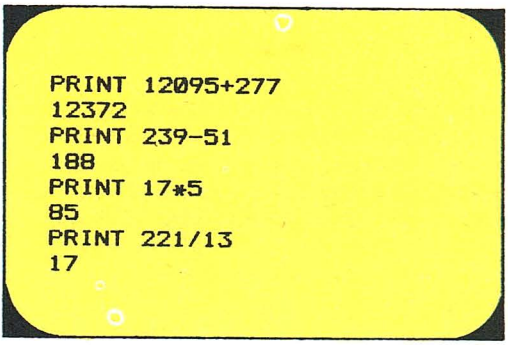
Prikáže počítaču napísať niečo na obrazovku. Ako vidíš na obrázku vpravo, písmená a symboly musia byť v úvodzovkách, čísla nie. Keďže na začiatku riadkov s príkazmi nie sú čísla, počítač vykoná príkazy hneď po stlačení riadkovača. Text v úvodzovkách zobrazí presne tak, ako si ho napísal, len bez úvodzoviek. Ak za PRINT nenapíšeš nič, vynechá riadok.

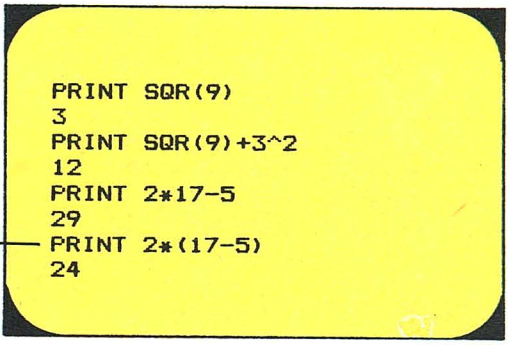


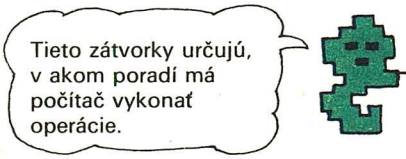
Výpočty

PRINT môže počítaču prikázať aj to, aby napísal výsledok výpočtu. Sčítanie a odčítanie sa označuje obvyklými znamienkami. Násobenie znamienkom *, delenie /. SQR(N) je príkaz na nájdenie druhej odmocniny čísla N. Symboly ^ alebo ** označujú umocňovanie. Napríklad 3^2 znamená druhú mocninu čísla 3.

Pri zložitejších výpočtoch počítač najprv násobí a delí, až potom sčituje a odčituje. Ak chceš, aby operácie vykonával v inom poradí, musíš výraz ozátvorkovať.

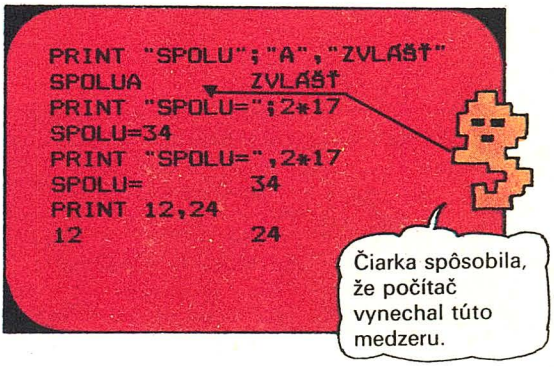






Čiarky a bodkočiarky

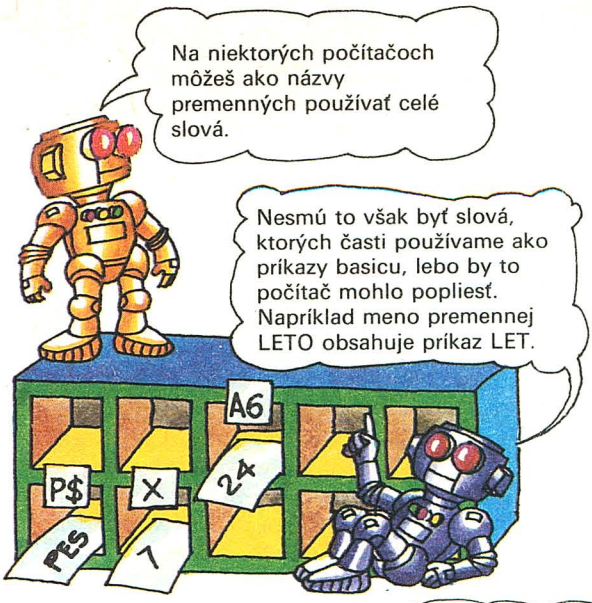
Pomocou čiarok a bodkočiarok môžeš počítaču povedať, kam má napísať nasledujúci znak. Čiarka hovorí, aby ho posunul do ďalšieho stĺpca (stĺpce mávajú šírku 13 až 15 znakov), bodkočiarka hovorí, aby ho napísal tam, kde prestal písať. Vyskúšaj si na počítači príklady z obrázka.



Premenné

Informácie, ktoré počítač spracúva, sa nazývajú údaje (dáta). Aby si mohol nájsť údaj, ktorý si vložil do pamäte, musíš ho označiť. Označené pamäťové miesto sa nazýva premenná, pretože v rôznom čase môže obsahovať rôzne hodnoty. Ak chceš pracovať s uloženým údajom, stačí napísať meno premennej, v ktorej je uložený.

Ako mená premenných môžeš používať písmená abecedy alebo písmená spolu s číslicami, napríklad A alebo A6. Ak má premenná obsahovať písmená alebo slová, pridáva sa značka dolára — $, napríklad P$ alebo P6$. Text je ako retiazka z písmen, preto hovoríme o reťazcoch a reťazcových premenných. Skontroluj si v príručke, či tieto pravidlá platia aj pre tvoj počítač.

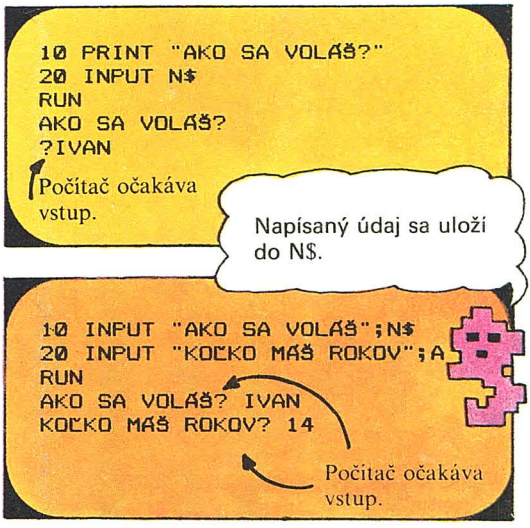


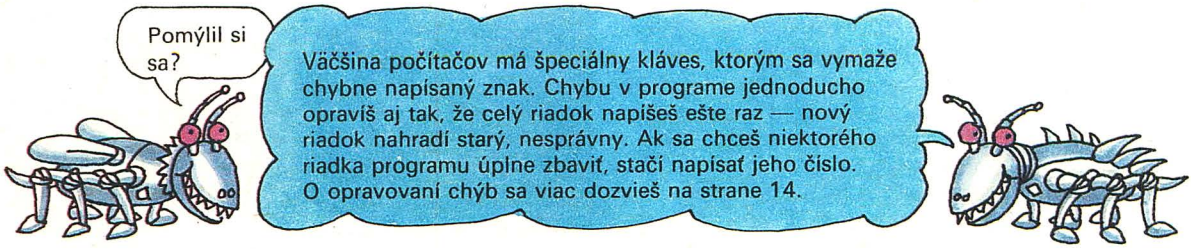
LET

Toto je jeden zo spôsobov, ako počítaču zadať údaje. LET A=3 mu prikáže na pamäťové miesto označené A uložiť číslo 3. LET C$="ZAJACE" mu prikáže na pamäťové miesto C$ uložiť reťazec ZAJACE. Reťazce musia byť vždy v úvodzovkách, čísla nie.

INPUT

Počítaču môžeš zadať údaj aj vtedy, keď program beží. INPUT sa používa spolu s menom premennej. Pri vykonávaní tohto príkazu počítač napíše otáznik (alebo zapípa) a čaká, kým niečo napíšeš. Ak za INPUT nasleduje číselná premenná, musíš napísať číslo, ak reťazcová premenná, počítač očakáva reťazec. Na väčšine počítačov môžeš za INPUT napísať do úvodzoviek slová, ktoré sa vypíšu ako otázka. Sprehľadní to program. Slová a meno premennej sa väčšinou oddeľujú bodkočiarkou.





READ/DATA

Príkazy READ a DATA ponúkajú iný spôsob zaznamenania údajov do pamäte. Za slovom READ nasleduje jedna alebo viac premenných. Údaje, ktoré má počítač uložiť do týchto premenných, sú za slovom DATA. Riadok DATA sa môže nachádzať kdekoľvek v programe. Ak počítač narazí na príkaz READ, nájde riadok s DATA a postupne ukladá údaje na určené pamäťové miesta, pričom zachováva poradie mien a príslušných hodnôt. Jednotlivé údaje musia byť oddelené čiarkou. Na niektorých počítačoch treba aj v riadkoch DATA dávať reťazce do úvodzoviek, na iných stačí úvodzovky použiť vtedy, keď reťazec obsahuje medzery alebo interpunkčné znamienka.

IF/THEN

Takto prikážeš počítaču, aby rozhodol, či je splnená zadaná podmienka, a podľa výsledku urobil rôzne činnosti. Môžeš napríklad porovnávať dva údaje — či sa rovnajú, nerovnajú alebo či jeden je väčší ako druhý. Na obrázku vidíš symboly, ktorými jednotlivé porovnania označujeme. Za THEN môže nasledovať ľubovoľný príkaz. Ak podmienka nie je splnená, počítač príkaz nevykoná a pokračuje ďalším riadkom.

GOTO

GOTO prikáže počítaču prejsť na iný riadok programu. Obyčajne sa používa spolu s príkazom IF/THEN na vetvenie programu. Pri používaní tohto príkazu dávaj pozor, aby nevznikol nekonečný cyklus — tak, ako je to na obrázku vpravo (riadok 85). Zacyklený program zastavíš klávesom STOP alebo BREAK (čítaj brejk, prerušenie).

GOSUB/RETURN

GOSUB prikáže vykonať podprogram. Slovo RETURN na konci podprogramu pošle počítač na príkaz za GOSUB. GOSUB bez RETURN môže spôsobiť chybu.

REM

REM je skratka od remark (čítaj rimark, poznámka). Používa sa na písanie poznámok a vysvetliviek.

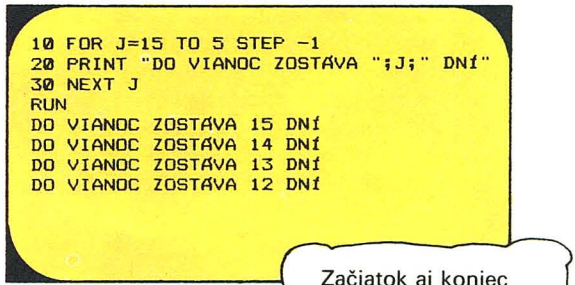
Príkaz v riadku 540 vráti počítač na príkaz, ktorý nasleduje za príkazom GOSUB.

Cyklus FOR/NEXT

Dvojica slov FOR a NEXT vytvára cyklus. Je to časť programu, ktorú má počítač vykonať toľkokrát, ako mu určíme. V našom príklade zopakuje cyklus v riadkoch 10–30 trikrát. Pri každom opakovaní napíše v riadku 20 správu. J je premenná, v ktorej ráta, koľkokrát už cyklus zopakoval. Riadok 30 vráti počítač späť na riadok 10, kde sa zvýši hodnota J o 1. Keď hodnota J prekročí 3, pokračuje riadkom 40.

STEP

Pomocou STEP môžeš meniť hodnotu premennej v cykle inak, než ju len zväčšovať o 1. Napríklad FOR J = 1 TO 10 STEP 2 znamená, že po zopakovaní cyklu sa zväčší hodnota J o 2. STEP X mení hodnotu premennej v cykle o hodnotu uloženú v X. V príklade vpravo je STEP −1, teda hodnota J sa znižuje.



Vnorené cykly

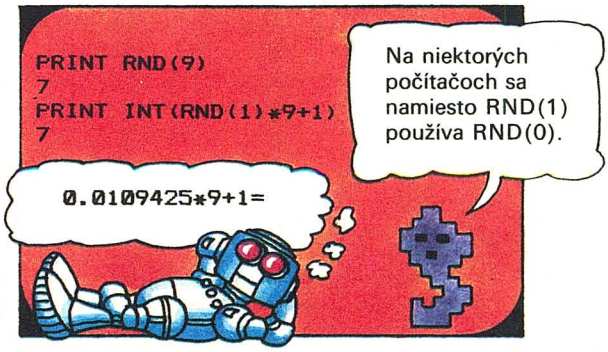
Ak použijeme cykly vnútri cyklov (hovorí sa im vnorené či vnútorné), môžeme robiť zložitejšie opakovania. V programe vpravo počítač pri každom vykonaní cyklu pre I (riadky 10 – 50) dvanásťkrát zopakuje cyklus pre J (riadky 20 – 40) a zakaždým napíše hodnotu súčinu J*I.

Grafické príkazy

Počítač kreslí tak, že na obrazovke rozsvecuje alebo zhasína malé plôšky — body. Príkazy na rozsvecovanie a zhasínanie bodov (grafické) sú na rôznych počítačoch rôzne. V knihe kreslíme body príkazom PLOT X, Y, kde X a Y sú súradnice bodov. Čiaru kreslíme príkazom DRAW X,Y. Väčšina počítačov používa podobné príkazy. Niektorým musíš navyše zadať príkaz určujúci, v ktorom grafickom režime chceš pracovať.*

RND

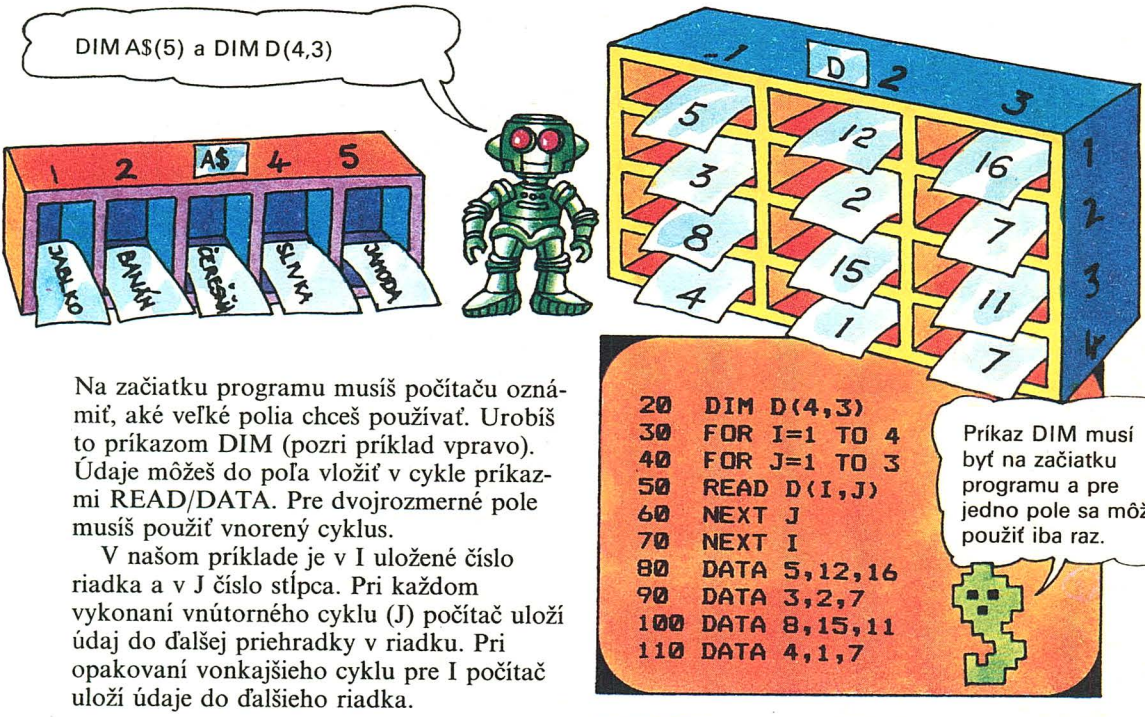
RND hovorí počítaču, aby náhodne vybral číslo. Presný tvar príkazu si pozri v príručke. Ak chceš náhodne vybrať celé číslo z intervalu 1 až 9, musíš na niektorých počítačoch napísať RND(9), na iných INT(RND(1)*9+1). RND(1) znamená, že počítač má náhodne vybrať číslo od 0 do 1. Číslo potom vynásobí dĺžkou intervalu (9) a k súčinu pripočíta dolnú hranicu intervalu (1). INT mu prikáže zmeniť výsledný súčet na celé číslo.



* Veľa počítačov má niekoľko grafických režimov. Líšia sa napríklad počtom farieb, ktoré možno používať.

Polia

Pole je postupnosť údajov uložených do pamäte pod jedným menom. Takúto premennú si predstav ako pamäťové miesto s viacerými priehradkami. Pole môže byť jednorozmerné (v pamäti mu zodpovedá jeden rad priehradiek) alebo dvojrozmerné (zodpovedá mu viac radov priehradiek). Na každý údaj jednorozmerného poľa sa môžeš odvolať číslom priehradky, v ktorej je uložený. Napríklad hodnota premennej A$(4) je reťazec SLIVKA. U dvojrozmerných polí musíš určiť číslo riadka a stĺpca, napríklad premenná D(3,2) má hodnotu 15. Čísla v zátvorkách sa nazývajú indexy. (Pravidlá, ako používať polia na počítačoch Sinclair, sú na stranách 34 a 147.)



Na začiatku programu musíš počítaču oznámiť, aké veľké polia chceš používať. Urobíš to príkazom DIM (pozri príklad vpravo). Údaje môžeš do poľa vložiť v cykle príkazmi READ/DATA. Pre dvojrozmerné pole musíš použiť vnorený cyklus.

V našom príklade je v I uložené číslo riadka a v J číslo stĺpca. Pri každom vykonaní vnútorného cyklu (J) počítač uloží údaj do ďalšej priehradky v riadku. Pri opakovaní vonkajšieho cyklu pre I počítač uloží údaje do ďalšieho riadka.

```
20  DIM D(4,3)
30  FOR I=1 TO 4
40  FOR J=1 TO 3
50  READ D(I,J)
60  NEXT J
70  NEXT I
80  DATA 5,12,16
90  DATA 3,2,7
100 DATA 8,15,11
110 DATA 4,1,7
```

Príkaz DIM musí byť na začiatku programu a pre jedno pole sa môže použiť iba raz.

LEFT$ a RIGHT$

Pomocou týchto slov môžeš narábať so znakmi reťazca. Napríklad LEFT$(A$,4) znamená: vezmi prvé štyri znaky zľava z premennej A$. RIGHT$(A$,5) znamená: vezmi posledných päť znakov z premennej A$. Počítače Sinclair tieto slová nepoznajú. Ako sa na nich pracuje so znakmi, dozvieš sa na stranách 26 a 147.

```
10 LET A$="DAŽDNIK"
20 PRINT LEFT$(A$,4)
25 PRINT LEFT$(A$,2)
30 PRINT
40 PRINT RIGHT$(A$,4)
RUN
DAŽD
DA

DNIK
```

MID$ a LEN

MID$ znamená: vezmi znaky zo stredu reťazca. Pomocou LEN počítač zistí dĺžku reťazca, teda počet znakov (vrátane medzier a interpunkčných znamienok). Napríklad MID$(K$,2,4) znamená: zober z reťazca K$ 4 znaky počnúc druhým. Na stranách 26 a 147 sa dozvieš, v akom tvare sa príkazy používajú na počítačoch Sinclair.

```
10 LET K$="POČASIE"
20 PRINT MID$(K$,3,3)
30 PRINT MID$(K$,4,3)
40 PRINT
50 PRINT LEN(K$)
RUN
ČAS
ASI

7
```

Toto je dĺžka reťazca K$.

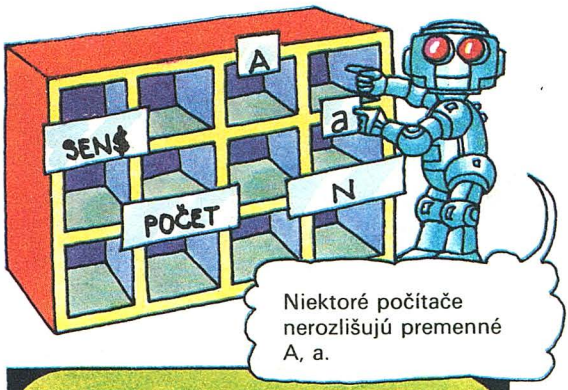
Aký basic používame

Všetky programy sú napísané v bežnom basicu. Pre niektoré počítače ich budeš musieť trochu upraviť. Niekoľko informácií o zvláštnostiach jednotlivých verzií basicu nájdeš na týchto dvoch stranách.

Programy sú napísané tak, aby fungovali na rôznych typoch počítačov. Pri ich zostavovaní sme nebrali do úvahy špeciálne vlastnosti ani jedného počítača. Keď pochopíš, ako programy pracujú, môžeš si ich prispôsobiť pre svoj počítač tak, aby sa stali rýchlejšími a prehľadnejšími.

Mená premenných

Na niektorých počítačoch premenné možno označiť celými slovami, na iných len písmenom alebo písmenom s číslicou. Napríklad na počítačoch Sinclair číselnú premennú možno označiť celým slovom, ale reťazcovú premennú len jedným písmenom. V našich programoch pre väčšiu prehľadnosť používame ako mená premenných celé slová. Ak to tvoj počítač nepripúšťa, použi len prvé písmeno slova.



LET

Na niektorých počítačoch nemusíš písať slovo LET, na iných ani slovo THEN v príkaze IF. V tejto knihe obe slová používame. Ak ich tvoj počítač nepotrebuje, pokojne ich vynechaj.

```
100 OVOCIE$="JABLKO"
110 POČET=POČET+1
120 IF POČET=10 PRINT "HOTOVO"
```

Počiatočné hodnoty premenných

Niektoré počítače vyžadujú, aby sme premennej pred použitím priradili nejakú hodnotu (pozri príklad). Iné pred použitím automaticky priradia číselným premenným nulu a reťazcovým prázdny reťazec. V našich programoch premenným priraďujeme počiatočnú hodnotu. Ak to na tvojom počítači netreba, tieto riadky vynechaj.

```
10  LET A=0
20  LET VETA$=""
.
.
.
.
100 LET VETA$=VETA$+"K"
110 LET A=A+1
```

"" označuje prázdny reťazec.

INPUT

Na väčšine počítačov môžeš za INPUT napísať slová v úvodzovkách.* Rozdiely na jednotlivých typoch počítačov sú v tom, či pred premennú musíš napísať bodkočiarku alebo nie, a či počítač automaticky vynechá alebo nevynechá medzeru medzi napísanými slovami a očakávaným vstupom. Pravidlá si over v príručke alebo zisti pokusmi.

```
10 INPUT "AKO SA VOLAŠ? ";N$
20 PRINT "AHOJ ";N$
```

Ak chceš, aby napísané slovo a vstup boli oddelené, na niektorých počítačoch musíš dať do úvodzoviek aj medzeru.

* Na počítačoch Commodore 64 a PMD 85 to však urobiť nemôžeš.

DATA

```
100 DATA MYŠ,MAČKA,PES
110 DATA "ŤAVA DVOJHRBÁ","BOCIAN BIELY"
120 DATA "ŽUBRIENKA, ŽABA","KUKLA, MOTÝĽ"
```

V úvodzovkách sú reťazce s medzerou alebo s interpunkčnými znamienkami.

Pri písaní údajov dávaj veľký pozor. Nezabudni, že musia byť navzájom oddelené čiarkou. Na niektorých počítačoch musíš dať všetky reťazce v riadkoch DATA do úvodzoviek, na iných treba dať do úvodzoviek len tie reťazce, v ktorých sú medzery alebo interpunkčné znamienka.

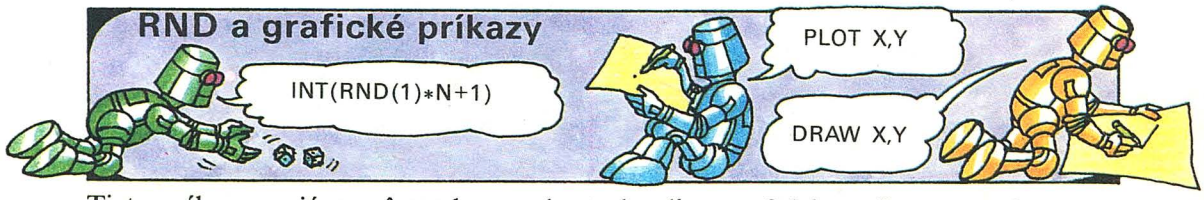
Niekoľko príkazov v jednom riadku

```
500 PLOT 40,1: DRAW 1,1
180 IF A = 10 THEN PRINT "SPRÁVNE": GOTO 100
```

Počítač vykoná príkaz len vtedy, ak A=10.

Väčšine počítačov nevadí, ak do jedného riadka napíšeš viac príkazov (príkazy sa oddeľujú dvojbodkou). Šetrí sa tým pamäť a program je prehľadnejší. Ak to však na tvojom počítači nie je možné, daj každý príkaz do nového riadka. Pozor, ak napíšeš viac príkazov do riadka začínajúceho sa IF, počítač vykoná príkazy len vtedy, ak je splnená podmienka za IF.

RND a grafické príkazy



Tieto príkazy majú na rôznych typoch počítačov rôzny tvar. V našich programoch vytvárame náhodné celé číslo od 1 do N (N je ľubovoľné číslo) príkazom INT(RND(1)*N+1). Bod kreslíme grafickým príkazom PLOT X, Y a čiaru príkazom DRAW X,Y. Ak treba, nahraď ich príkazmi, ktorým rozumie tvoj počítač, prípadne pridaj príkaz na voľbu grafického režimu.

Počítače Sinclair a Atari

Počítače Sinclair a Atari majú niektoré príkazy basicu iné ako väčšina ostatných počítačov (napríklad príkazy na prácu s reťazcami). Na strane 147 sa dozvieš, ako naše programy prispôsobiť pre počítače Sinclair. Príkazy, ktoré treba zmeniť, sú v programoch označené a ich tvar je buď hneď vedľa programu, alebo na strane 148.

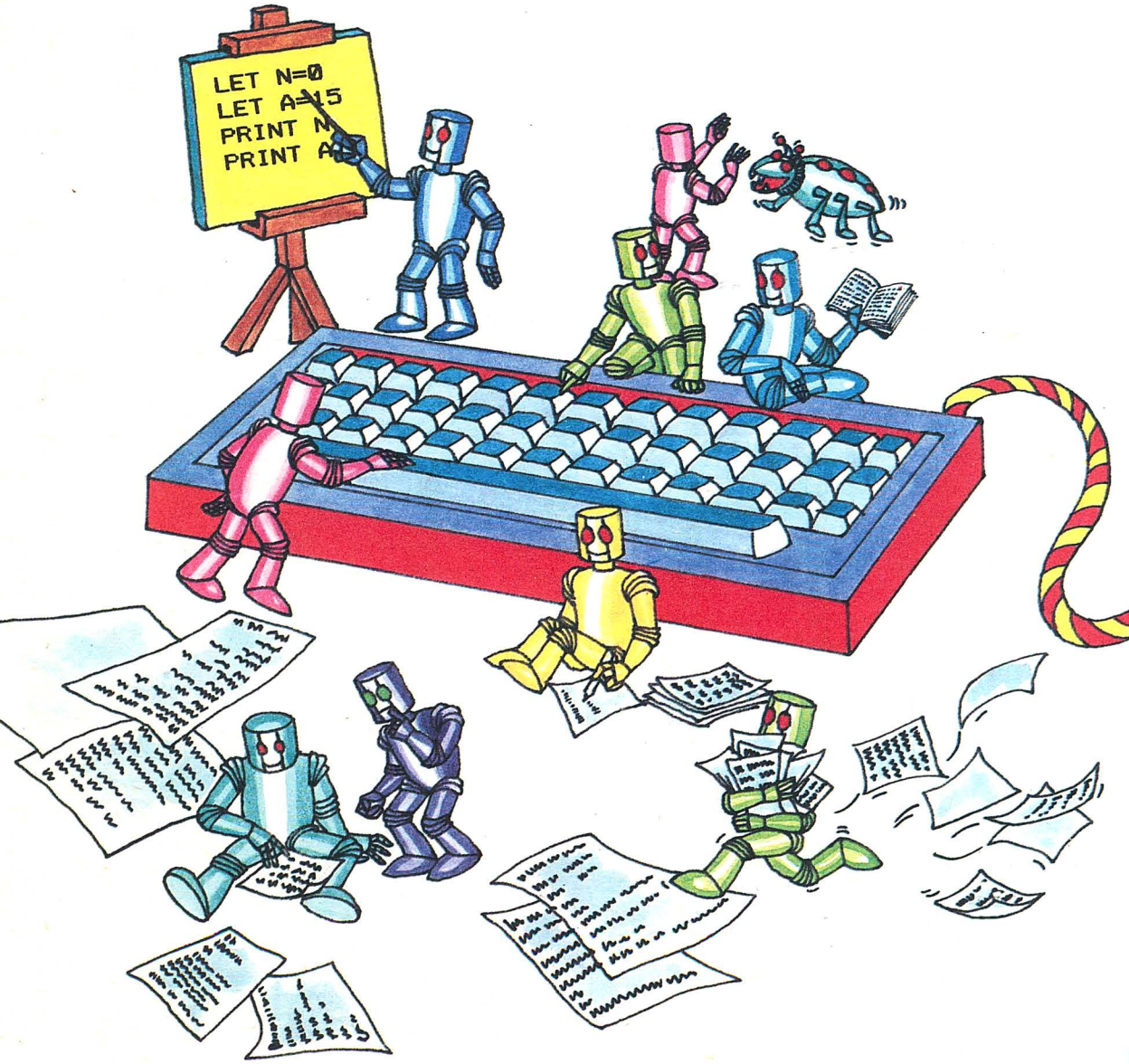
Atari 800XL a Atari 130XE tiež majú rozdielne príkazy pre prácu s reťazcami.

Zlepšovanie programov

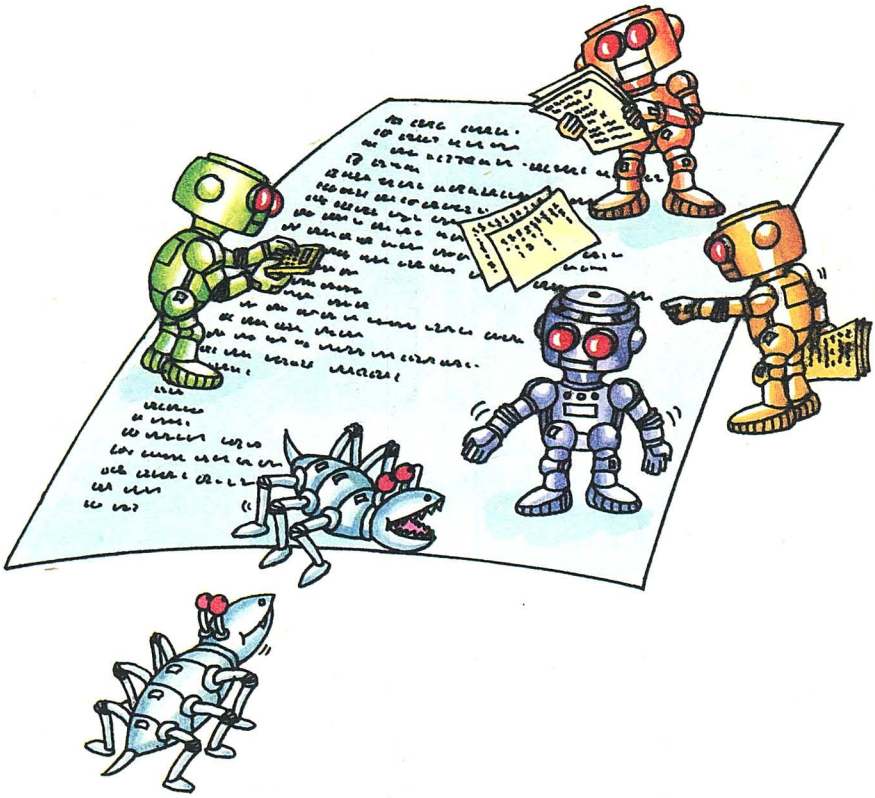
Keď program na tvojom počítači funguje a ty mu rozumieš, môžeš ho zmeniť a zlepšiť. Vlož doň iné údaje, ozvuč ho alebo pridaj príkazy na kreslenie farebných obrázkov.

Pri zmene príkazov nezabudni každý riadok pozorne skontrolovať. Ak pridávaš údaje, zmeň veľkosť cyklov a uprav príkazy DIM.

Píšeme jednoduché programy

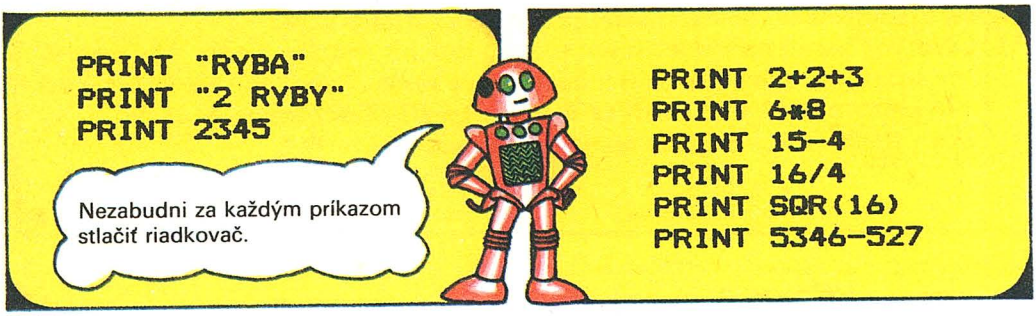


V tejto časti knihy je kopa cvičení, hlavolamov a zadaní programov, ktoré ti pomôžu pri učení basicu. Sú zoskupené do kapitoliek podľa príkazov, ktoré využívajú. Príkazy sú na začiatku každej kapitolky stručne vysvetlené. Ak zabudneš, ako sa niektorý používa, vráť sa k časti Prehľad príkazov basicu na začiatku knihy alebo pozri zoznam príkazov na stranách 145 – 146.
V niektorých hlavolamoch je tvojou úlohou doplniť neúplné riadky v programe (riadky sú označené hviezdičkou a miesto, kde niečo chýba, otáznikom). Nájdeš tu i programy plné chýb. Podarí sa ti chyby nájsť? (Upozorňuje ťa na ne ploštica.) Riešenia hlavolamov nájdeš na stranách 132 – 144.



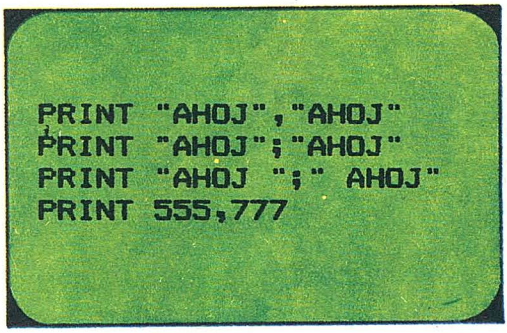
Príkaz PRINT

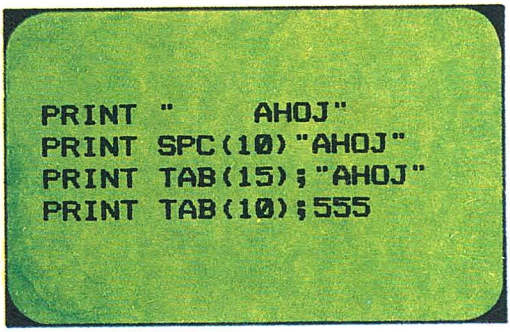
Na tejto dvojstrane nájdeš príklady, ako používať príkaz PRINT. Príkaz hovorí počítaču, že má niečo zobraziť na obrazovku. Môžeš ho používať ako priamy príkaz, teda riadok programu, v ktorom sa nachádza, nemusí mať číslo — vtedy počítač vykoná príkaz okamžite. Na konci priameho príkazu musíš stlačiť riadkovač (býva označený RETURN, ENTER, NEWLINE alebo EOL).



Skús tieto priame príkazy napísať do svojho počítača. Ak chceš, aby napísal číslice spolu s písmenami, musíš použiť úvodzovky. Samotné číslice nemusia byť v úvodzovkách.

PRINT sa dá použiť aj pri výpočtoch. Tu je niekoľko jednoduchých výpočtov, v ktorých sú matematické symboly basicu. Ak si si nie istý, čo znamenajú, odskúšaj si ich.

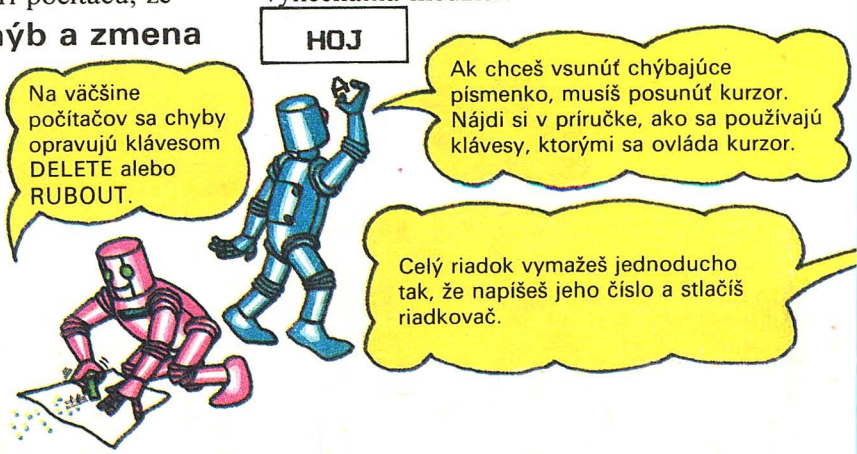




Príkazy na obrázku hovoria počítaču, kam má text napísať. Vyskúšaj ich, aby si videl, čo urobia. Interpunkčné znamienka majú v basicu zvláštny význam. Čiarka hovorí počítaču, že pred ďalším textom má vynechať niekoľko medzier. Bodkočiarka znamená, že nasledujúci text má napísať do toho istého riadka bez vynechania medzier.

Opravovanie chýb a zmena programu

Chyby vznikajú buď preto, že urobíš prieklep, alebo preto, že porušíš pravidlá basicu. Zisti si, ako sa na tvojom počítači opravujú chyby a menia programy.



Programátorské hlavolamy

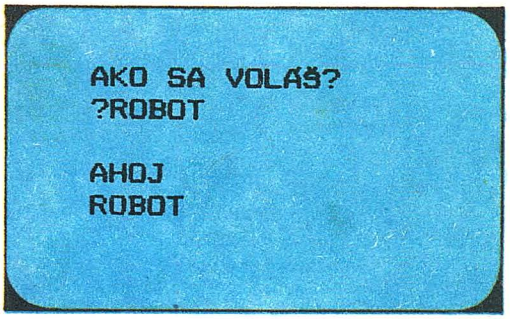
Odskúšaj si tento jednoduchý program. Čo počítač napíše na obrazovku, je na pravom obrázku. Skús program zmeniť tak, aby počítač písal správy v takých tvaroch, ako vidíš na dolných obrázkoch. Pomôž si priamymi príkazmi z predchádzajúcej strany. Ďalšie užitočné rady nájdeš nižšie.

```
10 PRINT "AKO SA VOLAŠ?"
20 INPUT A$
30 PRINT
40 PRINT "AHOJ"
50 PRINT A$
```

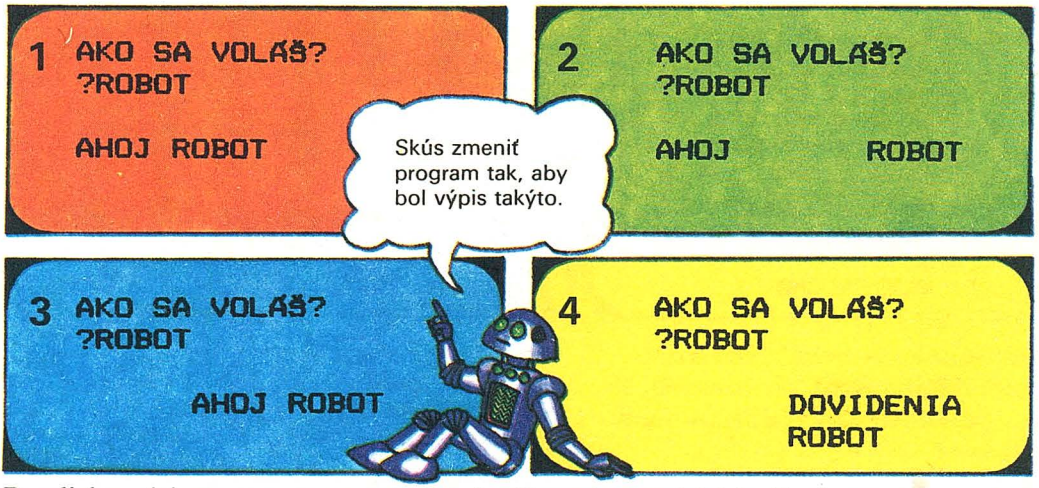
Počítač čaká, kým napíšeš svoje meno. Zaznamená ho v premennej A$.

Samotné PRINT prikáže urobiť prázdny riadok.

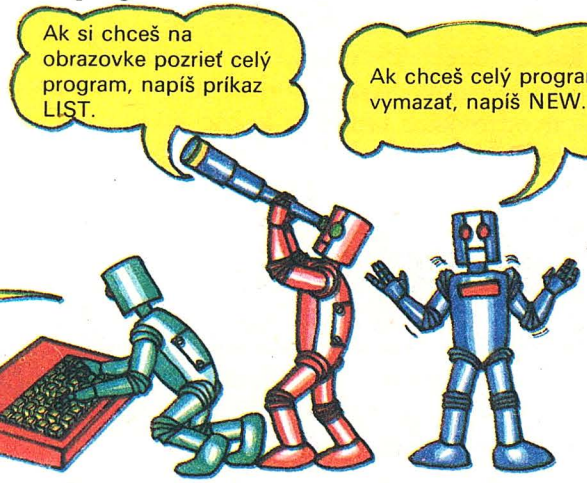
Počítač napíše slovo zaznamenané v premennej A$.



Program napíš do svojho počítača. Na konci riadka stlač riadkovač. Za programom napíš RUN alebo stlač riadkovač.



Rozdiel medzi programom a priamymi príkazmi z predchádzajúcej strany je v tom, že každý riadok príkazov v programe má číslo. Príkazy si počítač uloží do pamäte a vykoná ich, až keď napíšeš RUN. Ak zväčšuješ čísla riadkov po desať, môžeš pridať ďalší príkaz bez toho, aby si musel prečíslovať celý program.




```
10 PRINT "TOTO JE"
20 PRINT "PROGRAM"
15 PRINT "HLÚPY"
```

Nový riadok môžeš vložiť do programu kamkoľvek. Starý riadok opravíš tak, že ho prepíšeš. Skús si tento program napísať do počítača presne tak, ako je tu. Všimni si, čo sa stane, keď ho spustíš alebo keď ho dáš napísať na obrazovku.

Premenné

Premenná je označený úsek pamäte počítača, kam sa ukladá informácia. Pomocou slov LET a INPUT prikážeš počítaču, aby do premennej uložil konkrétny údaj tak, ako je to dolu na obrázkoch. Nezabudni, že údaj, ktorý pozostáva zo slov alebo písmen, číslic a iných symbolov, sa nazýva reťazec. Keď premennej priraďuješ reťazec, musíš ho dať do úvodzoviek.

```
10 LET A=16
20 LET HR$="HRDZAVÝCH ROBOTOV"
30 PRINT A
40 PRINT HR$

RUN
16
HRDZAVÝCH
ROBOTOV
```

A a HR$ sú mená premenných.

LET hovorí počítaču, že má vyhradiť a pomenovať pamäťové miesto a uložiť doň daný údaj.

```
10 PRINT "AKO SA VOLÁŠ?"
20 INPUT N$
30 PRINT "KOĽKO MÁŠ ROKOV?"
40 INPUT A
50 PRINT N$;" MÁ ";A
```

reťazcová premenná

INPUT spôsobí, že počítač vyhradí a pomenuje pamäťové miesto a čaká, kým na klávesnici napíšeš údaj. Potom program pokračuje ďalej.

Výber mena premennej

Ktoré z týchto mien sú na tvojom počítači prípustné?

```
LET B$="6422 MYŠI"
LET B5=6422
LET K1=99
LET BL$="BLCHA"
```

Niektoré z týchto mien premenných obsahujú slová basicu. Ktoré sú to?

```
LET LETO$="SLNKO"
LET DESAŤ=10
LET RUN$="KONIEC"
LET TOPOĽ$="1 STROM"
```

Pravidlá, ktoré určujú, čo možno používať ako mená premenných, sú pre jednotlivé typy počítačov veľmi odlišné.*

Ak ako mená premenných používaš krátke slová, nesmú to byť slová basicu. Počítač by to mohlo popliesť. Vyskúšaj si tieto štyri príkazy, aby si videl, ktorým tvoj počítač rozumie.

Slová a premenné

```
10 LET HR$="HRDZAVÝ ROBOT"
20 PRINT "AHOJ ";HR$
```

Namiesto bodkočiarky napíš čiarku a pozri sa, čo sa stane.

Ak v príkaze PRINT píšeš súčasne reťazce aj premenné, reťazce musia byť v úvodzovkách. Slová a premenné oddeľuj bodkočiarkou. Pri slove v úvodzovkách musíš na každej strane vynechať medzeru. Skús v tomto programe medzery nedať a uvidíš, čo sa stane.

```
10 LET HR$="HRDZAVÝ"
20 LET S=66
30 PRINT HR$;" ROBOT ZJE ";
40 PRINT S;" SAFALÁDOK"
```

Hlavolam s príkazom PRINT

```
LET A=66
LET B=77
LET HR$="HRDZAVÝCH ROBOTOV"
```

Skús napísať program s uvedenými premennými, ktorý po spustení prikáže počítaču napísať na obrazovku takýto text.

```
66 HRDZAVÝCH ROBOTOV ZJE
77 OPEKANÝCH SAFALÁDOK
```

*Ak máš ZX81, musíš zmeniť všetky viacpísmenkové mená reťazcových premenných v tejto knihe.

Pohľad do programu

Tento program mení stupne Fahrenheita na stupne Celzia. Ak nie si si celkom istý, ako pracuje alebo na čo slúžia jednotlivé premenné, vlož doň riadky s príkazom PRINT tak, aby sa na obrazovku písali obsahy premenných. Takto sa môžeš pozrieť „dovnútra" programu — lepšie ho pochopíš.

```
10 INPUT A
20 LET F=32
30 LET B=5
40 LET C=B/9
50 LET D=A-F
60 LET R=D*C
70 PRINT R
```

Stupne Fahrenheita uloží počítač do premennej A.

Skús pridať do programu ďalšie riadky, aby si zistil hodnoty premenných C a D.

Hlavolam Stupne Celzia a Fahrenheita

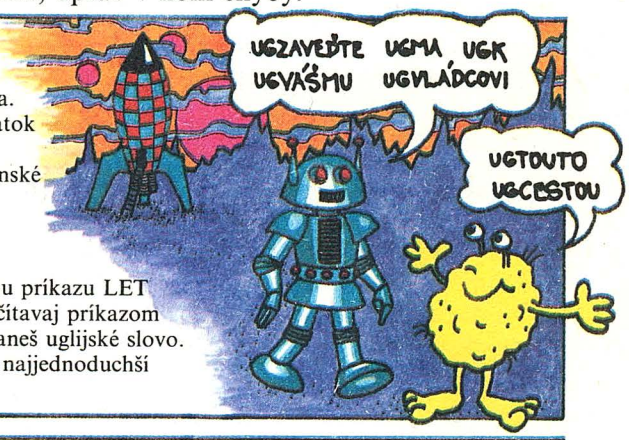
Pokús sa zmeniť program tak, aby menil stupne Celzia na stupne Fahrenheita.

Ďalšie programátorské hlavolamy

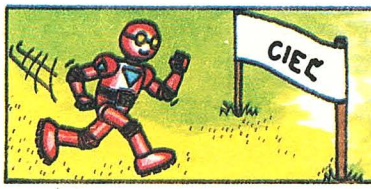
V nasledujúcich obdĺžnikoch nájdeš niekoľko tipov na jednoduché programy. Budeš potrebovať veľa premenných. Píš si poznámky o ich význame a o menách, ktoré si pre ne vybral. Je výhodné vybrať také meno, čo by ti pripomínalo, na čo premennú používaš alebo čo do nej ukladáš. Program najprv napíš na papier, potom ho prepíš do počítača, a ak treba, oprav v ňom chyby.

1 Hovoríš po uglijsky?

Ideš na výskumnú expedíciu na planétu Uglia. Uglijčania hovoria po slovensky, ale na začiatok každého slova pridávajú slabiku UG. Skús napísať program, ktorý bude prekladať slovenské slová do uglijčiny.

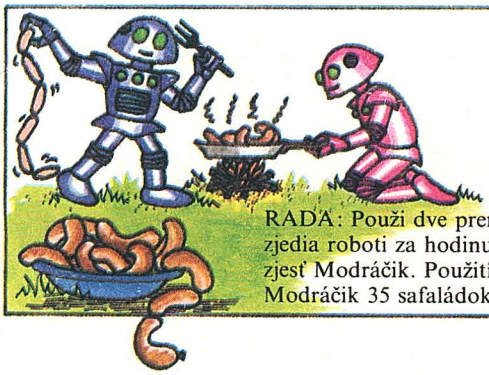


RADA: Písmená UG si zaznamenaj pomocou príkazu LET do reťazcovej premennej. Slovenské slová načítavaj príkazom INPUT. Napísaním oboch premenných dostaneš uglijské slovo. Využi vlastnosti príkazu PRINT a vytvor čo najjednoduchší program.



2 Výpočet rýchlosti

Vedel by si napísať program na výpočet rýchlosti? Pomocou príkazu INPUT načítaj čas a vzdialenosť. Vypočítaj rýchlosť a napíš ju príkazom PRINT.



3 Program Safaládky

Robot Červenáčik zje 30 safaládok za hodinu, robot Modráčik zje za ten istý čas iba 20 safaládok. Koľko safaládok si musia kúpiť a ako dlho ich budú jesť, ak Modráčik chce zjesť aspoň 35 safaládok a Červenáčik odmieta jesť pomalšie?

RADA: Použi dve premenné, v ktorých si zaznamenáš, koľko safaládok zjedia roboti za hodinu. V ďalšej premennej si ulož, koľko safaládok chce zjesť Modráčik. Použitím ďalších premenných si zaznamenáš, za aký čas zje Modráčik 35 safaládok a koľko ich zje za ten istý čas Červenáčik.

Opakovanie činností

Ak chceme prikázať počítaču, aby opakoval nejakú časť programu, môžeme použiť napríklad cyklus FOR/NEXT. Aby si videl, ako cyklus pracuje, spusti si program zo žltého obdĺžnika. Vyrieš cvičenia a hlavolamy.

```
10 FOR K=1 TO 10
20 PRINT K
30 NEXT K
```

V cykloch sa najčastejšie používajú premenné I, J, K, L.

```
10 FOR I=1 TO 10
20 PRINT I;" x 8 ";
30 PRINT "= ";I*8
40 NEXT I
```

```
10 FOR L=1 TO 15
20 PRINT TAB(L);
"AHOJ"
30 NEXT L
```

O tom, ako sa mení hodnota premennej v cykle, sa presvedčíš tak, že ju zobrazíš príkazom PRINT. Hodnotu premennej v cykle môžeš vnútri cyklu využiť. Vyskúšaj si to.

Hlavolamy s cyklami

1

```
AHOJ AHOJ AHOJ A
HOJ AHOJ AHOJ AH
OJ AHOJ AHOJ AHO
J AHOJ AHOJ AHOJ
```

2

```
AHOJ
AHOJ
AHOJ
AHOJ
AHOJ
AHOJ
```

3

```
10 FOR I=1 TO 20
20 LET I=I-1
30 PRINT I
40 NEXT I
```

Kde je v tomto programe chyba? Odskúšaj si ho.

Vieš napísať jednoduchý program s cyklom, ktorý zaplní obrazovku slovami AHOJ? A ďalší, ktorý napíše slová AHOJ do stĺpca v strede obrazovky?

```
10 FOR I=1 TO 25 STEP 5
20 PRINT I
30 NEXT I
```

```
10 FOR J=20 TO 1 STEP -1
20 PRINT J
30 NEXT J
```

Kroky

Počítanie krokov v cykle môžeš zmeniť, ak použiješ slovo STEP s číslom. Počítač vtedy pripočíta k premennej v cykle zadané číslo. Ak je záporné, kroky odčíta. Spusti programy v modrých obdĺžnikoch a zmeň číslo za STEP.

1

```
AHOJ
 AHOJ
  AHOJ
   AHOJ
    AHOJ
     AHOJ
```

Použi TAB.

2

```
5   25
4   16
3   9
2   4
1   1
0   0
```

3

```
ŠTART   ?12
STOP    ?0
KROK    ?-3
12  9   6   3   0
```

18 Skús napísať programy, ktoré napíšu na obrazovku to, čo je na týchto obrázkoch.

Zdržiavací cyklus

Počítače nepracujú rovnako rýchlo, preto možno budeš musieť cyklus predĺžiť alebo skrátiť zmenením čísla 1000.

```
10 FOR J=10 TO 1 STEP -1
20 PRINT J
30 NEXT J
40 FOR K=1 TO 1000
50 NEXT K
60 PRINT "HOTOVO"
```

Zdržiavací cyklus je prázdny cyklus. V tomto programe riadky 40 – 50 spôsobia, že počítač bude počítať od 1 do 1000. Nám sa zdá, že má prestávku.

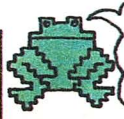
```
TAJNÉ POSOLSTVO
OBJAVÍ SA NA 5 SEKÚND
A POTOM ZMIZNE

ČAKAJ POSLA X 2.00  LETISKO
```

Napíš taký program, aby počítač napísal na obrazovku tajnú správu a po piatich sekundách ju zmazal. Potrebuješ na to prázdny cyklus a za ním príkaz na vymazanie obrazovky. Správne číslo pre prázdny cyklus nájdeš pokusmi.

Obrázkové hlavolamy

1

```
10  LET A$="***"
20  FOR J=1 TO 7
30  PRINT TAB(J);A$
40  NEXT J
50  FOR K=1 TO 3
60  PRINT TAB(J+1);A$
70  NEXT K
80  FOR L=7 TO 1 STEP -1
90  PRINT TAB(L);A$
100 NEXT L
```



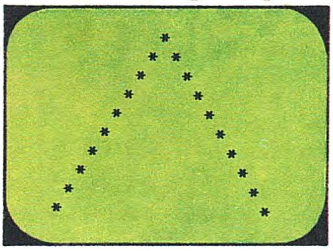
Ak tvoj počítač pozná grafické symboly, môžeš ich použiť namiesto hviezdičiek.

```
80  FOR L=8 TO 18
90  PRINT TAB(L);A$;"**"
100 NEXT L
```

Program vľavo nakreslí na obrazovku čiaru z hviezdičiek. Spusti ho a skús ho zmeniť tak, aby kreslil iné obrázky (pozri príkazy vpravo).

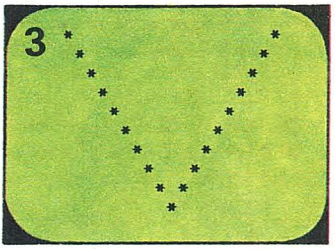
2

```
 10 GCLEAR
 20 LET A=15
 30 PRINT TAB(A);"*"
 49 FOR K=1 TO 9
*50 PRINT TAB( ? );"*";
*60 PRINT TAB( ? );"*"
*70 ?
```



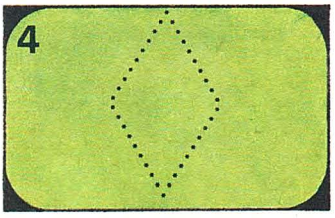
Vieš doplniť riadky 50 až 70 tak, aby počítač nakreslil takýto obrázok?

3



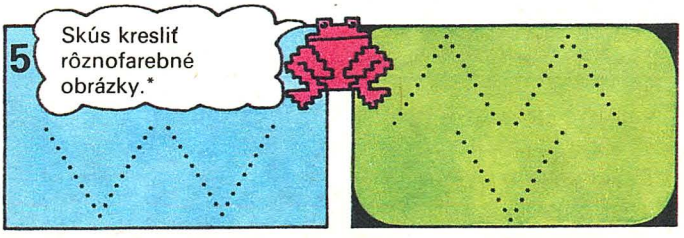
Zmeň program tak, aby bol obrázok naopak!

4



Spoj programy z hlavolamov 2 a 3 — nakreslí sa kosoštvorec.

5



Zmeň programy tak, aby vznikli takéto obrázky.

* Príkazy na kreslenie rôznymi farbami si preštuduj v príručke.

Cykly

Na nasledujúcich dvoch stranách sa v hlavolamoch a programoch využívajú vnorené cykly, teda cykly v cykloch. Ako pracujú, ukážeme si na príkladoch.

```
10 FOR K=1 TO 3
20 FOR L=1 TO 5
30 PRINT K,L
40 NEXT L
50 NEXT K
```

vnorený cyklus

Program píše hodnoty premenných v cykloch, takže uvidíš, ako vnorené cykly pracujú.

```
10 FOR I=1 TO 4
20 FOR J=1 TO 4
30 PRINT
40 NEXT J
50 PRINT "AHOJ"
60 NEXT I
```

Spusti program a skús zmeniť veľkosti cyklov. Vieš spomaliť program tak, že doň vnoríš ďalší cyklus?

Chyby v cykloch

```
10 FOR L=1 TO 15
20 PRINT TAB(5);"*";
   TAB(10);"*"
30 FOR J=1 TO 5
40 PRINT
50 NEXT J
60 NEXT L
```

```
10 FOR I=9 TO 0 STEP -1
20 FOR J=9 TO 0 STEP -1
30 FOR K=9 TO 0 STEP -1
40 PRINT I;J;K
50 NEXT I
60 NEXT J
70 NEXT K
```

Nezabudni napísať obe časti vnoreného cyklu dovnútra vonkajšieho cyklu. Ľavý program je správny, ale v pravom musíš nájsť a opraviť chyby.

Dvojkové počítadlo

```
10 FOR A=0 TO 1
20 FOR B=0 TO 1
30 FOR C=0 TO 1
40 FOR D=0 TO 1
50 PRINT D+C*2+B*4+A*8;" = ";
60 PRINT A;B;C;D
70 NEXT D
80 NEXT C
90 NEXT B
100 NEXT A
```

V riadku 50 sa vypočíta desiatková hodnota dvojkového čísla.

Počítač najskôr vykoná najvnútornejší cyklus.

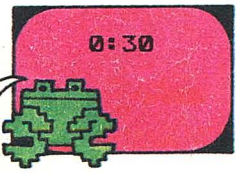
Dvojkové čísla sa v programe vytvárajú pomocou štyroch vnorených cyklov. Program preto vytvára najviac štvorciferné dvojkové čísla. Pridaj ďalšie cykly a zmeň riadky PRINT tak, aby sa vytvárali osemciferné dvojkové čísla.

Blikajúca správa

Správa sa má na obrazovke objaviť a zase zmiznúť. V programe použi cyklus, v ktorom sa vymaže obrazovka a napíše správa, a dva zdržiavacie cykly, aby správa neblikala príliš rýchlo.

Počítačové hodiny

Zvoľ veľkosť zdržiavacieho cyklu tak, aby trval sekundu.



Napíš program, ktorý spraví z počítača hodiny. Použi cykly pre sekundy a pre minúty a zdržiavací cyklus.

Program Štart rakety

Vedel by si napísať program s niekoľkými cyklami, ktorý by na obrazovke simuloval štart rakety tak, že by ju postupne posúval hore?

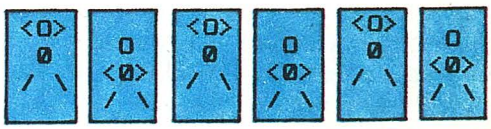
1. Najskôr vymaž obrazovku a urob toľko prázdnych riadkov, koľko treba na to, aby sa raketa zobrazila na spodný okraj obrazovky.

2. Jednoduchú raketu môžeš nakresliť pomocou obyčajných znakov v príkaze PRINT.

3. Vypisuj prázdne riadky a raketa sa bude posúvať hore. Aby sa neposúvala príliš rýchlo, použi zdržiavací cyklus.

Program Skákajúci panáčik

Skús napísať jednoduchý animačný program: panáčik nech skáče hore-dolu na obrazovke, ako to vidíš na obrázkoch vpravo.



Ako nakreslíš panáčika

Panáčika v dvoch rôznych polohách nakreslíš pomocou príkazu PRINT z písmen a symbolov.

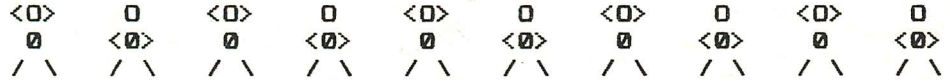
Ako napíšeš program

1. Použi cyklus, ktorý sa zopakuje toľkokrát, koľko dovolí šírka obrazovky. Kroky počítaj po dvoch. Prečo? To uvidíš pri písaní príkazu PRINT TAB.

2. Vnútri cyklu najprv vyčisti obrazovku. Pomocou PRINT nakresli prvého panáčika. Použi TAB, aby sa jeho pozícia menila pri každom kreslení.

3. Panáčik by mal na obrazovke chvíľku stáť, preto použi zdržiavací cyklus.

4. To isté, ale s odlišným TAB, zopakuj pre druhého panáčika.



Príkaz IF/THEN

Príkaz IF/THEN používame vtedy, keď chceme počítaču povedať, aby porovnal dva údaje a podľa výsledku porovnania vykonal rôzne činnosti.

> = znamená väčší alebo rovný
< = znamená menší alebo rovný

```
10 LET A=8
20 INPUT B
30 IF B = A THEN PRINT "SÚ ROVNAKÉ"
40 IF B <> A THEN PRINT "NIE SÚ
   ROVNAKÉ"
50 IF B < A THEN PRINT B;" JE
   MENEJ AKO ";A
60 IF B > A THEN PRINT B;" JE
   VIAC AKO ";A
```

Program spusti niekoľkokrát, aby si sa presvedčil, či správne rozumieš symbolom, ktoré sa používajú na porovnávanie informácií.

```
10 PRINT "KOĽKO PALACINIEK
   CHCEŠ NA OBED?"
20 INPUT C
30 IF C >= 0 AND C <= 5 THEN
   PRINT "NENAJEŠ SA"
40 IF C > 5 AND C <= 10 THEN
   PRINT "AKURÁT"
50 IF C > 10 AND C <= 1000 THEN
   PRINT "NENÁSYTNÍK"
60 IF C < 0 OR C > 1000 THEN PRINT "!!!
```

V príkaze IF/THEN môžeme súčasne porovnávať niekoľko údajov. Používajú sa pri tom slová AND a OR. Ich význam ti odhalí tento program.

Násobilka

```
10 LET A=13
20 FOR J=1 TO 13
30 PRINT "KOĽKO JE ";J;" x ";A;" ";
40 INPUT B
50 IF B = J*A THEN PRINT "SPRÁVNE"
60 NEXT J
```

Keď zadáš počítaču tento program, vyskúša ťa z násobilky 13. Vlož do programu príkaz IF/THEN tak, aby ti počítač odpovedal aj vtedy, ak je tvoja odpoveď nesprávna, a aby napísal, aká je (<,>).

Heslo

```
10 LET S$="SAFALÁDKY"
20 PRINT "NAPÍŠ HESLO"
30 INPUT P$
40 IF P$ = S$ THEN PRINT
   "DOBRE. POKRAČUJ"
```

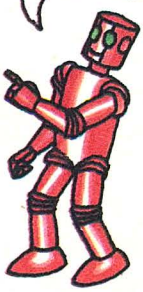
Doplň program tak, aby počítač napísal správu i vtedy, ak zadáš nesprávne heslo. Pridaj aj príkazy, ktoré budú zisťovať tajné číslo.

Rozhodovanie

Za slovom THEN môžeš napísať rôzne príkazy, napr. aby sa počítač zastavil. Pomocou slova GOTO mu môžeš prikázať, aby skočil na určitý riadok.

```
10 LET C=0
20 PRINT "SI UNAVENÝ?"
30 INPUT B$
40 LET C=C+1
50 IF B$ = "ÁNO" THEN STOP
60 IF C > 10 THEN
   PRINT "ASI UŽ SI"
70 IF B$ = "NIE" THEN GOTO 20
80 PRINT "ODPOVEDAJ SLUŠNE!"
90 GOTO 20
```

Čo sa stane, ak odpovieš BANÁNY?



Podľa toho, ako odpovieš na otázku SI UNAVENÝ?, vykoná počítač rôzne príkazy. Program spusti niekoľkokrát a zakaždým odpovedaj inak.

Počítač — kalkulačka

```
ZADAJ ČÍSLO ? 7
ĎALŠIE ? 11
CHCEŠ ICH:
SČÍTAŤ, ODČÍTAŤ,
VYDELIŤ ALEBO VYNÁSOBIŤ?
? VYNÁSOBIŤ
VÝSLEDOK JE 77
```

Tu je výstup programu, ktorý sčituje, odčituje, delí a násobí dve zadané čísla. Skús takýto program napísať. Doplň ho, aby robil aj iné operácie, napríklad umocňoval a odmocňoval.

Hádanie čísel a slov

```
1
10 INPUT X
20 GCLEAR
30 PRINT "HÁDAJ
   ČÍSLO"
40 INPUT Y
50 IF Y = X THEN
   GOTO 70
60 GOTO 30
70 PRINT "SPRÁVNE"
```

1. Hráč uhádne číslo ľahšie, ak mu počítač vždy povie, či bol jeho odhad vysoký alebo nízky.
2. Do programu pridaj premennú, v ktorej sa bude počítať, koľkokrát hráč hádal. Pomocou IF/THEN povoľ iba určitý počet pokusov.

Takto vyzerá program pre jednoduchú hru, v ktorej sa hádajú čísla. Prvý hráč vyberie číslo, druhý ho háda. Zlepši program podľa 1. a 2. bodu.

Program doplň tak, aby sa slovo hádalo ľahšie.

```
2
POMÔCKA
KVÁKAJÚCE ZVIERA
? ROPUCHA
NIE
? ROSNIČKA
ÁNO
```

Hra, v ktorej sa hádajú slová, je podobná hre s hádaním čísel. Rozdiel je iba v tom, že sa v nej používajú reťazce. Podarí sa ti hru naprogramovať?

Dostihy

Pozri si program hry Dostihy — nie je úplný. V riadkoch s hviezdičkou chýba číslo za slovom GOTO. Návod, ako hrať a ako hru zlepšiť, nájdeš úplne dolu.

Na mnohých počítačoch môžeš príkazom INPUT vypísať aj otázku (pozri riadky 20 a 30).

```
 10  LET N=0
 20  INPUT "1. MIESTO :";H1
 30  INPUT "2. MIESTO :";H2
 40  GCLEAR
 50  LET N=N+1

 60  INPUT  "HÁDAJ 1. MIESTO:";G1
 70  INPUT  "HÁDAJ 2. MIESTO:";G2

*80  IF G1 = H1 AND G2 = H2 THEN GOTO ?
*90  IF G1 = H2 OR G2 = H1 THEN GOTO ?
*100 IF (G1 = H1 AND G2 <> H2) OR
     (G2 = H2 AND G1 <>  H1) THEN GOTO ?

 110 PRINT "ZLE"
*120 IF N = 4 THEN GOTO ?
 130 PRINT "MÁŠ ĎALŠÍ POKUS": GOTO 50
 140 PRINT "JEDEN ODHAD SPRÁVNY": GOTO 120
 150 PRINT "KÔŇ SPRÁVNY, ALE ZLE UMIESTNENÝ": GOTO 120
 160 PRINT "SPRÁVNE"
 170 PRINT "1. MIESTO:";H1;"  2. MIESTO:";H2
```

Prvý hráč rozhodne, ktorý kôň dobehne prvý a ktorý druhý.

V premennej N sa počíta, koľkokrát druhý hráč hádal.

Nezabudni doplniť chýbajúce čísla.

Väčšine počítačov neprekáža viac príkazov v jednom riadku. Ak tvoj protestuje, premiestni druhý príkaz na ďalší riadok a zmaž dvojbodku.

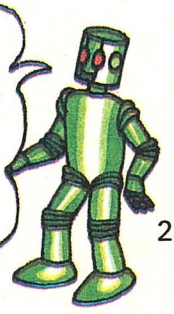
Ako hrať

Dostihov sa zúčastnilo šesť koní očíslovaných od 1 po 6. Prvý hráč určí, ktorý zvíťazil a ktorý dobehol druhý. Na uhádnutie správneho poradia má spoluhráč štyri pokusy.

Ako hru zlepšiť

1. Vymysli bodovací systém a doplň ho do hry.

2. Umožni hráčom, aby sa po skončení hry rozhodli, či chcú pokračovať, alebo nie.



Náhodné čísla

Príkaz RND hovorí počítaču, aby náhodne vybral číslo. Existuje niekoľko variantov tohto príkazu (nižšie nájdeš niektoré z nich). Ktorý je správny pre tvoj počítač, zisti v zozname na konci knihy alebo vo svojej príručke. Pokús sa vyriešiť všetky cvičenia a hlavolamy z tejto dvojstrany.

```
PRINT RND(99)
PRINT RND(10)
```

Na niektorých počítačoch stačí napísať RND a číslo v okrúhlych zátvorkách. Počítač náhodne vyberie číslo väčšie než 1 a menšie alebo rovné číslu v zátvorke.

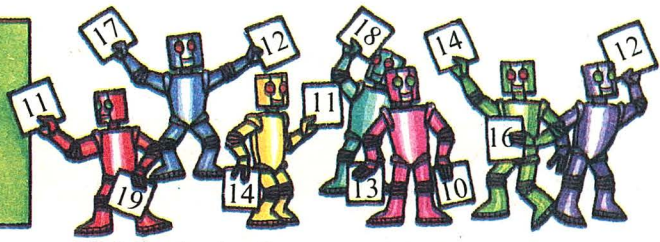
```
PRINT INT(RND(1)*99+1)
PRINT INT(RND(0)*99+1)
PRINT INT(RND*99+1)
```

Na iných počítačoch treba použiť slová INT a RND, prípadne ešte (0) alebo (1). Výsledok sa vynásobí hornou hranicou intervalu, z ktorého má byť náhodné číslo, a pripočíta sa dolná hranica. Všetky príkazy v oranžovom obdĺžniku znamenajú, že počítač má zvoliť číslo od 1 do 99.

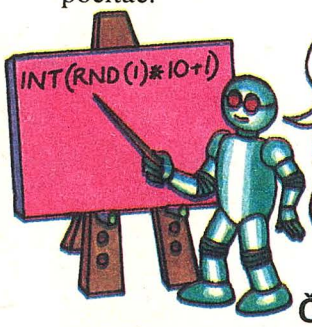
Skontroluj, či na tvojom počítači má za RND nasledovať (1) alebo (0), prípadne nič.

```
PRINT RND(6)+5
PRINT INT(RND(1)*6+5)
PRINT INT(RND(0)*6+5)
PRINT INT(RND*6+5)
```

Tieto príkazy vytvoria náhodné číslo od 5 do 10. Zisti, ktorému rozumie tvoj počítač.



Vieš napísať príkaz, ktorý bude vytvárať náhodné čísla od 10 do 20?



V knihe budeme používať tento tvar príkazu RND. Ak pre tvoj počítač nevyhovuje, nahraď ho iným.

Programátorský hlavolam

Vráť sa na stranu 23 k hre, v ktorej sa hádajú čísla. Zmeň ju tak, aby počítač vybral číslo od 1 do 20, ktoré má hráč uhádnuť.

Číselná postupnosť

```
UHÁDNI NASLEDUJÚCE
ČÍSLO V TEJTO
POSTUPNOSTI
 4  13  22
? 31
SPRÁVNE
```

```
*10 LET X= ?
*20 LET Y= ?
 30 FOR I=1 TO 3
 40 PRINT X+I*Y
 50 NEXT I
```

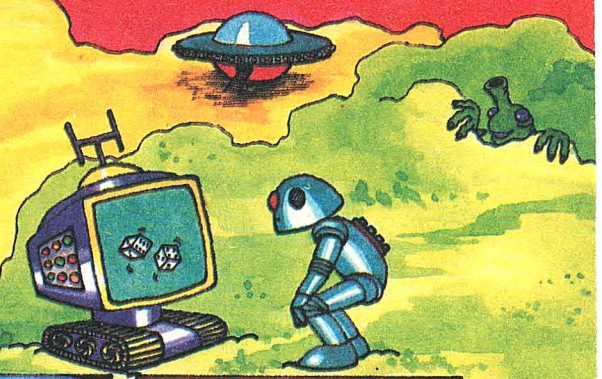
```
PRINT X+I*I
PRINT I*I-Y
PRINT X+Y-I*I
PRINT Y+X^I
```

Vedel by si napísať program pre hru Číselná postupnosť, ktorá je opísaná na ľavom obrázku? Časť tohto programu je na strednom obrázku.

V riadkoch 10 a 20 musíš doplniť príkaz na vytváranie náhodného čísla. Na koniec programu pridaj riadky s príkazmi PRINT, INPUT a IF/THEN. Ak zmeníš riadok 40 (pozri pravý obrázok), program bude vytvárať iné číselné postupnosti. Vymysli spôsob, ako rozšíriť program tak, aby počítač po každom spustení programu náhodne vybral predpis, podľa ktorého vypočíta členy postupnosti.

Odlet z planéty Zorgos

Skús napísať program pre takúto hru: Uviazol si na planéte Zorgos. Aby tvoj palubný počítač našiel cestu späť na Zem, potrebuje dostatočne veľkú pamäť — 50 čipov. Máš ich však iba 10 a jediný spôsob, ako získať ďalšie, je vyhrať ich v riskantnej hre s počítačom na planéte Zorgos. Dole na obrázkoch vidíš, ako hra prebieha.



```
POČET TVOJICH ČIPOV:10
TVOJ VKLAD: 5
AK SI PRIPRAVENÝ, STLAČ P:P
     5     2
VKLAD SA TI VRÁTIL
```

```
POČET TVOJICH ČIPOV:10
TVOJ VKLAD: 9
AK SI PRIPRAVENÝ, STLAČ P:P
     6     4
VYHRAL SI TROJNÁSOBOK VKLADU
POČET TVOJICH ČIPOV:28
```

HRA	VÝHRA
ROVNAKÉ:	DVOJNÁSOBOK
10 ALEBO 11:	TROJNÁSOBOK
6 ALEBO 7:	VKLAD
INÉ:	ŽIADNA

Do hry môžeš vložiť ľubovoľný počet svojich čipov. Podľa toho, aké čísla padnú, vyhrávaš alebo prehrávaš. Ak padnú dve rovnaké čísla, vyhráš dvojnásobok vkladu, ak je súčet čísel 10 alebo 11, trojnásobok vkladu atď. — pozri si tabuľku.

Papier, kameň alebo nožničky: odhaľ chyby

Tento program pre hru Papier, kameň alebo nožničky je plný chýb. Pomocou poznámok na pravej strane, ktoré ti povedia, čo by program mal robiť, chyby nájdi a oprav ich.

```
10  GCLEAR
20  LET C=0
30  LET A=0
40  LET F=0
50  LET R=INT(RND(1)*4+1)
60  IF  R = 1 THEN LET C$="PAPIER"
70  IF  R = 2 THEN LET C$="KAMEŇ"
80  IF  R = 3 C$="NOŽNIČKY"
90  PRINT "SOM PRIPRAVENÝ"
100 PRINT "VYBER SI PAPIER, KAMEŇ ALEBO NOŽNIČKY"
110 INPUT A$
120 PRINT
130 IF C$ = "PAPIER" AND A$ = "NOŽNIČKY" THEN LET F=1
140 IF C$ = "KAMEŇ" AND A$ = "PAPIER" THEN LET F=1
150 IF A$ = "NOŽNIČKY" AND A$ = "KAMEŇ" THEN LET F=1
160 IF C$ = A$ THEN LET F=2
170 PRINT "TY SI MAL ";A$
180 PRINT "A JA ";C$
190 PRINT "TAKŽE
200 IF F = 0 THEN PRINT "SOM VYHRAL"
210 IF F = F THEN PRINT "SI VYHRAL"
220 IF F = 2 THEN PRINT "JE TO REMÍZA"
230 IF F = 0 THEN LET A=A+1
240 IF F = 1 THEN LET C=C+1
250 PRINT "SKÓRE:"
260 PRINT "JA     :";C
270 PRINT "TY     :";A
280 IF C > 10  AND A > 10 THEN GOTO 40
290 PRINT "KONIEC"
```

V C je uložený počet výhier počítača. V A je uložený počet výhier hráča. V riadkoch 130–160 počítač zaznamená do F, kto práve vyhral.

Vyberie (podľa hodnoty v R) papier, kameň alebo nožničky a zaznamená to v C$.

Tu zistí, kto vyhral. Ak vyhral hráč, tak F=1, ak je remíza, F=2. Inák F=0, čo znamená, že vyhral počítač.

Podľa hodnoty premennej F počítač napíše, kto je víťazom, a upraví skóre.

Pamätáš si pravidlá hry? Papier balí kameň, kameň tupí nožničky a nožničky strihajú papier.

Pracujeme so znakmi

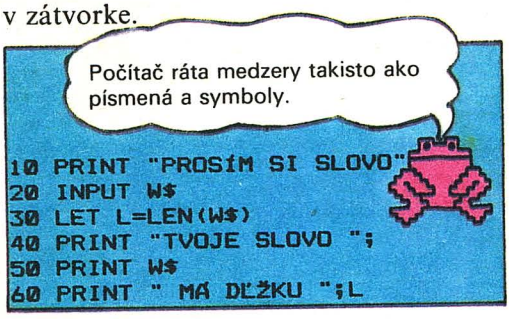
Počítač vie robiť so znakmi v reťazcoch všeličo. Tu nájdeš niekoľko príkazov, ktoré sa používajú v basicu na prácu so znakmi, spolu s vysvetlením, čo znamenajú. Na počítačoch Sinclair sa musia použiť príkazy v zelenom štvorci.

```
LET R$="ROBOT"
PRINT LEFT$(R$,3)
PRINT RIGHT$(R$,3)
LET C$="POČÍTADLO"
PRINT LEFT$(C$,6)
PRINT RIGHT(C$,7)
```

Príkazy LEFT$ a RIGHT$ znamenajú, že počítač má vziať niekoľko znakov z ľavého alebo pravého konca reťazca (LEFT je ľavý, RIGHT je pravý). Koľko znakov má vziať, určuje číslo v zátvorke.

```
10 LET K$="KENGURA"
20 PRINT "POZÍCIA"
30 PRINT "PRVÉHO PÍSMENA";
40 INPUT S
50 PRINT "KOĽKO PÍSMEN";
60 INPUT N
70 PRINT MID$(K$,S,N)
```

Príkaz MID$ znamená, že počítač má vziať znaky zo stredu reťazca (MID je uprostred). Prvé číslo v zátvorke hovorí, odkiaľ má začať počítať, a druhé, koľko znakov má vziať.



Príkazom LEN sa spočíta počet znakov v reťazci (LENGTH je dĺžka). Vyskúšaj si tento program.

Počítače Sinclair

```
LET A$="ABCDEFGHIJKLMNOP"
PRINT A$(4 TO 6)
DEF
PRINT A$(14 TO 16)
NOP
```

Keďže počítače Sinclair nepoznajú LEFT$, RIGHT$ a MID$, takto im oznámiš, s ktorými znakmi chceš pracovať.

Hlavolamy s reťazcami

1

```
10 INPUT "SLOVO";N$
20 LET L=LEN(N$)
30 FOR K=1 TO L
*40 PRINT TAB(K);
    MID$(?,?,?)
50 NEXT K
```

```
Z O S O R O N E C
```

Nahraď otázniky v riadku 40 tak, aby počítač na obrazovku napísal slovo šikmo. **Rada:** pomocou príkazu MID$ napíš vždy jeden znak slova.

2

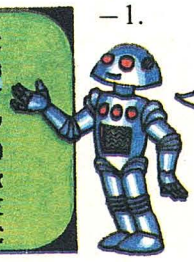
```
RUN
PROSÍM SI SLOVO
? KROKODÍL
LÍDOKORK
```

Napíš program, ktorý napíše slovo odzadu. Použi v ňom príkaz MID$ a cyklus s krokom −1.

3

```
10 LET S$=
   "ŠTVOREC"
20 LET L=LEN(S$)
30 PRINT S$
40 FOR J=1 TO L
*50 ?
60 NEXT J
```

```
Š T V O R E C
T V O R E C Š
V O R E C Š T
O R E C Š T V
R E C Š T V O
E C Š T V O R
C Š T V O R E
Š T V O R E C
```



Využi príkazy LEFT$ a RIGHT$ a doplň riadok 50 tak, aby sa na obrazovke objavilo toto!

Najdlhšie slovo

V riadku 50 si má počítač v premennej A$ zaznamenať doteraz najdlhšie načítané slovo. Využi pritom IF/THEN a LEN.

```
10 LET A$=""
20 PRINT "PROSÍM SI SLOVO"
30 FOR J=1 TO 5
40 INPUT S$
*50 ?
60 NEXT J
70 PRINT "NAJDLHŠIE SLOVO :"
80 PRINT A$
```

```
PROSÍM SI SLOVO
? MAČKA
? KOBRA
? NOSOROŽEC
? LEV
? MRAVEC
NAJDLHŠIE SLOVO :
NOSOROŽEC
```

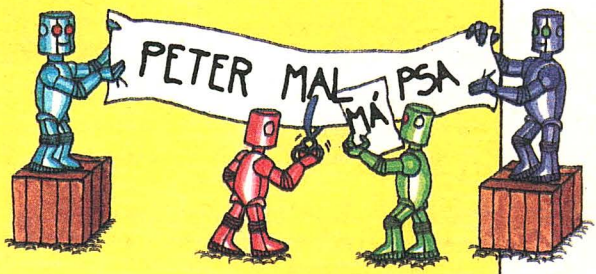
Program nájde najdlhšie slovo spomedzi piatich slov. Doplň chýbajúci riadok a program spusti.

Najkratšie slovo

Vieš napísať program, v ktorom by sa hľadalo najkratšie slovo? Podobá sa predchádzajúcemu. Budeš však potrebovať počiatočnú hodnotu premennej A$ dlhšiu než ktorékoľvek vstupné slovo, aby počítač mohol slová navzájom porovnávať. Zmeniť sa musí aj príkaz IF/THEN.

Zámena slov vo vete

Nasledujúci program ti umožní napísať vetu a potom v nej meniť slová. Skôr než ho spustíš, nahraď otázniky v riadkoch s hviezdičkou. Pomôžu ti poznámky na pravej strane.




```
  10  GCLEAR
  20  PRINT "NAPÍŠ VETU"
  30  INPUT V$
* 40  LET V$= ?
  50  PRINT "ČO TREBA ZAMENIŤ";
  60  INPUT S$
* 70  LET S$= ?
  80  PRINT "A ČÍM";
  90  INPUT N$
* 100 LET LV= ?
* 110 LET LS= ?
  120 LET A$=""
  130 LET K=1
* 140 IF MID$(V$, ? , ? ) = S$ THEN
      LET A$=V$
* 150 IF A$ = V$ THEN LET V$=LEFT$
      ( ? , ? )+ ? +RIGHT$(A$,LV-(K+LS-2))
  160 LET LV=LEN(V$)
  170 LET K=K+1
* 180 IF K <= LV-LS+1 THEN GOTO ?
  190 PRINT V$
  200 GOTO 50
```

Ako program funguje, zistíš najjednoduchšie tak, že si na papier napíšeš vetu a postupne vykonávaš príkazy programu.

Počítač by mal na začiatok a koniec reťazcov V$ a S$ pridať medzeru. Prikáž mu to.

LV sa má rovnať dĺžke vety (V$) a LS dĺžke slova (S$).

Počítač má vo vete vyhľadať slovo, ktoré treba nahradiť. Rada: na počítanie znakov použi premennú K.

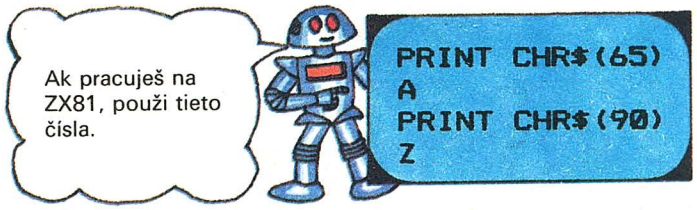
Počítač vytvorí novú vetu: vezme znaky vľavo od slova, ktoré nahrádza, vezme nové slovo a nakoniec pridá zvyšok vety. Doplň chýbajúce premenné a čísla.

Ešte raz o znakoch

Vnútri počítača je namiesto každého znaku uložené číslo — jeho kód. Pri narábaní so znakmi môžeš kódy využívať. Príkaz CHR$ prevedie kód na znak, príkaz ASC (alebo CODE) robí opačnú činnosť — prevedie znak na jeho číselný kód. Väčšina mikropočítačov (okrem ZX81) pracuje s kódom ASCII. Zoznam číselných kódov jednotlivých znakov nájdeš vo svojej príručke alebo v knihe Skúsiš to s mikropočítačom?.

Používame CHR$

```
PRINT CHR$(38)
A
PRINT CHR$(63)
Z
```



Vyskúšaj si príkaz PRINT CHR$ s rôznymi číslami. Niektoré čísla sú priradené napríklad medzerníku alebo riadkovaču, a preto sa na obrazovke neobjaví nič. Ktoré sú to čísla, zisti vo svojej príručke.

Písmenkové hlavolamy

1

```
* 10 FOR K= ? TO ?
  20 PRINT CHR$(K);
  30 NEXT K
```

Skús do cyklu doplniť čísla tak, aby program napísal abecedu.

2

```
abcdefghijklm
nopqrstuvwxyz
```

Zostav program, ktorý napíše malú abecedu (pravdaže, ak ju tvoj počítač pozná).

3

```
I D I P Q C H K A
S K Q C H U A M S P
S N M Q T O P X A
```

Zostav krátky program, ktorý napíše postupnosť náhodne vybraných písmen.

Používame ASC alebo CODE

```
PRINT ASC("P")
PRINT ASC("+")
PRINT ASC(" ")
        medzera
```

```
PRINT CODE("4")
PRINT CODE("U")
PRINT CODE("C")
```

Vyskúšaj príkaz ASC (na počítačoch Sinclair CODE) s rôznymi znakmi a uvidíš, aké čísla počítač napíše.

Porovnávanie písmen

```
?P,O
O JE PRED P
?L,B
B JE PRED L
?S,C
C JE PRED S
```

Napíš program, ktorý porovná dve ľubovoľné písmená a napíše ich v abecednom poradí. Využi symboly <,>.

Malé — veľké

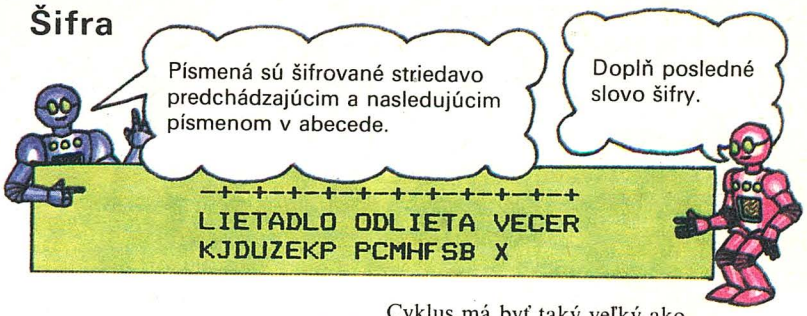
```
  10 PRINT "NAPÍŠ SPRÁVU "
  20 INPUT M$
  30 FOR J=1 TO LEN(M$)
  40 LET X$=MID$(M$,J,1)
* 50 IF X$ >= "a" AND X$ <= "z" THEN
     PRINT CHR$(ASC(X$)- ? );
* 60 IF X$ >= "A" AND X$ <= "Z" THEN
     PRINT CHR$(ASC(X$)+ ? );
  70 PRINT IF X$ < "A" OR X$ > "z" THEN PRINT X$;
  80 NEXT J
```

V riadkoch 50 a 60 nahraď otázniky číslami tak, aby program v tvojej správe zamenil veľké písmená za malé a naopak. Program sa dá použiť len na počítačoch, ktoré rozlišujú veľké a malé písmená.

Šifrujeme správu

Na tejto strane nájdeš niekoľko šifier. Tabuľka vpravo vysvetľuje princíp prvého šifrovacieho programu. Do programu doplň chýbajúce čísla a symboly.

Šifra



Program Šifra

```
  10  PRINT "NAPÍŠ TAJNÚ SPRÁVU"
  20  INPUT S$
* 30  FOR J=1 TO ?
* 40  LET X= ?
* 50  IF X < ? OR X > ? THEN LET N=X:
      GOTO 100
* 60  IF INT(J/2) = J/2 THEN LET N=X ? 1
* 70  IF INT(J/2) <> J/2 THEN LET N=X ? 1
* 80  IF N < ? THEN LET N=N+26
* 90  IF N > ? THEN LET N=N-26
* 100 PRINT ?
  110 NEXT J
```

Cyklus má byť taký veľký ako dĺžka správy S$.

Počítač vezme písmeno a do X uloží jeho kód ASCII.*

Doplň chýbajúce čísla tak, aby počítač o každom znaku zisťoval, či je to písmeno (čísla a medzery zostávajú v šifre nezmenené).

Počítač písmená šifruje tak, že k X buď pripočíta 1 (ak je premenná cyklu J párna), alebo odčíta 1 (ak je J nepárna).

Ak posunuté písmeno (teda jeho kód N) vypadne na niektorom konci z abecedy, vrátiš ho na druhý koniec pripočítaním alebo odpočítaním čísla 26.

Použi CHR$ a napíš zašifrované písmeno.

Číselná šifra

$N = X + kľúč$ ⟵ ⟶ $N = N - 26$

abeceda	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
kódy	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F

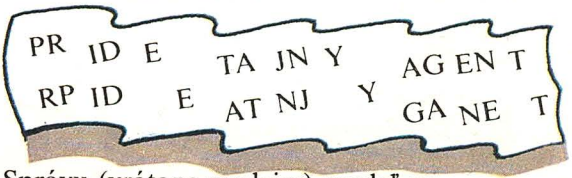
V tejto šifre sa abeceda posunie o istý počet (N) písmen. O koľko sa má posunúť, určuje číselný kľúč. V našom prípade je posunutá o 6 písmen. Napíš program, ktorý bude šifrovať správu týmto spôsobom. Môžeš používať ľubovoľné číselné kľúče.

Šifrovanie v cykle

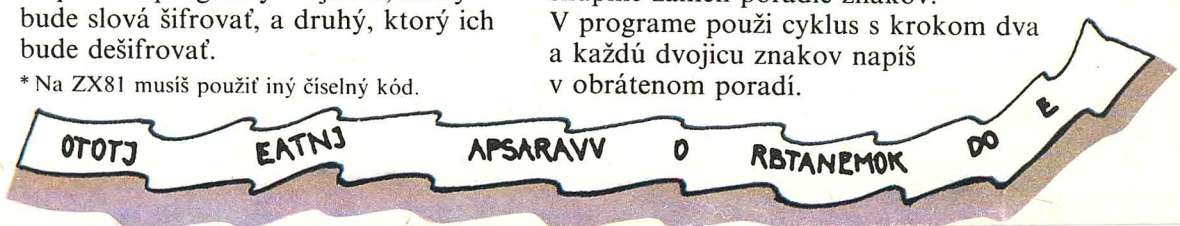
Tu sa ku kódu ASCII každého písmena pripočíta hodnota premennej cyklu (J). Napíš dva programy — jeden, ktorý bude slová šifrovať, a druhý, ktorý ich bude dešifrovať.

* Na ZX81 musíš použiť iný číselný kód.

Obrátená šifra



Správu (vrátane medzier) rozdeľ na dvojznakové skupinky. V každej skupine zameň poradie znakov. V programe použi cyklus s krokom dva a každú dvojicu znakov napíš v obrátenom poradí.



Príkaz INKEY$

Príkaz INKEY$ spôsobí, že počítač zisťuje, či bol stlačený kláves, a ak áno, ktorý. Na rozdiel od príkazu INPUT však počítač nečaká, kým kláves stlačíš, ale hneď pokračuje vo vykonávaní programu.
U niektorých počítačov sa namiesto INKEY$ používa iné slovo.

Skontroluj si v príručke alebo v tabuľke na strane 147, ktoré slovo pozná tvoj počítač.

```
10 LET A$=INKEY$
20 IF A$ = "" THEN PRINT "!";
30 IF A$ <> "" THEN PRINT A$;
40 GOTO 10
```

Nahraď INKEY$ správnym príkazom a program spusti. Ak stlačíš niektorý kláves, počítač napíše zodpovedajúci znak, inak napíše výkričník.

```
AHOJ        AHOJ     A
HOJ      AHOJ        AHO
J           AHOJ     AHOJ
AHOJ        AHOJ     AH
OJ       AHOJ        AHOJ
```

Zostav takýto program: ak stlačíš niektorý kláves, počítač vypíše na obrazovku AHOJ, inak napíše medzeru (pozri obrázok). Program je veľmi podobný programu vľavo.

Ako prikázať počítaču, aby čakal

```
10 LET A$=INKEY$(50)
20 IF A$ = "" THEN
   PRINT " ";
30 IF A$ <> "" THEN
   PRINT A$;
40 GOTO 10
```

```
10 LET N=0
20 LET A$=INKEY$
30 IF A$ <> "" THEN GOTO 70
40 LET N=N+1
50 IF N < 50 THEN GOTO 20
60 PRINT " ";:GOTO 10
70 PRINT A$;:GOTO 10
```

Niekedy je užitočné, aby počítač chvíľku počkal a až potom ďalej pracoval. V niektorých verziách basicu stačí za INKEY$ napísať číslo v zátvorke (pozri program vľavo). Číslo udáva v zlomkoch sekundy, ako dlho má počítač stáť. Ak nemáš túto možnosť, môžeš počítaču prikázať, aby stál tak, že príkaz INKEY$ uložíš do cyklu a použiješ GOTO (pozri program vpravo).

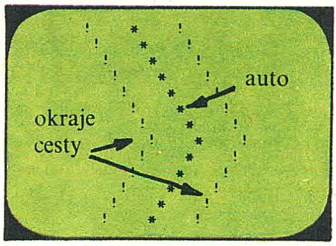
Rýchla matematika

Nájdeš v tomto programe chyby a opravíš ich? Počítač má náhodne vybrať dve čísla medzi 1 a 25. Hráč ich má sčítať a stlačiť ľubovoľný kláves hneď, ako sa na obrazovke objaví správny súčet.

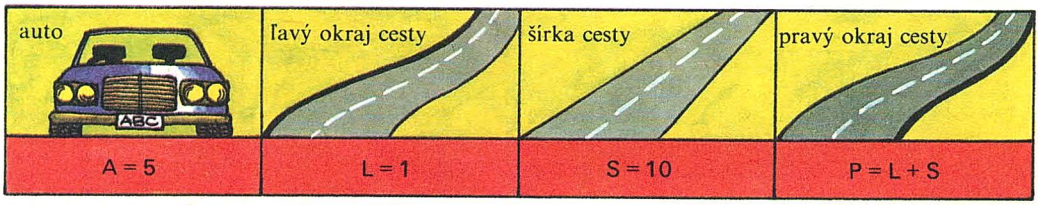
```
10  GCLEAR
20  PRINT "HNEĎ, AKO UVIDÍŠ SPRÁVNY SÚČET,
30  PRINT "STLAČ ĽUBOVOĽNÝ KLÁVES"
40  LET N=0
50  LET X=INT(RND(1)+25+1)
60  LET Y=INT(RND(1)*25+1)
70  PRINT
80  PRINT "X;" + ";Y;" = "
90  LET N=N+1
100 PRINT N
110 LET INKEY$=A$
120 IF A$ <> "" THEN GOTO 180
130 FOR K=1 TO 100:NEXT K
140 IF N < 30 THEN GOTO 90
150 PRINT "SMOLA. ODPOVEĎ JE ";X+Y
160 FOR K=1 TO 1000:NEXT T
170 GOTO 30
180 IF N <> X+Y THEN GOTO 150
190 PRINT "SPRÁVNE: ODPOVEĎ JE X+Y"
```

Ako napísať program Autoškola

Podľa nasledujúceho návodu skús napísať program pre hru Autoškola. Na obrazovke vpravo vidíš, ako hra prebieha. Auto a okraje cesty nakresli na obrazovku príkazom PRINT TAB s * znázorňujúcou auto a ! znázorňujúcim okraj cesty. Cesta sa kľukatí naprieč obrazovkou a hráč pomocou dvoch klávesov ovláda auto, aby nezišlo z cesty.

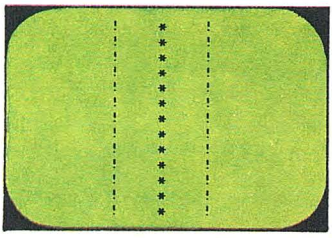


1. Definícia premenných

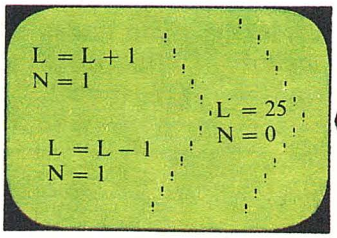


Na nastavenie pozície auta a okrajov cesty budeš potrebovať štyri premenné: A,L,S,P. Možno budeš musieť zmeniť čísla, aby sa cesta zmestila na obrazovku. V prvých piatich riadkoch programu vymaž obrazovku a priraď premenným počiatočné hodnoty.

2. Kreslenie cesty

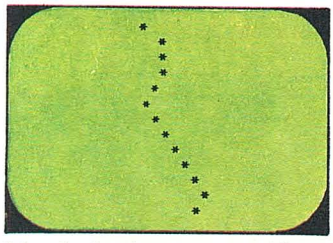


Príkazom PRINT TAB s premennou a symbolom nakresli okraje cesty a auto. Pomocou GOTO príkaz opakuj a nakresli priamu cestu s autom.

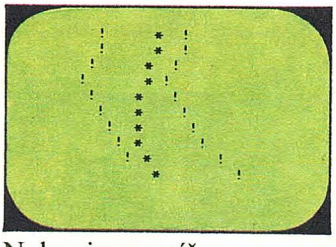


Aby sa cesta kľukatila, vlož do programu riadky, ktoré pri každom opakovaní príkazu PRINT TAB zmenia hodnotu premennej L. Musíš sa však vždy presvedčiť, či cesta nevybehla z obrazovky. Použi na to ďalšiu premennú (N). Priraď jej hodnotu 1, ak L< =1, a 0, ak L> =šírke obrazovky. Potom pripočítaj alebo odpočítaj od L 1 podľa toho, aká je hodnota N.

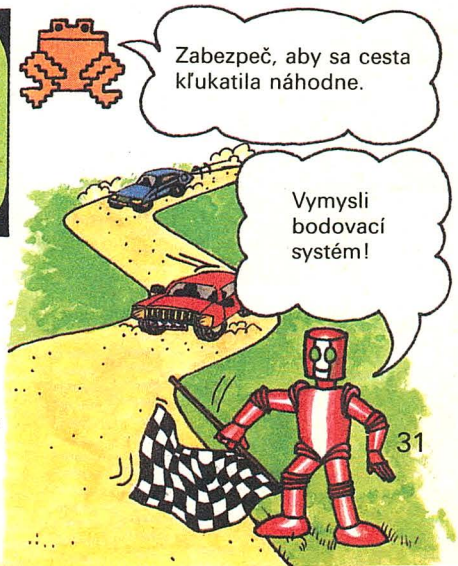
3. Šoférovanie



Na riadenie auta použi príkaz INKEY$. Vyber 2 klávesy (napr. < a >) a podľa toho, ktorý bol stlačený, pripočítaj alebo odpočítaj od C 1. Ak program beží prirýchlo a hráč nestíha riadiť auto, vlož doň zdržiavací cyklus.



Nakoniec musíš skontrolovať, či auto nenarazilo na okraje vozovky — porovnaj A s L a P. Ak došlo k havárii, povedz hráčovi, že má začať odznova.



Používame DATA

Jeden z najjednoduchších spôsobov, akým možno počítaču zadať veľké množstvo údajov, je použiť príkaz READ/DATA. Riadok DATA obsahuje zoznam slov a čísel a READ znamená, že údaje sa majú uložiť do premenných. Programy z tejto strany nefungujú na ZX81, pretože tento počítač uvedené príkazy nepozná. Na strane 34 nájdeš inú metódu, ako zaznamenať údaje.

Načítanie údajov

```
10 FOR I=1 TO 6
20 READ X,X$
30 PRINT X;" ";X$
40 NEXT I
50 DATA 13,RÝB,77,ŽIAB,91,SLIMÁKOV
60 DATA 23,MAČIEK,62,PSOV,2,MYŠI
```

Údaje navzájom oddeľuj čiarkou.

Na niektorých počítačoch reťazce treba dať do úvodzoviek.

Aby si sa zoznámil s príkazom READ/DATA, spusti tento program. Za READ nasledujú dve premenné. Pri každom vykonaní cyklu počítač uloží ďalšiu dvojicu údajov do premenných X a X$.

Kontrola mena

```
 10 PRINT "PROSÍM SI MENO ";
 20 INPUT N$
 30 READ X$
 40 IF X$ = N$ THEN GOTO 70
*50 IF X$ = " ? " THEN PRINT "MENO
    NIE JE V ZOZNAME":STOP
 60 GOTO 30
 70 PRINT "MENO JE
    V ZOZNAME"
*80 DATA ?
```

Otáznik v riadku 50 nahraď posledným údajom z riadka DATA.

Po spustení tohto programu sa ťa počítač spýta na meno a postupne ho porovnáva s menami za DATA. Do riadka 80 napíš za DATA mená (ľubovoľne veľa). Aby sa po porovnaní posledného mena počítač zastavil, napíš toto meno do riadka 50.

Používame RESTORE

```
*10  FOR J= ?  TO ?
*20  FOR I= ?  TO ?
 30  READ N$
*40  IF LEFT$(N$,1) = CHR$( ? ) THEN PRINT N$
 50  NEXT I
 60  RESTORE
 70  NEXT J
 80  DATA VIERA,XAVER,ĽUBOŠ,JURAJ,BEATA,JANO,LACO
 90  DATA DANA,TOMÁŠ,ANKA,PETER,BORKA,IGOR,MIRO
 100 DATA IVAN,GABA,ALENA,MILOŠ,MAŤO
```

Cyklus J sa má zopakovať toľkokrát, koľko je písmen v abecede. Doplň číselné kódy znakov A a Z.

Cyklus I sa má zopakovať toľkokrát, koľko je mien v zozname.

Úlohou programu je napísať mená v abecednom poradí. Vieš ho dokončiť? Príkaz RESTORE (čítaj ristor, znovu ulož) hovorí počítaču, že pri opakovaní cyklu J má čítať údaje z riadkov DATA od začiatku. Doplň do programu chýbajúce čísla a premenné a spusti ho.

1 Oprav chyby

```
10 FOR K=1 TO 6
20 READ N : PRINT N
30 NEXT K
40 DATA 112-32,222-36,341-22/2
50 DATA 972-411,47-29-01,23-640
```

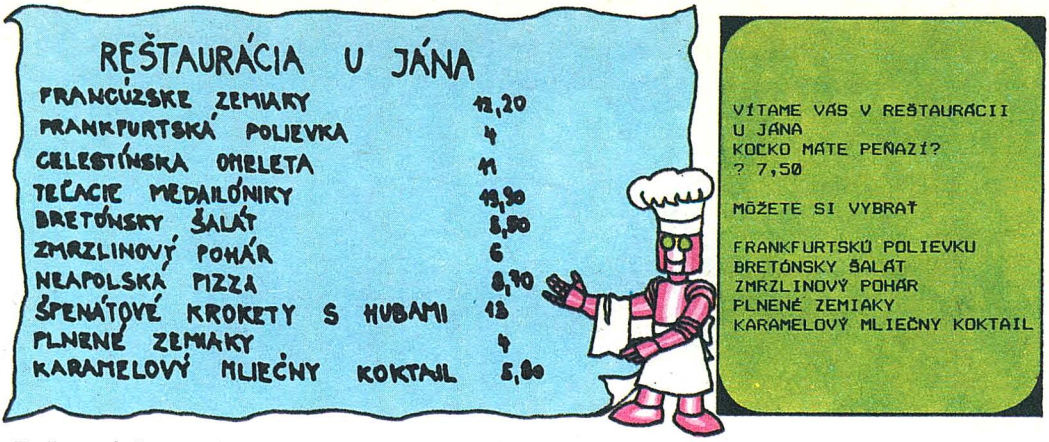
V riadkoch DATA sú telefónne čísla. Nájdeš v programe chybu a opravíš ju?

2

```
10 READ X
20 PRINT X
30 GOTO 10
40 DATA 1,461,892,66,1471,4462,1,3
50 DATA 53,80,241,90,371,825,33,13
```

Aj v tomto programe je chyba. Ak ju nevieš nájsť, program si spusti na počítači. Mal by ti napísať správu o chybe. Vieš vymyslieť jednoduchý spôsob, ako chybu odstrániť?

Reštaurácia U Jána



Reštaurácia U Jána má takýto jedálny lístok. Napíš program, v ktorom budú jedlá a ich ceny v riadkoch DATA. Počítač ti napíše, čo si môžeš vybrať, ak máš len istú sumu peňazí, tak ako vidíš na obrazovke vpravo. Ak chceš, pridaj do jedálneho lístka ďalšie pochúťky.

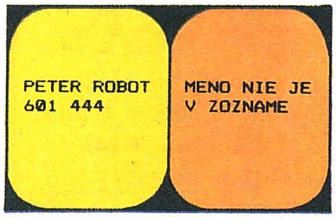
Telefónny zoznam

Dolu je napísaný postup, ako vytvoriť program Telefónny zoznam. Vpravo vidíš, ako vyzerá obrazovka po spustení programu. Skús program napísať.

1. Do DATA napíš mená priateľov a ich telefónne čísla (pozri obrázok).

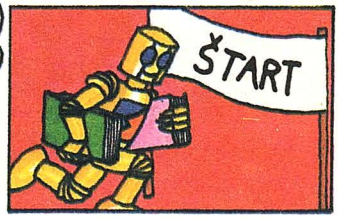
2. Pomocou PRINT a INPUT prikáž počítaču, aby sa spýtal, koho číslo hľadáš, a zapamätal si jeho meno.

3. V cykle prehľadávaj zoznam príkazom READ. Pre mená a čísla používaj rôzne premenné.



4. Napíš meno a číslo (malo by byť nasledujúcou položkou v zozname) alebo správu, že meno nie je v zozname.

5. Prikáž počítaču, aby sa ťa spýtal, či hľadáš ďalšie číslo. Odpoveď načítaj pomocou INPUT.



6. Ak je odpoveď kladná, pošli počítač na začiatok programu. Nezabudni na príkaz RESTORE.

Používame polia

Veľmi výhodné je ukladať údaje do poľa. Pole je množina údajov zaznamenaných spolu pod jedným menom. Každý údaj poľa sa nazýva prvok a číslo jeho pozície sa nazýva index prvku.

Číselné polia

Toto číselné pole sa nazýva N. Obsahuje šesť údajov. Počítaču musíš povedať, aké bude pole veľké, aby si v pamäti vyhradil dosť miesta. Urobíš to tak, že napíšeš slovo DIM a zaň názov poľa a počet údajov. Hovoríme tomu deklarácia poľa.

```
*10 DIM ?
 20 FOR K=1 TO
 30 READ N(K)
 40 NEXT K
*50 DATA ?
```

Pre ZX81
(Timex 1000)

```
*10 DIM ?
 20 FOR K=1 TO
 30 INPUT N(K)
 40 NEXT K
```

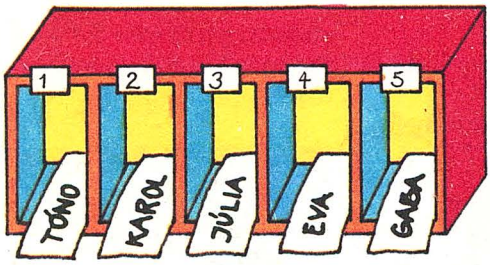
```
N(1) JE 1066
N(2) JE 1216
N(3) JE 1485
N(4) JE 1603
N(5) JE 1665
N(6) JE 1959
```

```
1486   1216
1959   1485
1959   1665
1603   1603
1485   1216
```

Údaje vložíš do poľa príkazom READ/DATA v cykle. Dokonči tento program tak, aby sa do poľa uložili údaje z horného obrázka. Ak máš ZX81, pracuj s programom vpravo a údaje postupne načítavaj príkazom INPUT.

Teraz zostav program, ktorý údaje zaznamenané v poli napíše na obrazovku. Použi príkaz PRINT a počítadlo cyklu ako index poľa. Na obrazovke vpravo sú prvky poľa v náhodnom poradí, ako index poľa je použité náhodné číslo.

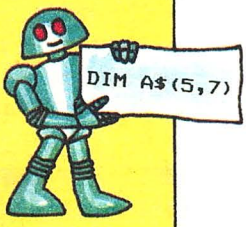
Reťazcové polia



Reťazcové pole M$ obsahuje päť mien, má teda päť prvkov. Ak nemáš Sinclair, môžeš pracovať s reťazcovými poľami presne tak ako s číselnými. Napíš jednoduchý program, ktorý uloží do poľa mená z obrázka a napíše ich na obrazovku.

Počítače Sinclair

ARRAY A$
1 LEOPARD
2 LEV
3 SLON
4 TIGER
5 SOMÁR



Počítače Sinclair si v jednom riadku poľa pamätajú jeden reťazec a každý znak reťazca je uložený v jednom prvku. Pri deklarácii poľa musíš počítaču oznámiť, koľko máš reťazcov a aké sú dlhé. Urobíš to tak, že do zátvorky napíšeš počet reťazcov a dĺžku najdlhšieho — pozri tabuľku v rukách robota.

Počítač — kalendár

```
*10 ?
 20  FOR K=1 TO 12
*30  ?
 40  NEXT K
 50  PRINT "KTORÝ MESIAC ";
 60  INPUT N
*70  PRINT M$( ? );" MÁ ";
*80  PRINT D( ? );" DNÍ"
*90  DATA ?
*100 DATA ?
*110 DATA ?
```

Deklarácia poľa.

Príkazom READ načítaj údaje do polí M$ a D.

Doplň chýbajúce indexy, aby počítač z polí vypísal potrebné údaje.

V týchto riadkoch uveď v príkazoch DATA postupne všetky mesiace a počet ich dní.

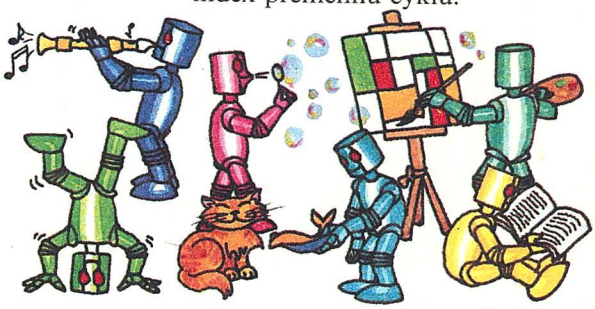
Keď program dokončíš, počítač by mal po načítaní čísla mesiaca napísať jeho meno a počet dní.

```
KTORÝ MESIAC?
JÚL

JÚL MÁ 31 DNÍ
```

Program zmeň tak, aby sa ťa počítač spýtal na meno ľubovoľného mesiaca a napísal, koľko má dní. Meno mesiaca hľadaj v poli M$ v cykle príkazom IF/THEN. Na vyberanie údajov z poľa D použi ako index premennú cyklu.

Čo robiť?

```
VYBER SI ČÍSLO OD 1 DO 10
? 9
A ČO TAKTO POSTAVIŤ SA NA HLAVU?
SÚHLASÍŠ  ? NIE

VYBER SI ČÍSLO OD 1 DO 10
? 5
A ČO TAKTO NAKÁMIŤ MAČKU?
SÚHLASÍŠ  ? ÁNO
```



Nebolo by zlé mať program, ktorý by ti poradil, čo máš robiť, keď sa nevieš rozhodnúť. Vybral by si si číslo a počítač by ti niečo navrhol.

Pri písaní programu použi reťazcové pole (I$), v ktorom budú uložené návrhy. Vybrané číslo ulož príkazom INPUT do premennej N a pri výbere údaja z poľa I$ použi N ako index.

Dvadsať hádaniek

```
ČO RASTIE KOREŇOM
HORE?
? ZUB
SPRÁVNE

DO AKÝCH SUDOV
NEMOŽNO
NALIAŤ VÍNO?
? DO DERAVÝCH
HLÚPOSŤ. DO PLNÝCH.
```

Takto by mohol vyzerať počítačový hádankový kvíz. Pospomínaj si na 20 hádaniek a ulož ich v programe do reťazcového poľa. Odpovede ulož do druhého poľa. Otázka a jej odpoveď budú mať rovnaký index.

Graf náhodných čísel

```
 1 *********
 2 ***************
 3 *********
 4 ***************
 5 *********
 6 *******
 7 **********
 8 *********
 9 ***********
10 **********
```

Na začiatku priradí počítač každému prvku poľa hodnotu 0.

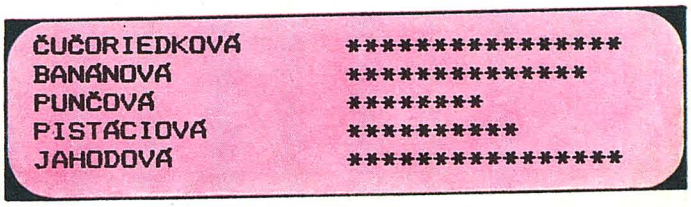
```
10 LET N=0
20 DIM A(10)
30 FOR K=1 TO 10
40 LET A(K)=0
50 NEXT K
60 LET R=INT(RND(1)*10+1)
70 LET A(R)=A(R)+1
80 LET N=N+1
90 IF N < 100 THEN GOTO 60
```

V tomto programe sa používa pole na zaznamenanie údajov pre graf. Počítač náhodne vyberie 100 čísel od 1 do 10. V prvkoch poľa zaznamená, koľkokrát bolo každé vybrané. Skús program dokončiť tak, aby počítač na obrazovku nakreslil graf, v ktorom každé vybrané číslo predstavuje jedna hviezdička. Stačí napísať cyklus, ktorý sa zopakuje 10-krát. Vnútri cyklu bude ďalší cyklus, ktorý nakreslí jeden riadok grafu.

Podprogramy

Podprogram je časť programu. Vykonáva činnosť, ktorú treba pri výpočte viac ráz na rozličných miestach programu zopakovať. Ak počítač narazí na príkaz GOSUB, za ktorým nasleduje číslo prvého riadka podprogramu, začne podprogram vykonávať. Ak narazí na príkaz RETURN, vráti sa späť na príkaz nasledujúci za GOSUB.

Zmrzlinová anketa	
ČUČORIEDKOVÁ	16
BANÁNOVÁ	14
PUNČOVÁ	8
PISTÁCIOVÁ	10
JAHODOVÁ	18

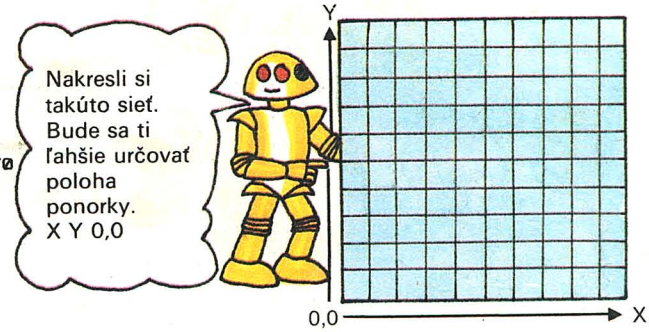


V tabuľke vľavo sú výsledky ankety o najobľúbenejšej zmrzline. Pokús sa vytvoriť program, ktorý napíše výsledky ankety na obrazovku podľa vzoru na pravom obrázku. Použi cyklus, v ktorom sa údaje načítavajú do dvoch polí. Okrem toho pošli počítač vnútri cyklu do podprogramu, ktorý napíše jeden riadok tabuľky.

Ponorka

Zahraj sa s počítačom hru Ponorka. Počítač umiestni do siete 10×10 štvorčekov ponorku. Jej polohu určí tak, že náhodne vyberie dve čísla (súradnice x, y v sieti). Na uhádnutie súradníc ponorky máš štyri pokusy. Ak hádaš zle, počítač skočí do podprogramu, kde napíše, ktorým smerom od tvojho odhadu sa ponorka nachádza. Skôr ako program spustíš, doplň doň podprogramy.

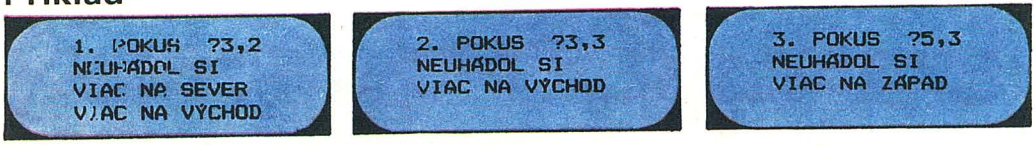
```
10  GCLEAR
20  LET N=0
30  LET X=INT(RND(1)*10+1)
40  LET Y=INT(RND(1)*10+1)
50  LET N=N+1

60  PRINT N;". POKUS  ";
70  INPUT A,B
80  IF A = X AND B = Y THEN GOTO 170

90  GOSUB 200
100 PRINT

110 IF N <= 4 THEN GOTO 50
120 PRINT "UŽ NEMÁŠ VIAC POKUSOV"

130 PRINT "CHCEŠ HRAŤ ODZNOVU?"

140 INPUT R$
150 IF LEFT$(R$,1) = "A" THEN GOTO 10

160 STOP

170 PRINT "PONORKU SI NAŠIEL NA ";N;". POKUS"

180 GOTO 130
```



Ako napísať podprogram

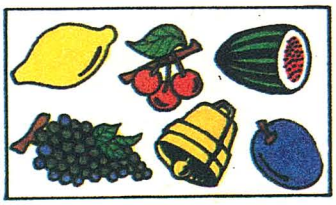
Budeš potrebovať niekoľko riadkov s príkazom IF/THEN na porovnanie svojho odhadu (A,B) so súradnicami ponorky (X,Y) a na napísanie správy na obrazovku. Napríklad ak B je menšie než Y, ponorka je severne od miesta, ktoré si určil, ak A je menšie než X, ponorka je východne.

Príklad

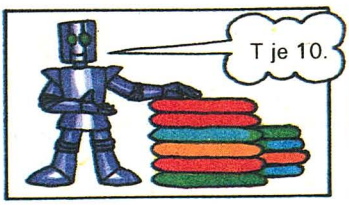


Ovocná lotéria

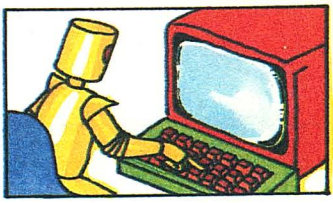
Vedel by si naprogramovať hru Ovocná lotéria? Na obrázku vpravo vidíš, ako by mala vyzerať. Na začiatku má hráč desať žetónov. Za každú hru musí zaplatiť jeden žetón. Hru odštartuje stlačením ľubovoľného klávesu a počítač mu napíše tri náhodne vybrané druhy „ovocia". Pri písaní programu ti pomôže nasledujúcich sedem rád.



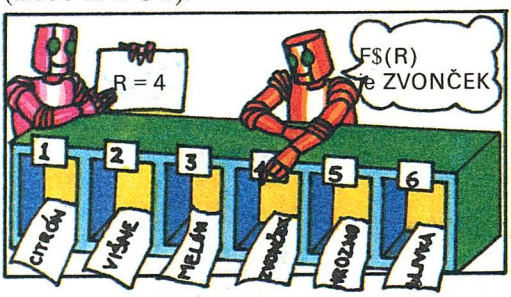
1. Najprv deklaruj pole F$, v ktorom zaznamenáš názvy šiestich druhov „ovocia" (citrón, višne, melón, hrozno, zvonček, slivka). Údaje ulož do poľa príkazom READ/DATA (alebo INPUT).



2. Vymaž obrazovku. Počet žetónov zaznamenaj v premennej T. Na začiatku jej priraď hodnotu 10 a hráčovi oznám, koľko má žetónov.

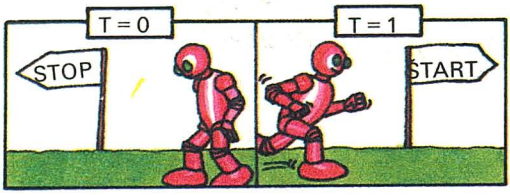


3. Aby počítač čakal, kým hráč stlačí ľubovoľný kláves, použi INKEY$. Od T odčítaj 1 (cena hry) a znova vymaž obrazovku.

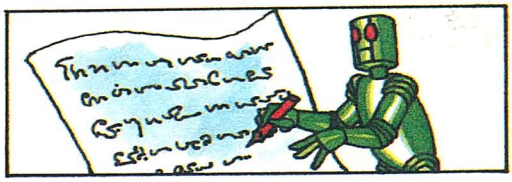


4. Počítač náhodne vyberie číslo od 1 do 6 a zaznamená ho v R. R použije ako index poľa F$ a údaj z poľa uloží do premennej A$. Zopakuje to ešte dvakrát, pričom názvy „ovocia" uloží do B$ a C$.

5. Vybrané druhy „ovocia" počítač napíše na obrazovku a rozhodne sa, či hráč niečo vyhráva, alebo nie. Tabuľka hore ukazuje, akými troma spôsobmi možno vyhrať. Pre každý typ výhry existuje jeden podprogram, ktorý napíše príslušnú správu a zmení hodnotu premennej T.

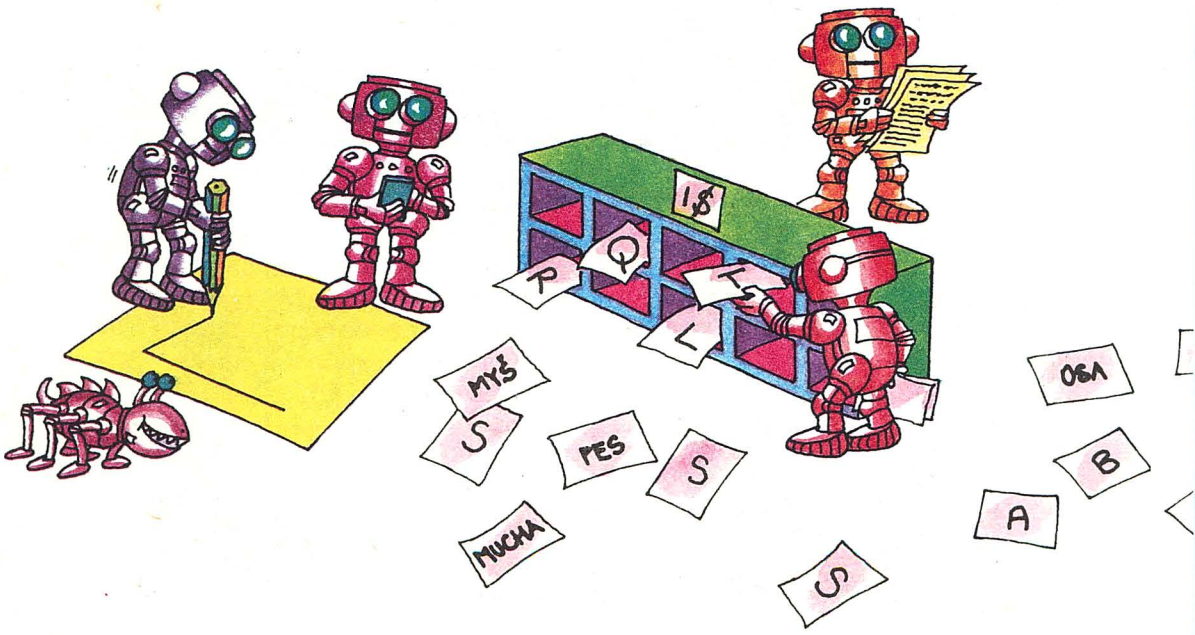


6. Skontroluj, či hráč má ešte aspoň jeden žetón. Ak nemá (T = 0), program sa končí. Inak sa začne ďalšia hra.

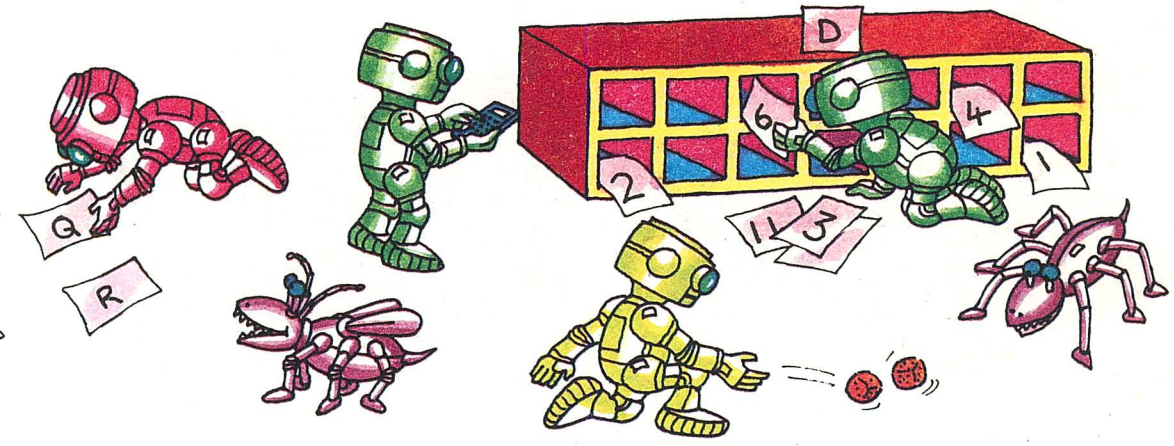


7. Ak si na načítanie údajov použil príkaz READ, nezabudni na koniec programu dať DATA a údaje.

Študujeme programy v basicu



Ak sa chceš naučiť dobre programovať, pomôže ti, ak budeš študovať hotové programy a budeš sa usilovať porozumieť im. V tejto časti knihy nájdeš niekoľko zložitejších programov s podrobným vysvetlením: vytvárajú banky údajov, kreslia grafy a obrázky, triedia údaje. Pri ich čítaní a spúšťaní možno pochopíš, ako sa v basicu náročnejšie programy robia. Programy sú napísané v basicu, ktorý nie je špecializovaný pre nijaký konkrétny typ počítača, a preto by mali fungovať na väčšine mikropočítačov. Ak si si nie istý, ktoré príkazy treba pre tvoj počítač zmeniť, preštuduj si tabuľku na strane 147. Podrobné vysvetlenie, ako treba meniť programy pre počítače Sinclair, nájdeš na strane 148.

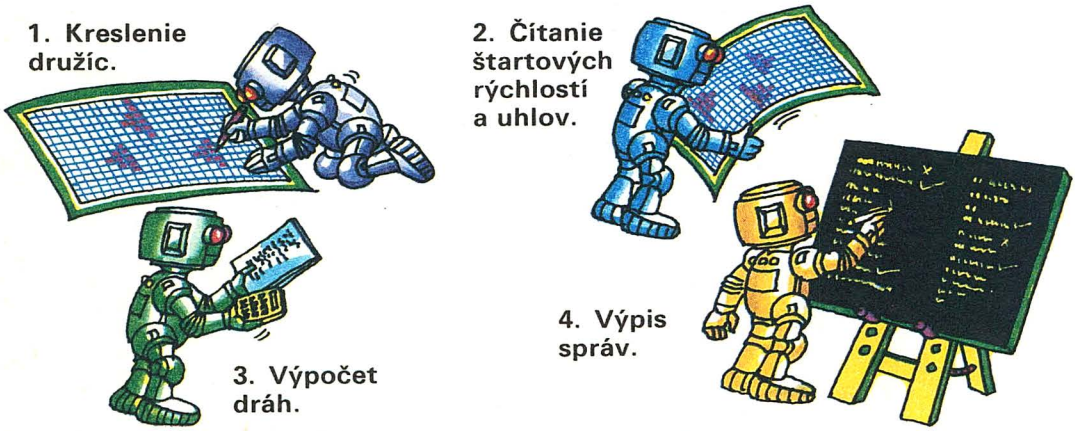


Rady, ako študovať programy

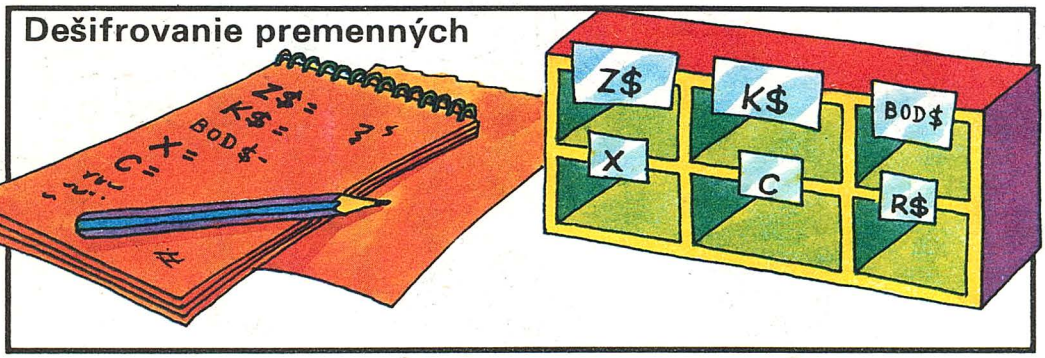
Niektoré programy vyzerajú na prvý pohľad dosť komplikovane. Nech ťa to však neodradí — uvedom si, že zložitý program nie je nič iné ako dlhý zoznam príkazov pospájaných istým spôsobom. Na tejto dvojstrane nájdeš zopár návodov a rád, ako programy študovať. Keď im porozumieš, skús experimentovať — zmeň ich tak, aby vykonávali trochu odlišné úlohy.

Študujeme programy

Programy sa väčšinou skladajú z niekoľkých nezávislých častí (podprogramov, modulov), ktoré vykonávajú rôzne úlohy. Napríklad v hre, kde štartujú družice, jedna časť programu kreslí na obrazovku družice, ďalšie časti čítajú štartové rýchlosti a uhly, počítajú dráhy letu a píšu správy.

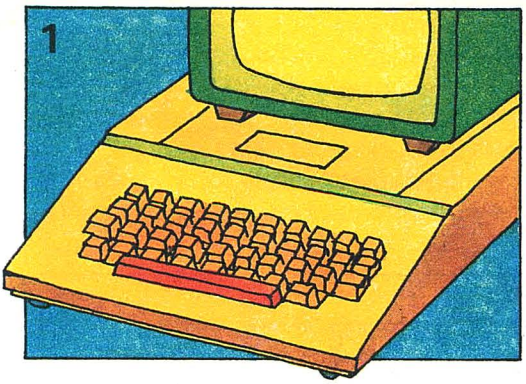


Pri čítaní programu sa najskôr usiluj nájsť takéto samostatné časti a prísť na to, čo vykonávajú — získaš tak predstavu o práci celého programu. Ako odhaľovať samostatné časti programu? Hľadaj podprogramy, riadky začínajúce sa slovom REM a veľké skoky v číslovaní riadkov (napríklad ak za riadkom 370 nasleduje riadok 500).

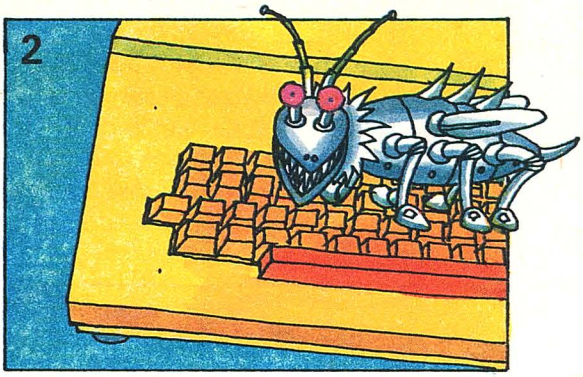


Pri čítaní programu je asi najťažšie porozumieť významu premenných. Je preto rozumné ešte pred napísaním programu do počítača vypátrať zmysel každej premennej a poznamenať si ho. Niektoré premenné by ti nemali robiť problémy, lebo sa používajú vždy na to isté. Napríklad I, J, K, L sa vyskytujú v cykloch, v Z a Z$ sa uchovávajú údaje potrebné len krátky čas.

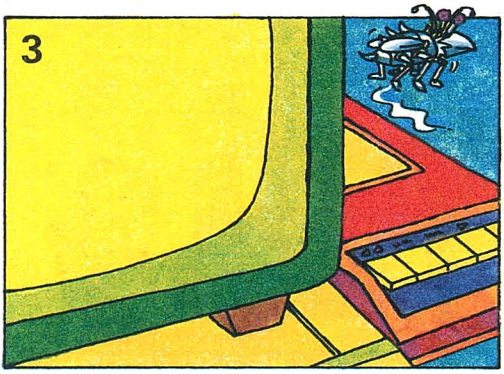
Spúšťanie a ladenie programov



Keď si zistil význam všetkých premenných, napíš program do počítača. Niektoré príkazy pravdepodobne budeš musieť zmeniť.

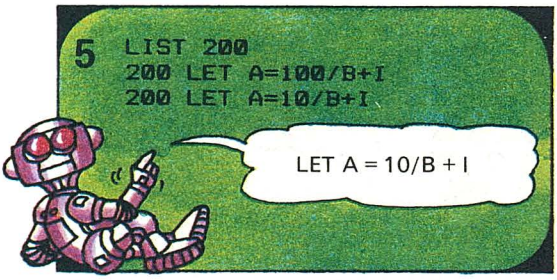


Program spusti. Iste v ňom budú chyby — nechaj si ho napísať na obrazovku a hľadaj prieklepy a príkazy, ktorým tvoj počítač nerozumie.



Keď si všetky chyby odstránil, program niekoľkokrát spusti a sleduj, ako pracuje. Nahraj ho na kazetu, aby si ho neskôr nemusel zase celý písať do počítača.

Porozmýšľaj, čo robia jednotlivé riadky programu, a urob si poznámky. Niektoré krátke programy možno neskôr využiješ vo svojich vlastných programoch.



Pri študovaní programu môžeš využiť aj počítač. Zmeň hodnotu jednej premennej a pozoruj, ako zmena ovplyvní celý program. Zakaždým sprav len jednu malú zmenu, aby si presne videl jej výsledok. Keď skončíš, nezabudni všade napísať pôvodné čísla. Ak chceš vidieť, ako sa hodnota premennej postupne mení, vsuň do programu príkaz, ktorý ju zakaždým napíše.

Niekedy je užitočné vsunúť aj príkaz STOP, aby si mal čas na premýšľanie. Niektoré počítače poznajú príkaz CONTINUE, ktorý môžeš použiť po príkaze STOP a znamená, že počítač má pokračovať vo vykonávaní programu. Nezabudni vsunuté príkazy vymazať.

Hra so slovami

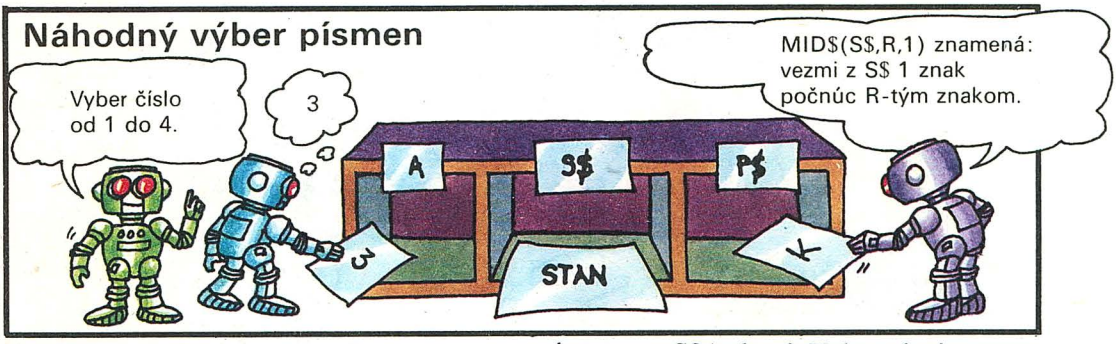
V programe Hra so slovami vidno, ako možno kombinovaním jednoduchých príkazov počítaču povedať, aby vykonával zložité úlohy. Počítač si najprv vyžiada slovo. Písmená zadaného slova napíše v náhodnom poradí po obrazovke a spýta sa, koľkokrát sa tam zadané slovo nachádza. V programe sa používajú príkazy pre prácu s reťazcami — MID$, RIGHT$, LEN, a náhodné čísla.*

Program má dve hlavné časti. V prvej sa na obrazovku píšu v náhodnom poradí písmená zadaného slova, v druhej sa počíta, koľkokrát sa slovo objavilo na obrazovke.

```
HRA SO SLOVAMI
NAPÍŠ KRÁTKE SLOVO
? STAN
SLEDUJ, KOĽKOKRÁT SA
TVOJE SLOVO OBJAVÍ
V POSTUPNOSTI PÍSMEN.
STLAČ RIADKOVAČ
```

```
S N A T T S N A A T S N N S S
T A N N A T S N A A S T A N A
S N A A S T N N S A T N S T A
KOĽKOKRÁT JE NA OBRAZOVKE
NAPÍSANÉ TVOJE SLOVO? 1
ZLE!
TVOJE SLOVO SOM NAPÍSAL 2-KRÁT
```

Náhodný výber písmen



Počítač vyberie písmeno príkazom MID$ s náhodným číslom. Zadané slovo je v S$. V riadku 160 náhodne vyberie číslo medzi 1 a počtom písmen slova a zaznamená ho v R. Podľa čísla R v riadku 170 rozhodne, ktoré písmeno z S$ vybrať. Vybrané písmeno zaznamená v P$ a v riadku 180 ho napíše na obrazovku. Pri každom opakovaní cyklu v riadkoch 150 až 220 uloží do R nové číslo a z S$ vyberie nové písmeno.

Hľadanie slova

Na začiatku programu počítač priradí premennej OVER$ reťazec z toľkých hviezdičiek, koľko písmen má zadané slovo. Vždy, keď náhodne vyberie nové písmeno, reťazec OVER$ skráti o prvý znak a na koniec pridá vybrané písmeno (riadok 200). V riadku 210 porovná OVER$ s S$, a ak sú písmená v rovnakom poradí, pripočíta k N jednotku.

* Návod, ako program prerobiť pre počítače Sinclair, je na strane 148.

Program Hra so slovami

```
10  GCLEAR
20  PRINT "HRA SO SLOVAMI": PRINT
30  LET OVER$=""
40  LET N=0
45  PRINT "NAPÍŠ KRÁTKE SLOVO"
50  INPUT S$
60  FOR I=1 TO LEN(S$)
70  LET OVER$=OVER$+"*"
80  NEXT I
90  PRINT
100 PRINT "SLEDUJ, KOĽKOKRÁT SA"
110 PRINT "TVOJE SLOVO OBJAVÍ"
115 PRINT "V POSTUPNOSTI PÍSMEN"
120 INPUT "STLAČ RIADKOVAČ";Z$
```

Na vymazanie obrazovky použi príkaz vhodný pre tvoj počítač.

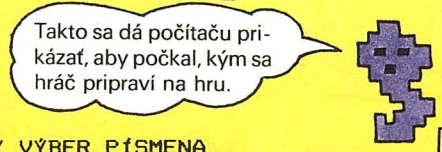
V riadku sú dva príkazy navzájom oddelené dvojbodkou.

Tieto premenné sa využijú ďalej v programe.

Počítač načíta slovo do premennej S$.

Cyklus zopakuje toľkokrát, koľko písmen má zadané slovo (teda LEN(S$)). Pri každom opakovaní cyklu k reťazcu OVER$ pripojí *.

Počítač čaká, kým hráč stlačí riadkovač.




```
130 GCLEAR
140 REM NÁHODNÝ VÝBER PÍSMENA
150 FOR I=1 TO 50*LEN(S$)
160 LET R=INT(RND(1)*LEN(S$)+1)
170 LET P$=MID$(S$,R,1)
```

Cyklus v riadkoch 150–220 zopakuje 50× počet písmen zadaného slova.

Do R uloží náhodne vybrané číslo z intervalu 1 až dĺžka zadaného slova.

Pomocou čísla v R vyberie písmeno z S$ a uloží ho do P$.

```
180 PRINT P$+" ";
190 REM HĽADANIE SLOVA
200 LET OVER$=RIGHT$(OVER$,LEN(S$)-1)+P$
210 IF OVER$ = S$ THEN LET N=N+1
220 NEXT I
```

Napíše písmeno z P$ a medzeru. Bodkočiarka znamená, že nasledujúce písmeno má napísať do toho istého riadka.

Vezme posledných LEN(S$)-1 písmen reťazca OVER$, pripojí k nim písmeno z P$ a takto utvorený reťazec uloží do OVER$.

V N uchováva, koľkokrát sa zadané slovo vyskytlo v napísanom reťazci.

```
230 FOR I=1 TO 1000
240 REM NIČ SA NEROBÍ
250 NEXT I
```

Zdržiavací cyklus. Nevykonáva sa v ňom nijaký príkaz. Počítač akoby niekoľko sekúnd stál, kým cyklus prejde cez všetky hodnoty I.

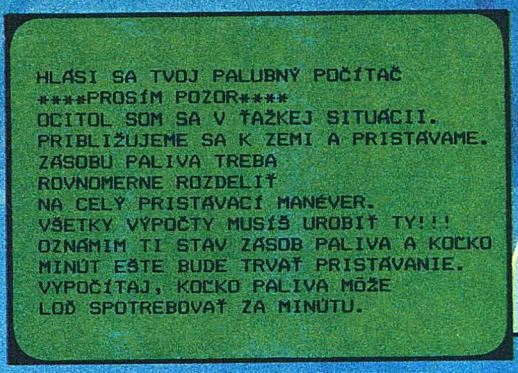
```
260 GCLEAR
265 PRINT "KOĽKOKRÁT JE NA OBRAZOVKE"
270 PRINT "NAPÍSANÉ TVOJE SLOVO?"
275 INPUT G
280 PRINT
290 IF G = N THEN PRINT "SPRÁVNE!"
300 IF G <> N THEN PRINT "ZLE!"
310 PRINT "TVOJE SLOVO SOM NAPÍSAL ";N;" KRÁT"
```

Hráčovu odpoveď uloží do G.

Počítač porovná G s N, teda s premennou, v ktorej počítal, koľkokrát napísal zadané slovo.

Pristátie medziplanetárnej lode

Táto hra vyskúša hráča z delenia. V programe nájdeš niekoľko spôsobov, ako využiť cyklus a náhodné čísla. Zaujímavo sa tu kreslia na obrazovku náhodné obrazce — možno sa ti to zíde v tvojich programoch. Program je pomerne dlhý, ale jednoduchý, lebo väčšina riadkov obsahuje iba príkaz PRINT.



Program Pristátie medziplanetárnej lode

```
100 GCLEAR
110 FOR I=1 TO 20
115 PRINT "**********POZOR**********"
118 PRINT "********POPLACH********"
120 FOR J=1 TO 10
125 REM NIČ
130 NEXT J
135 NEXT I

140 GCLEAR
150 FOR I=1 TO 20
155 PRINT "****POŠKODENÝ OBVOD****"
160 FOR J=1 TO 10
165 REM NIČ
170 NEXT J
180 NEXT I

190 GCLEAR
200 PRINT "HLÁSI SA TVOJ PALUBNÝ POČÍTAČ"
210 PRINT "**********PROSÍM POZOR********"
220 PRINT "OCITOL SOM SA V ŤAŽKEJ SITUÁCII.
    PRIBLIŽUJEME SA K ZEMI A PRISTÁVAME."
230 PRINT
240 PRINT "ZÁSOBU PALIVA TREBA
    ROVNOMERNE ROZDELIŤ
    NA CELÝ PRISTÁVACÍ MANÉVER."
250 PRINT
260 PRINT "VŠETKY VÝPOČTY MUSÍŠ UROBIŤ TY!!!"
265 PRINT
270 PRINT "OZNÁMIM TI STAV ZÁSOB PALIVA A KOĽKO
    MINÚT EŠTE BUDE TRVAŤ PRISTÁVANIE."
275 PRINT
280 PRINT "VYPOČÍTAJ, KOĽKO PALIVA MÔŽE
    LOĎ SPOTREBOVAŤ ZA MINÚTU."
290 PRINT
300 PRINT "VYSKÚŠAJME SI TO!"

310 PRINT "_________________"
320 PRINT "PALIVO=24"
330 PRINT "ČAS=6"
345 PRINT

350 INPUT "PROSÍM, VYDEĽ PALIVO ČASOM A
    ODPOVEDAJ RÝCHLO ";ODPOVEĎ
360 PRINT
370 IF ODPOVEĎ <> 4 THEN PRINT "ZLE, SKÚS EŠTE RAZ
    - ZÁVISÍ OD TOHO TVOJ ŽIVOT!":GOTO 350
380 GCLEAR
```

Obrazovku vymaž príkazom vhodným pre tvoj počítač.

Riadky 110 až 135 tvoria vnorený cyklus. Pri každom opakovaní cyklu I napíše počítač oznam o poplachu (riadky 115 a 118) a cyklus J zopakuje 10-krát. Cyklus J je zdržiavací — spôsobí, že počítač počká, kým si oznam stihneš prečítať.

Ak treba, zmeň číslo v zdržiavacom cykle.

Riadky 150–180 sú podobné riadkom 110–135.

Tu sa začína na obrazovku písať priebeh hry. Ak sú niektoré vety pre tvoju obrazovku príliš dlhé, rozdeľ ich a vlož do programu ďalšie riadky s príkazom PRINT.

Počítač napíše prázdny riadok.

Ak časť textu zmizne z obrazovky skôr, než si ho stihneš prečítať, vyhoď z programu riadky so samotným PRINT.

Podčiarkne napísaný text.

Hráčovu odpoveď uloží do ODPOVEĎ.

Počítač vykoná príkaz GOTO len vtedy, ak je hráčova odpoveď nesprávna. Ak tvoj počítač nesmie mať v riadku dva príkazy, pridaj ďalší riadok s IF/THEN a GOTO.

```
390 PRINT "DOBRE. ODTERAZ MUSÍŠ
    ODPOVEDAŤ VŽDY SPRÁVNE,"
395 PRINT "INAK SA LOĎ POŠKODÍ."
400 PRINT" AK SA POMÝLIŠ VIAC AKO DVAKRÁT,"
405 PRINT "LOĎ SA ÚPLNE ZNIČÍ."
410 PRINT
420 INPUT "AK SI PRIPRAVENÝ,
    STLAČ RIADKOVAČ!";Z$
430 GCLEAR
440 PRINT "POZOR - ZAČÍNAME!!!"
450 LET ŠKODA=0

460 FOR PALIVO=720 TO 120 STEP -120

470 LET ČAS=INT(RND(1)*5+2)

480 PRINT
490 PRINT "PALIVO=";PALIVO
500 PRINT "ČAS=";ČAS
510 PRINT

520 LET R=PALIVO/ČAS
530 INPUT "UDAJ SPOTREBU ZA MINÚTU";ODPOVEĎ
540 IF ODPOVEĎ = R THEN GOTO 600

550 LET ŠKODA=ŠKODA+1
560 PRINT
570 PRINT "***POŠKODENIE***"
580 PRINT
590 IF ŠKODA > 2 THEN GOTO 640
600 NEXT PALIVO
610 GCLEAR

620 PRINT "BLAHOŽELÁM - UROBIL SI
    VŠETKO, ČO BOLO V TVOJICH SILÁCH.
    ÚSPEŠNE SME PRISTÁLI!"
630 GOTO 720
640 GCLEAR

650 FOR I=1 TO 20
660 PRINT "*";
670 FOR J=1 TO INT(RND(1)*I+50)
680 PRINT "    ";
690 NEXT J
700 NEXT I
710 PRINT: PRINT "LOĎ JE ZNIČENÁ"
720 END
```

Počítač čaká, kým hráč stlačí riadkovač.

Premennú ŠKODA nastaví na nulu.

PALIVO je premenná cyklu a zároveň sa používa vo výpočtoch. V riadku 460 jej počítač priradí hodnotu 720 a v cykle ju postupne znižuje o 120. Tieto čísla boli vybrané tak, aby podiel paliva a času v riadku 520 bol vždy celé číslo.

Počítač náhodne vyberie číslo od 2 do 6. PALIVO bude vždy násobkom vybraného čísla.

Podiel PALIVA a ČASU uloží do R. Ak je hráčova odpoveď správna, program pokračuje riadkom 600.

V premennej ŠKODA ráta počet nesprávnych odpovedí.

Ak sa hráč pomýli viac ako dvakrát, počítač vyskočí z cyklu a pokračuje príkazom v riadku 640.

Napíše len vtedy, ak hráč spravil viac ako dve chyby.

Vnorené cykly. Počítač nakreslí na obrazovku náhodný obrazec. Pri každom opakovaní cyklu I napíše hviezdičku a pri každom opakovaní cyklu J napíše náhodný počet medzier.

Banka údajov o majstrovstvách sveta vo futbale

Banka údajov je súbor údajov uložených do pamäte počítača. Údaje sú usporiadané tak, aby v nich počítač mohol vyhľadávať viacerými spôsobmi a aby používateľ dostal požadovanú informáciu v čo najkratšom čase.

Program Majstrovstvá sveta je príkladom jednoduchej banky údajov, ktorá poskytuje jednak informácie o finalistoch majstrovstiev sveta vo futbale od roku 1930, a jednak prehľad o účasti mužstiev vo finále. Na konci je zopár rád, ako banku údajov zmeniť, aby sa v nej dali uchovávať iné informácie, napríklad zoznam časopisov, údaje o prírodovedných pozorovaniach. Skús do banky údajov pridať informácie o finalistoch posledných majstrovstiev sveta.

Program sa skladá z troch hlavných častí — uloženie údajov do pamäte, aktualizácia (obnova) údajov a „menu" (akýsi jedálny lístok činností, ktoré vie program vykonať, spomedzi ktorých si môžeš vyberať). Dôležité je, aby bol program „priateľský", čiže aby dal používateľovi jasné pokyny a nezastavil sa, ak sa používateľ pomýli.

Čo sa objaví na obrazovke

```
NAPÍŠ MENO TÍMU ALEBO MENU,
AK SI CHCEŠ POZRIEŤ PONUKU
NSR

NSR
MAJSTER SVETA ROKU 1954
FINALISTA ROKU 1966
MAJSTER SVETA ROKU 1974
FINALISTA ROKU 1982

PO STLAČENÍ RIADKOVAČA
SA OBJAVÍ MENU
```

```
NAPÍŠ ROK ALEBO MENU,
AK SI CHCEŠ POZRIEŤ PONUKU
1938

MAJSTER SVETA ROKU 1938
TALIANSKO

PO STLAČENÍ RIADKOVAČA
SA OBJAVÍ MENU
```

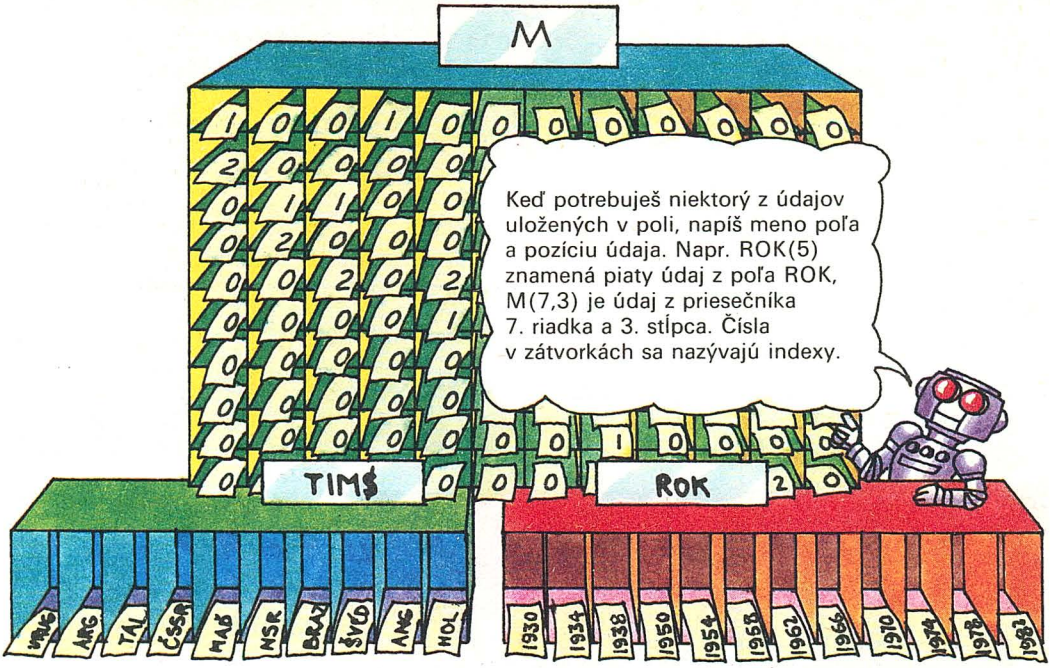
Uloženie údajov

	1930	1934	1938	1950	1954	1958	1962	1966	1970	1974	1978	1982
URUGUAJ	1	0	0	1	0	0	0	0	0	0	0	0
ARGENTÍNA	2	0	0	0	0	0	0	0	0	0	1	0
TALIANSKO	0	1	1	0	0	0	0	0	2	0	0	1
ČSFR	0	2	0	0	0	0	2	0	0	0	0	0
MAĎARSKO	0	0	2	0	2	0	0	0	0	0	0	0
SRN	0	0	0	0	1	0	0	2	0	1	0	2
BRAZÍLIA	0	0	0	0	0	1	1	0	1	0	0	0
ŠVÉDSKO	0	0	0	0	0	2	0	0	0	0	0	0
ANGLICKO	0	0	0	0	0	0	0	1	0	0	0	0
HOLANDSKO	0	0	0	0	0	0	0	0	0	2	2	0

Údaje sa v programe zaznamenávajú do tabuľky podobnej tejto a počítač v nej hľadá údaj podobne ako ty. Číslica 1 znamená víťazstvo, 2 účasť vo finále. Ak si pozrieš napríklad číslice v riadku Anglicko, zistíš, že tento tím bol víťazom iba roku 1966. Počítač vyhľadá informáciu (hlavne ak je údajov veľmi veľa) oveľa rýchlejšie ako ty.

Vytvorenie tabuľky

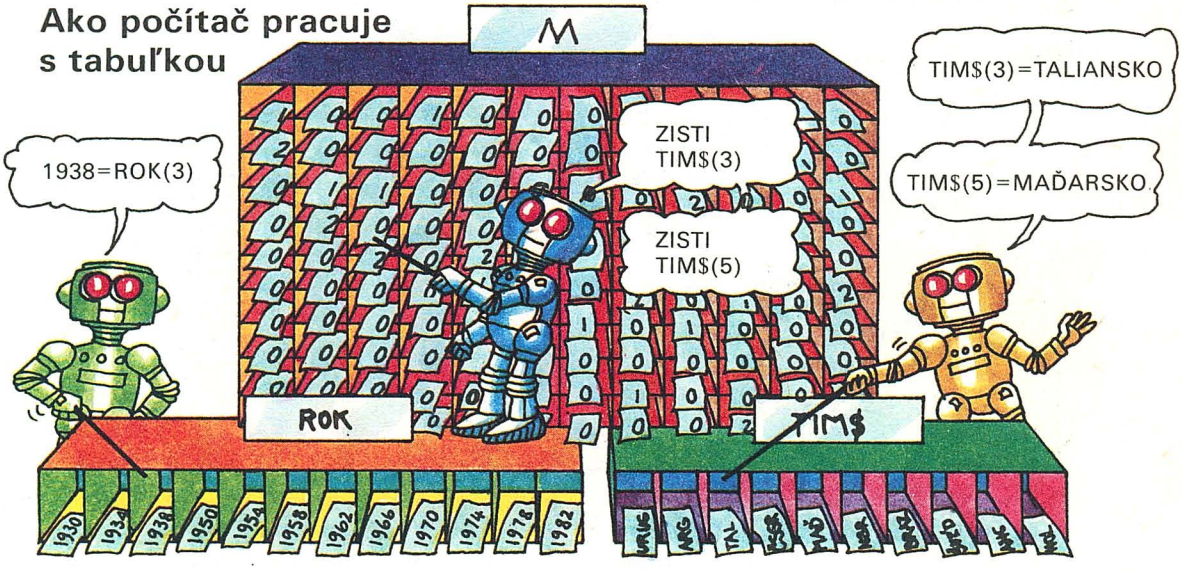
Tabuľka z predchádzajúcej strany sa do počítača uloží pomocou polí. Pole je premenná, ktorá obsahuje niekoľko nezávislých údajov.



Budeš potrebovať dve jednorozmerné polia — jedno s 12 položkami na zaznamenanie rokov (ROK) a druhé s 10 položkami pre mená tímov (TIM$).

Na zaznamenanie čísel z tabuľky potrebuješ pole s 10 riadkami a 12 stĺpcami. V programe má názov M (matica).

Ako počítač pracuje s tabuľkou



Účastníkov finále majstrovstiev sveta roku 1938 zistí počítač takto: v poli ROK nájde 1938 a zapamätá si, že je na treťom mieste. V treťom stĺpci tabuľky hľadá číslice 1 a 2. Poradové čísla nájdených riadkov (3 a 5) použije ako index poľa TIM$.

Program Majstrovstvá sveta

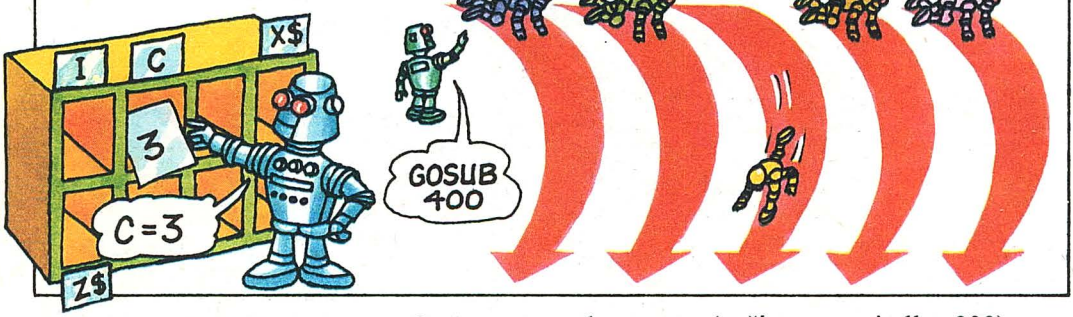
Program je rozdelený na sedem častí, z ktorých každú tvorí samostatný podprogram. V prvých riadkoch sa rozhoduje, ktorý podprogram má počítač vykonať. Po jeho vykonaní pokračuje nasledujúcim príkazom. Podprogramy, ktoré sa začínajú v riadkoch 200 a 300, napíšu roky konania majstrovstiev a zoznam mužstiev. V riadkoch 400 – 500 sa zisťuje, na ktorých majstrovstvách sa to-ktoré mužstvo prebojovalo do finále, a v riadkoch 500 – 600 finalisti tých-ktorých majstrovstiev. Na konci programu sú všetky údaje a za nimi nasleduje menu. V takýchto programoch sa kvôli prehľadnosti údaje zvyčajne umiestňujú na koniec a „výkonná časť" programu na začiatok.

```
MENU
****

1 - ZOZNAM ÚČASTNÍKOV
2 - ROKY KONANIA MAJSTROVSTIEV
3 - ZADÁVANIE MENA TÍMU
4 - ZADÁVANIE ROKU KONANIA
5 - KONIEC
NAPÍŠ ČÍSLO VYBRANEJ ČINNOSTI
A STLAČ RIADKOVAČ
```

Menu je časť programu, ktorá ponúka používateľovi zoznam možných činností spolu s návodom, ako si z nich vybrať. V našom programe sa činnosť zvolí napísaním čísla. Počítač uloží číslo do premennej C a použije ho pri výbere podprogramu, ktorý vykonáva požadovanú činnosť.

Vyvolanie podprogramov

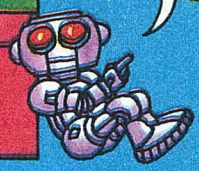
```
120 ON C GOSUB 200, 300, 400, 500, 600
```



O tom, ktorý podprogram počítač vykoná, sa rozhoduje v riadku 120. V premennej C je uložené číslo, ktoré napísal používateľ po prezretí menu. Počítač podľa neho vyberie podprogram. Ak C = 1, vykoná prvý z podprogramov vymenovaných v riadku 120, teda ten, ktorý sa začína v riadku 200. AK C = 2, vykoná druhý podprogram (začína sa v riadku 300), ak C = 3, vykoná tretí atď. Príkaz ON GOSUB má takýto význam: v závislosti od veľkosti hodnoty za ON vykonaj jeden z podprogramov. Ak tvoj počítač tento príkaz nepozná, nahraď ho niekoľkými príkazmi IF/THEN, napríklad IF C = 1 THEN GOSUB 200.

Aké premenné používame

ROK	TIM$	M
Jednorozmerné pole s rokmi majstrovstiev.	Jednorozmerné pole s menami účastníkov.	Dvojrozmerné pole s výsledkami.
C	Z$	X$
Do C sa ukladá číslo voľby v menu.	Meno zadaného tímu alebo rok.	Dočasné údaje.



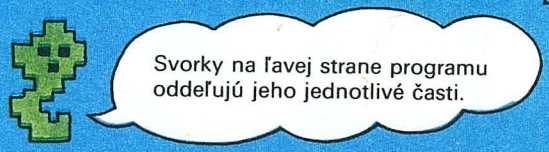
Program*

```
10  DIM TIM$(10):DIM ROK(12):DIM M(10,12)

100 GOSUB 1000: REM ČÍTA ÚDAJE

110 GOSUB 2000: REM PÍŠE MENU
```

Tieto príkazy povedia počítaču, koľko pamäte má vyhradiť pre polia.

Hneď po spustení programu počítač vykoná podprogram, ktorý sa začína riadkom 1000: načíta v ňom údaje.

Pokračuje príkazom v riadku 2000: vypíše menu.




```
120 ON C GOSUB 200,300,400,500,600
130 GOTO 110
```

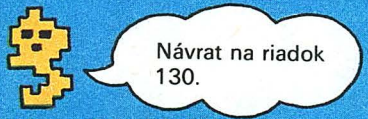
Počítač vyberie podprogram podľa čísla zadaného v menu. Po vykonaní podprogramu pokračuje riadkom 130, v ktorom dostane príkaz vrátiť sa na riadok 110 a znova napísať menu.

```
200 REM ZOZNAM ÚČASTNÍKOV
210 GCLEAR
220 PRINT "ZOZNAM ÚČASTNÍKOV": PRINT
    "_________________"
230 FOR I=1 TO 10: PRINT TIM$(I): NEXT I
235 PRINT
240 INPUT "PO STLAČENÍ RIADKOVAČA
    SA OBJAVÍ MENU";X$
250 RETURN
```

Podčiarkne slová ZOZNAM ÚČASTNÍKOV.

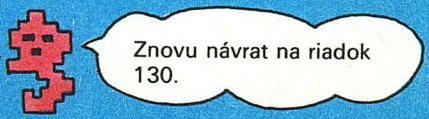
Cyklus, v ktorom napíše mená účastníkov. Pri každom opakovaní zvýši I o 1 a napíše ďalší údaj z poľa TIM$.

Program pokračuje až po stlačení riadkovača.




```
300 REM ROKY KONANIA MAJSTROVSTIEV
310 GCLEAR
320 PRINT "ROKY KONANIA MAJSTROVSTIEV":
    PRINT"_____________________________"
330 FOR I=1 TO 12: PRINT ROK(I): NEXT I
335 PRINT
340 PRINT "V 1942 A 1946 SÚŤAŽ NEBOLA"
345 PRINT
350 INPUT "PO STLAČENÍ RIADKOVAČA
    SA OBJAVÍ MENU";X$
360 RETURN
```

Cyklus, v ktorom počítač napíše roky konania majstrovstiev.



* Návod ako program prerobiť pre počítače Sinclair je na strane 148.

```
400 REM INFORMÁCIE O TÍME
405 GCLEAR
410 INPUT "NAPÍŠ MENO TÍMU ALEBO MENU,
    AK SI CHCEŠ POZRIEŤ PONUKU";Z$
415 IF Z$ = "MENU" THEN RETURN
420 FOR I=1 TO 10
425 IF Z$ = TIM$(I) THEN GOTO 440
430 NEXT I
435 PRINT: PRINT "O TOMTO TÍME NEMÁM
    INFORMÁCIE": PRINT: GOTO 410
440 PRINT
445 PRINT Z$: PRINT
```

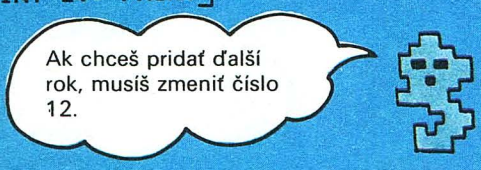
Meno tímu alebo slovo MENU uloží počítač do premennej Z$.

Ak Z$ = MENU, pokračuje riadkom 130 a napíše menu.

V cykle postupne porovnáva Z$ s menami v poli TIM$. V prípade rovnosti pokračuje riadkom 440.

Prípad, keď mužstvo nie je v banke údajov alebo vznikol prieklep.

Počítač napíše meno mužstva.




```
450 FOR J=1 TO 12
455 IF M(I,J) = 1 THEN PRINT "MAJSTER
    SVETA ROKU ";ROK(J): PRINT
460 IF M(I,J) = 2 THEN PRINT "FINALISTA
    ROKU ";ROK(J): PRINT
465 NEXT J
470 PRINT
480 INPUT "PO STLAČENÍ RIADKOVČA SA
    OBJAVÍ MENU ";X$
490 RETURN
```

V cykle hľadá v poli M umiestnenie zadaného mužstva. V premennej I je uložené číslo riadka. Hodnotu tejto premennej zistil v riadkoch 420–430, teraz ju použije ako index poľa. Číslo stĺpca je uložené v J. Pri každom opakovaní cyklu počítač testuje ďalší údaj z I-tého riadka.

To isté ako riadok 240.

```
500 REM INFORMÁCIE O MAJSTROVSTVÁCH
505 GCLEAR
510 INPUT "NAPÍŠ ROK ALEBO MENU, AK
    SI CHCEŠ POZRIEŤ PONUKU ";Z$
515 IF Z$ = "MENU" THEN RETURN
520 FOR I=1 TO 12
525 IF VAL(Z$) = ROK(I) THEN GOTO 540
530 NEXT I
535 PRINT: PRINT "V TOMTO ROKU SA MAJSTROVSTVÁ
    NEKONALI": PRINT: GOTO 510
```

To isté ako riadok 410, do Z$ uloží rok alebo slovo MENU.

V cykle porovnáva Z$ s rokmi v poli ROK. Pretože nemôže porovnávať reťazcovú a číselnú premennú, musíme použiť príkaz VAL, ktorý mu umožní narábať s reťazcom Z$ ako s číslom.

```
540  PRINT "MAJSTER SVETA ROKU ";Z$
550  FOR J=1 TO 10
555  IF M(J,I) = 1 THEN PRINT TIM$(J)
560  NEXT J

565  PRINT
570  INPUT "PO STLAČENÍ RIADKOVAČA SA
     OBJAVÍ MENU";X$
580  RETURN
```

Cyklus podobný ako cyklus v riadkoch 450–460. Tentokrát je číslom stĺpca index zadaného roka v poli ROK a číslo riadka sa pri každom opakovaní cyklu mení.

Návrat na riadok 130.

```
600  REM KONIEC PROGRAMU
610  INPUT "SKUTOČNE CHCEŠ SKONČIŤ?(A/N)";X$
620  IF X$ <> "A" THEN RETURN ELSE END
```

Počítač sa presvedčí, či naozaj chceš skončiť. Ak odpovieš A, vykoná príkaz END a program skončí. Ak odpovieš inak, pokračuje príkazom v riadku 130. Podrobnosti o príkaze IF/THEN-ELSE v riadku 620 sú na nasledujúcej strane.

Ak tvoj počítač nepozná slovo ELSE, príkaz END napíš do zvláštneho riadka.

```
1000 FOR I=1 TO 12: READ ROK(I): NEXT I
1010 DATA 1930,1934,1938,1950
1020 DATA 1954,1958,1962,1966
1030 DATA 1970,1974,1978,1982
```

Cyklus, v ktorom počítač načíta údaje do poľa ROK.

```
1100 FOR I=1 TO 10: READ TIM$(I): NEXT I
1110 DATA URUGUAJ,ARGENTÍNA
1115 DATA TALIANSKO,ČSSR,MAĎARSKO
1120 DATA NSR,BRAZÍLIA,ŠVÉDSKO
1125 DATA ANGLICKO,HOLANDSKO
```

Cyklus, v ktorom načíta údaje do poľa TIM$.

Údaje odpisuj pozorne. Každá vynechaná čiarka alebo číslo spôsobia v programe chybu.

```
1200 FOR I=1 TO 10: FOR J=1 TO 12
1205 READ M(I,J)
1210 NEXT J: NEXT I

1215 RETURN
```

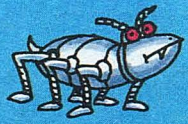
Vnorený cyklus, v ktorom načíta údaje do dvojrozmerného poľa M.

Tentokrát návrat na riadok 110.

```
1220 DATA 1,0,0,1,0,0,0,0,0,0,0,0
1230 DATA 2,0,0,0,0,0,0,0,0,0,1,0
1240 DATA 0,1,1,0,0,0,0,0,2,0,0,1
1250 DATA 0,2,0,0,0,0,2,0,0,0,0,0
1260 DATA 0,0,2,0,2,0,0,0,0,0,0,0
1270 DATA 0,0,0,0,1,0,0,2,0,1,0,2
1280 DATA 0,0,0,0,0,1,1,0,1,0,0,0
1290 DATA 0,0,0,0,0,2,0,0,0,0,0,0
1300 DATA 0,0,0,0,0,0,0,1,0,0,0,0
1310 DATA 0,0,0,0,0,0,0,0,0,2,2,0
```

Údaje pre M.

Údaje skontroluj najprv po riadkoch a potom po stĺpcoch. Ak zle napíšeš číslo, počítač bude poskytovať nesprávne informácie.



PROGRAM POKRAČUJE NA NASLEDUJÚCEJ STRANE.

```
2000 REM VÝPIS MENU
2010 GCLEAR
2020 PRINT "               MENU"
2030 PRINT "               ****"
2040 FOR I=1 TO 3: PRINT: NEXT I
2050 PRINT "1 - ZOZNAM ÚČASTNÍKOV"
2060 PRINT
2070 PRINT "2 - ROKY KONANIA MAJSTROVSTIEV"
2080 PRINT
2090 PRINT "3 - ZADÁVANIE MENA TÍMU"
2100 PRINT
2110 PRINT "4 - ZADÁVANIE ROKU KONANIA"
2120 PRINT
2130 PRINT "5 - KONIEC"
2140 PRINT
2150 PRINT "NAPÍŠ ČÍSLO VYBRANEJ ČINNOSTI"
2160 INPUT "A STLAČ RIADKOVAČ";C
2170 IF C < 1 OR C > 5 THEN PRINT
     "NAPÍŠ ČÍSLO OD 1 DO 5": GOTO 2150
2180 RETURN
```

Počítač vynechá 15 medzier, aby bol názov umiestnený v strede.

Vynechá 6 prázdnych riadkov.

Výpis menu. Zoznam ponúkaných možností musí byť jasný, aby používateľ presne vedel, čo má robiť.

Číslo vybranej činnosti uloží do C.

Prípad, keď zadané číslo nie je z intervalu 1—5. Slovo OR je vysvetlené nižšie.

Návrat na riadok 120, kde sa vyberie vhodný podprogram.

AND, OR a ELSE*

```
IF A = 3 AND C$ = "ÁNO" THEN LET D=D+1

IF X < 0 OR X > 100 THEN PRINT "MIMO ROZSAHU"

IF VEK < 36 THEN PRINT "MLADÝ" ELSE
PRINT "STARÝ"
```

Slová AND a OR sa používajú, ak v príkaze IF/THEN treba testovať súčasne viac podmienok. Slovo AND znamená, že príkaz za THEN sa vykoná len vtedy, ak sú obe podmienky splnené. Ak je v podmienke OR, na vykonanie príkazu za THEN stačí splniť jednu z podmienok. Príkaz za ELSE sa vykoná len vtedy, ak nie je splnená ani jedna z podmienok.

* Nie všetky počítače poznajú tieto slová basicu.

Hlavolam

Robot Zak ubehne 500 metrov za sekundu. Ak sa teplota zvýši nad 25 °C alebo klesne pod 25 °C, zvýši alebo zníži rýchlosť o 10 metrov za sekundu. Napíš program, ktorý bude vypisovať, koľko Zak ubehne za daný čas pri danej teplote! (Riešenie je na strane 143.)

Iné banky údajov

Keď si už porozumel, ako program pracuje, skús podobne vytvoriť ďalšie banky údajov. Nemalo by ti to robiť žiadne problémy. Vyskúšaj si niektorú z nasledujúcich bánk údajov.

Už vieš, akú banku údajov chceš vytvoriť? Najprv si zapíš všetky údaje do tabuľky podobnej tabuľke na strane 46. Ak tvoja tabuľka bude mať iný počet stĺpcov a riadkov, musíš v programe zmeniť rozmery polí. Zmeň vstupné údaje a vytvor si vlastné menu. Nezabudni zmeniť aj príkazy DIM a počet opakovaní cyklov, v ktorých sa údaje načítavajú.

Časopisecká banka údajov

TÉMA$

ČÍSLO$ | TABUĽKA

Vytvor banku údajov obsahujúcu témy článkov v nejakom časopise a prehľad o tom, v ktorých číslach sa články na tú-ktorú tému nachádzajú. Banka ti ušetrí hodinové prehľadávanie starých časopisov. Budeš potrebovať polia ČÍSLO$, TÉMA$ a tabuľku.

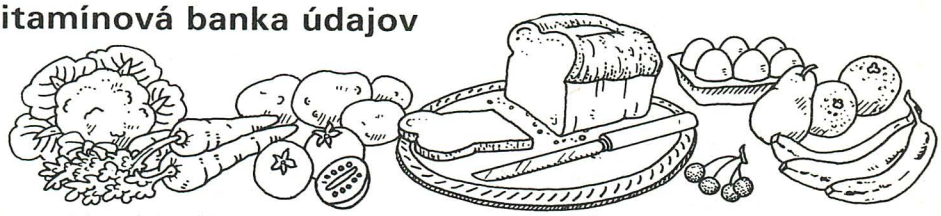
Prírodovedecká banka údajov

ŽIVOČÍCH$

MIESTO$ | TABUĽKA

Banka obsahuje údaje o živočíchoch, ktoré si videl, a o tom, kde si ich pozoroval. Potrebuješ pole na zaznamenanie mien živočíchov a ďalšie pre miesta. Podobne si môžeš urobiť banku údajov o tom, kedy kvitnú rastliny. Prírodovedci vytvárajú podobné banky údajov — zaznamenávajú v nich napr. rozšírenia jednotlivých druhov rastlín a živočíchov.

Vitamínová banka údajov



POTRAVINA$

VITAMÍN$ | TABUĽKA

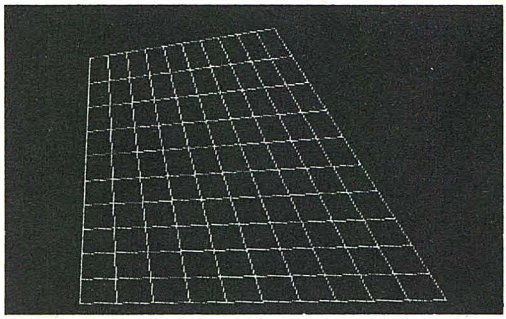
Mala by poskytovať tieto údaje: v ktorých potravinách sa vyskytuje určitý vitamín a aké vitamíny obsahujú jednotlivé potraviny. Podobne by sa dala vytvoriť banka údajov o jouloch. Informovala by o energetickej hodnote potravín a o potravinách, ktoré obsahujú viac joulov, než je žiadúce.

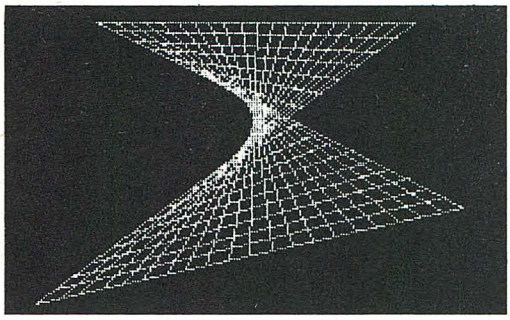
Jednoduchá grafika

Nasledujúci program nakreslí na obrazovku jednoduchý obrázok a vyšrafuje ho vodorovnými a zvislými čiarami. Šrafovaním sa v počítačovej grafike často vytvára dojem trojrozmernosti.

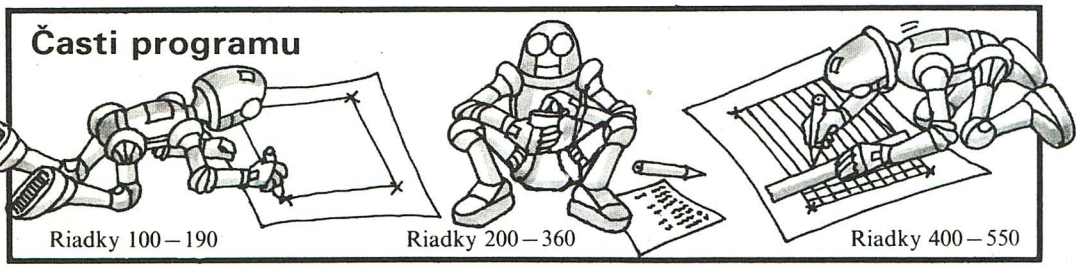
V programe sa používajú grafické príkazy PLOT X, Y na kreslenie bodu a DRAW X, Y na kreslenie čiary. Súradnice X, Y sa merajú od okraja obrazovky. Príkazy pravdepodobne budeš musieť prispôsobiť svojmu počítaču a možno budeš musieť pridať ďalšie grafické príkazy.*

Existujú dva spôsoby písania grafických programov. Počítač buď rad za radom vypočítava súradnice a obrázok postupne kreslí na obrazovku, alebo najprv urobí všetky výpočty, výsledky uloží do poľa a celý obrázok nakreslí naraz. Náš program je napísaný druhým spôsobom.

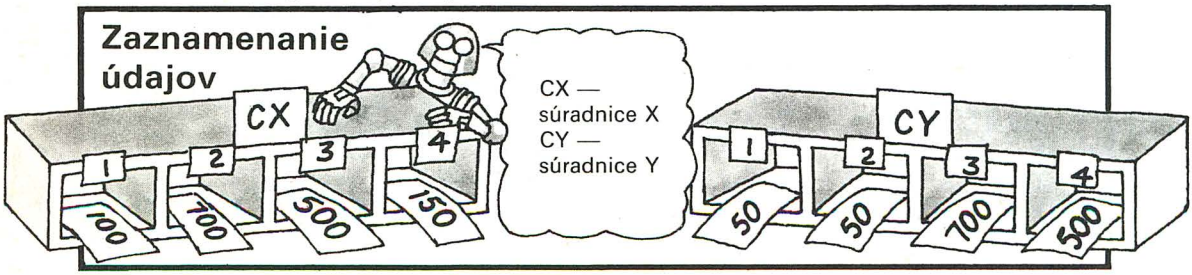




Tvar obrázka závisí od vstupných údajov napísaných v programe. Ak na tvojom počítači možno kresliť farebne, obrázky vypĺňaj rôznofarebnými čiarami. Rady, ako program zmeniť, nájdeš na konci programu.

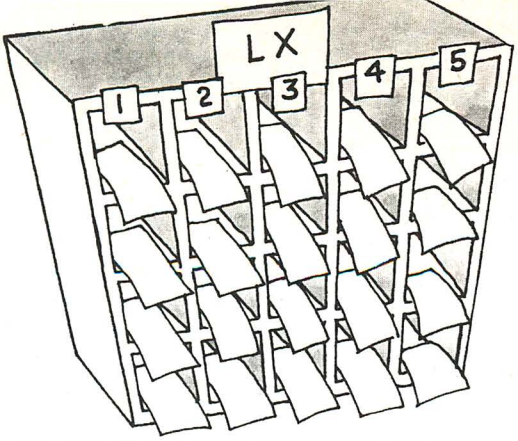


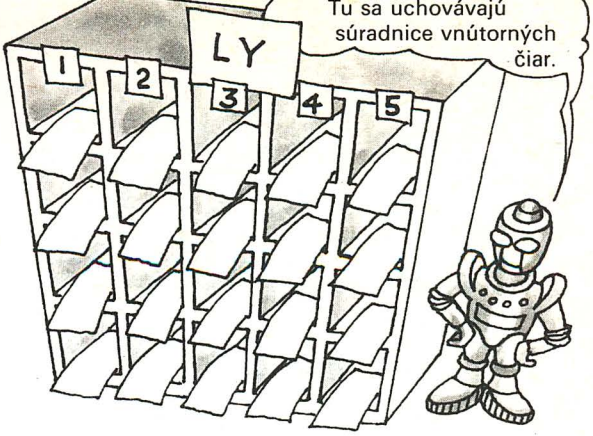
Program sa skladá z troch hlavných častí. V prvej (riadky 100 – 190) sa nakreslia vrcholy obrázka a spoja sa čiarami. V druhej časti (riadky 200 – 360) sa vypočítajú súradnice čiar, ktoré vypĺňajú obrázok, a v tretej časti sa tieto čiary nakreslia.



Na zaznamenanie všetkých súradníc sa v programe používajú štyri polia. X-ové a y-ové súradnice vrcholov sú uložené v poliach CX a CY. Ich hodnoty sa zadávajú hneď na začiatku. CX(1) a CY(1) obsahujú súradnice prvého vrcholu, CX(2) a CY(2) druhého atď.

* Na niektorých počítačoch, napríklad na ZX Spectrum, sa x-ové a y-ové súradnice merajú od posledného nakresleného bodu. Návod na prerobenie programu je na strane 149.





V poliach LX a LY sú uložené súradnice čiar, ktoré vypĺňajú obrázok. LX a LY sú dvojrozmerné polia, každé má štyri riadky. Počet stĺpcov sa určí v programe podľa požadovaného počtu vypĺňajúcich čiar. V jednom riadku sú uložené súradnice priesečníkov vnútorných čiar s jednou stranou — v prvom riadku poľa LX x-ové súradnice priesečníkov s prvou stranou atď. Hodnoty súradníc sa do LX a LY ukladajú postupne — tak, ako sa vypočítavajú.

Program

```
100 INPUT "KOĽKO ČIAR MÁ
    VYPĽŇAŤ OBRÁZOK? ";N

110 DIM CX(4),CY(4)
115 DIM LX(4,N),LY(4,N)

120 GCLEAR
130 REM KRESLENIE STRÁN OBRÁZKA
135 REM VSUŇ GRAFICKÉ PRÍKAZY POTREBNÉ
    PRE TVOJ POČÍTAČ

140 FOR I=1 TO 4
145 READ CX(I),CY(I)
150 NEXT I

160 DATA 100,50,700,50,500,700,150,500

170 PLOT CX(4),CY(4)

180 FOR I=1 TO 4
185 DRAW CX(I),CY(I)
190 NEXT I
```

Pre počítače s vyššou rozlišovacou schopnosťou skús 20 čiar, pre ostatné 5.

Tieto príkazy povedia počítaču, koľko pamäte má vyhradiť pre polia.

Toto sú súradnice ľavého obrázka z predchádzajúcej strany. Ak zmeníš čísla, nakreslí sa iný obrázok.

V cykle počítač načíta súradnice vrcholov obrázka. Pri každom opakovaní cyklu uloží do polí CX a CY nasledujúce dve čísla z riadka 160.

Súradnice vrcholov. Ak sa ti obrázok nezmestí na obrazovku, zmeň ich.

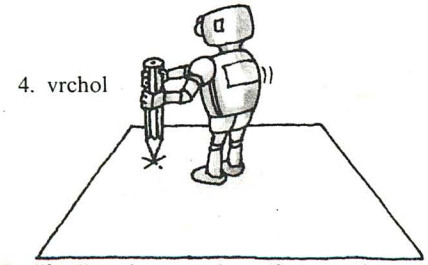
Ak treba, príkazy PLOT a DRAW nahraď príkazmi vhodnými pre tvoj počítač.

Počítač nakreslí 4. vrchol, určený súradnicami CX(4) a CY(4).

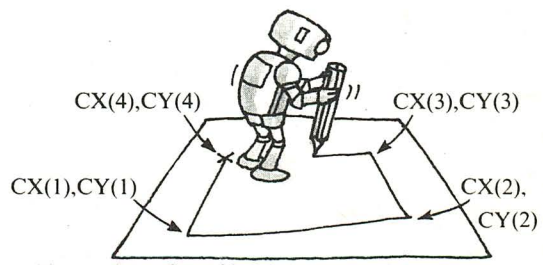
V cykle nakreslí strany obrázka.

PROGRAM POKRAČUJE NA NASLEDUJÚCEJ STRANE.

Kreslenie strán



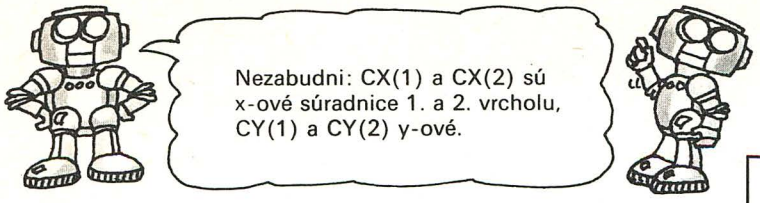
Počítač najprv nakreslí 4. vrchol (riadok 170). Pri prvom opakovaní cyklu v riadkoch 180—190 nakreslí



čiaru, ktorá spája 4. a 1. vrchol. Potom zvýši hodnotu premennej I a nakreslí čiaru, ktorá spája ďalšie dva vrcholy.

```
200 REM VÝPOČET SÚRADNÍC VNÚTORNÝCH ČIAR
```

Táto časť programu obsahuje štyri cykly na výpočet súradníc vnútorných čiar. Postup výpočtu je znázornený na obrázkoch nižšie.



N je zadaný počet vnútorných čiar.

```
210 FOR I=1 TO N
```

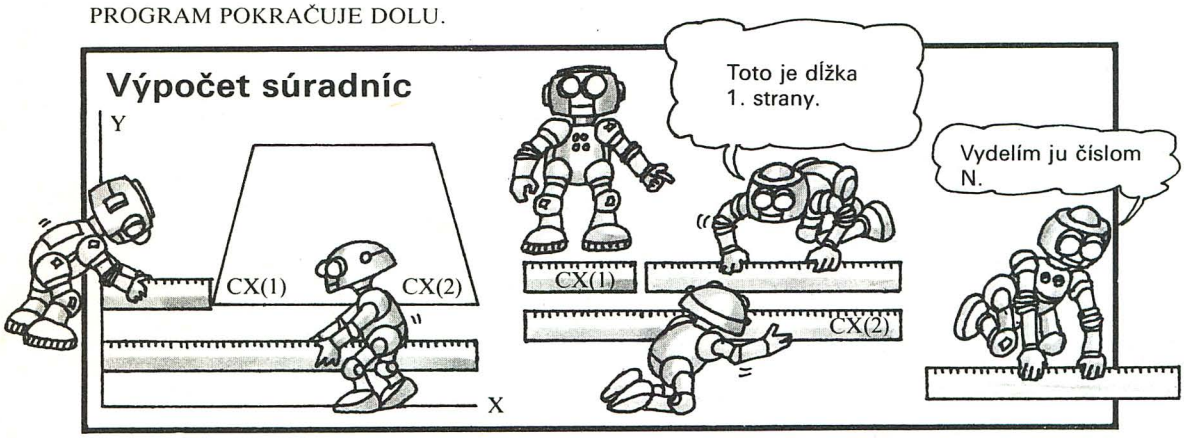
Počítač vypočíta x-ové súradnice bodov, v ktorých sa pretínajú vnútorné čiary a 1. strana. Výsledky uloží do 1. riadka a 1. až N-tého stĺpca poľa LX.

```
220 LET LX(1,I)=CX(1)+(CX(2)-CX(1))*I/N
```

Vypočíta y-ové súradnice tých istých bodov a uloží ich do 1. riadka a 1. až N-tého stĺpca poľa LY.

```
230 LET LY(1,I)=CY(1)+(CY(2)-CY(1))*I/N
240 NEXT I
```

PROGRAM POKRAČUJE DOLU.



Počítač v každom cykle vypočíta súradnice bodov, ktoré ležia na jednej zo strán obrázka. Napríklad v riadkoch 210 – 240 sú to body na 1. strane. V riadku 220 od CX(2) odčíta CX(1). Ich rozdiel je dĺžka 1. strany obrázka. Pri prvom opakovaní cyklu vynásobí dĺžku strany číslom 1/N. Ak N má hodnotu 5, výsledkom násobenia bude 1/5 dĺžky strany. Tú pripočíta k hodnote CX(1) a výsledok uloží do premennej LX(1,1). Pri druhom opakovaní cyklu bude I = 2, takže vypočíta 2/5 dĺžky a výsledok uloží do LX(1,2). Cyklus zopakuje pre všetky I od 1 po N a vypočíta x-ové súradnice všetkých bodov na 1. strane. Podobne v riadku 230 vypočíta y-ové súradnice.

```
250 FOR I=1 TO N
260 LET LX(3,I)=CX(4)+(CX(3)-CX(4))*I/N
270 LET LY(3,I)=CX(4)+(CY(3)-CY(4))*I/N
280 NEXT I
```

V cykle vypočíta súradnice bodov, v ktorých sa pretínajú vnútorné čiary a 3. strana. Aby boli uložené v tom istom poradí, teda zľava doprava, od CX(4) odčíta CX(3) a súčin pripočíta k CX(4).

Skús zmeniť cyklus tak, aby sa podobal ostatným cyklom, a pozoruj, čo sa stane.

```
290 FOR I=1 TO N
300 LET LX(2,I)=CX(2)+(CX(3)-CX(2))*I/N
310 LET LY(2,I)=CY(2)+(CY(3)-CY(2))*I/N
320 NEXT I
```

Vypočíta súradnice bodov na 2. strane.

PROGRAM POKRAČUJE NA ĎALŠEJ STRANE.

```
330 FOR I=1 TO N
340 LET LX(4,I)=CX(1)+(CX(4)-CX(1))*I/N
350 LET LY(4,I)=CY(1)+(CY(4)-CY(1))*I/N
360 NEXT I
400 REM KRESLENIE VNÚTORNÝCH ČIAR
410 LET RI=1: GOSUB 500
420 LET RI=2: GOSUB 500
430 STOP
500 REM PODPROGRAM
510 FOR I=1 TO N
520 PLOT LX(RI,I),LY(RI,I)
530 DRAW LX(RI+2,I),LY(RI+2,I)
540 NEXT I
550 RETURN
```

Počítač vypočíta súradnice bodov na 4. strane.

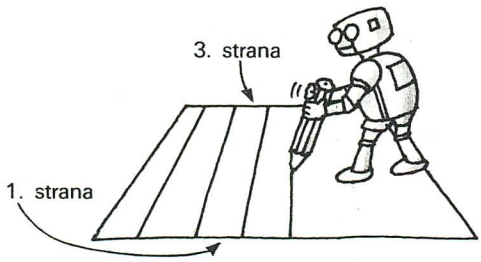
Premennej RI priradí hodnotu 1 a vykoná podprogram, ktorý sa začína riadkom 500. Hodnotu RI zmení na 2 a opäť vykoná ten istý podprogram.

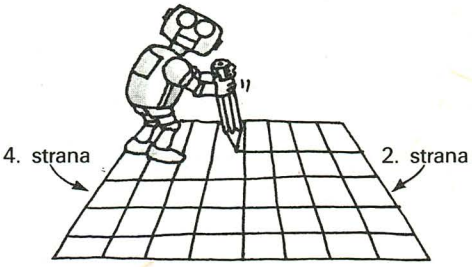
Po druhom vykonaní podprogramu sa program končí.

RI a I sú indexy polí LX a LY, RI určuje riadok a I stĺpec. Pomocou nich počítač zistí súradnice koncových bodov kreslenej čiary.

Pri prvom vykonávaní podprogramu sa vráti na riadok 420, pri druhom na riadok 430.

Kreslenie vnútorných čiar





Pri prvom vykonávaní podprogramu je RI = 1 a v riadku 520 počítač nakreslí bod na 1. strane obrázka. V riadku 530 k RI pripočíta 2 a nakreslí čiaru, ktorá spája 1. a 3. stranu. Pri druhom opakovaní podprogramu je RI = 2 a počítač kreslí čiary spájajúce 2. a 4. stranu.

Ako zmeniť program

1. Ak chceš, aby počítač kreslil iný obrázok, musíš zmeniť súradnice vrcholov v riadku 160. Nezabudni, že prvé dve čísla sú súradnice prvého vrcholu, druhé dve sú súradnice druhého vrcholu atď. Ak zadáš dva rovnaké vrcholy, počítač nakreslí trojuholník. Skús naprogramovať obrázok, ktorý je na strane 54 vpravo.
2. V cykle môžeš použiť príkaz INPUT tak, aby si počítač pýtal údaje. Nahraď riadky 140 – 160 takto:

```
140 FOR I=1 TO 4
150 PRINT "AKÉ SÚ SÚRADNICE VRCHOLU ";I
155 INPUT CX(I),CY(I)
160 NEXT I
```

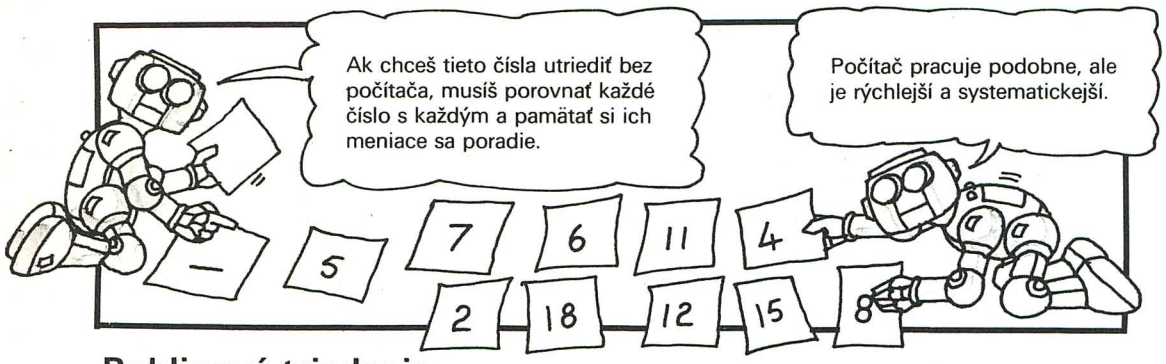
3. Obrázok vypĺňaj rôznofarebnými čiarami. Pred príkaz GOSUB v riadkoch 410 a 420 napíš príkaz vhodný pre tvoj počítač. Vsunutý príkaz nezabudni od príkazu GOSUB oddeliť dvojbodkou.

Triedenie

Z času na čas potrebujeme všeličo utriediť — napríklad abecedne zoradiť knihy či mená svojich priateľov alebo chronologicky usporiadať dátumy historických udalostí. Pokiaľ nie je údajov veľa, ľahko to spravíme sami. Keď však ich počet narastie, počítač to urobí rýchlejšie a presnejšie.

Programy na triedenie údajov sa nazývajú triediace. Dajú sa použiť pre rôzne úlohy. Uplatňujú sa v nich viaceré programátorské postupy.

Na nasledujúcej strane nájdeš programy založené na dvoch postupoch. Prvý sa nazýva bublinové triedenie (neskôr sa dozvieš, prečo) a druhý Shellovo triedenie (čítaj šelovo) podľa matematika, ktorý ho vymyslel. Bublinové triedenie je veľmi pomalé, a tak sa hodí len pre malé množstvo údajov. Shellovo triedenie je oveľa rýchlejšie. Na strane 63 je niekoľko riadkov, ktoré môžeš vložiť do programov, aby si ich mohol navzájom porovnať a aby si videl, aký rýchly je tvoj počítač.



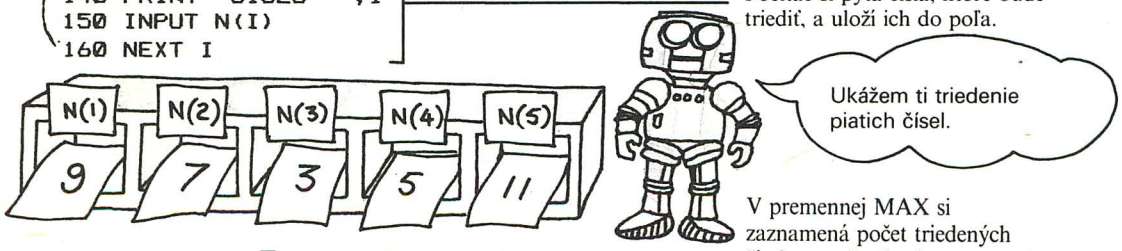
Bublinové triedenie

Pri bublinovom triedení vezme počítač prvé dva členy z neutriedenej postupnosti a porovná ich. Ak sú v nesprávnom poradí, navzájom ich vymení. Potom porovná druhý a tretí člen a takto pokračuje, až kým nepríde na koniec postupnosti. Najväčší člen pritom „prebuble" na koniec postupnosti. Tu je program na bublinové triedenie čísel a za ním program na bublinové triedenie slov.

```
100 REM BUBLINOVÉ TRIEDENIE ČÍSEL
110 INPUT "KOĽKO ČÍSEL CHCEŠ
    UTRIEDIŤ? ";T
120 DIM N(T)
130 FOR I=1 TO T
140 PRINT "ČÍSLO  ";I
150 INPUT N(I)
160 NEXT I
```

Deklarácia poľa N s T prvkami. T je počet čísel, ktoré treba utriediť.

Počítač si pýta čísla, ktoré bude triediť, a uloží ich do poľa.



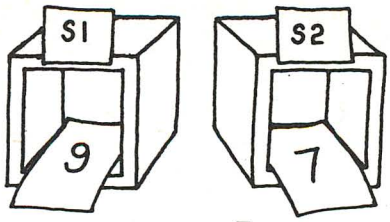

```
170 LET MAX=T
```

V premennej MAX si zaznamená počet triedených čísel, pretože obsah premennej T sa počas triedenia mení.

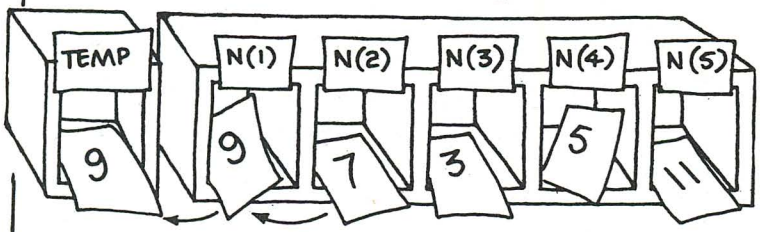
```
175 LET X=0
```

X je počítadlo.

```
180 FOR C=1 TO T-1
190 LET S1=N(C): LET S2=N(C+1)
```

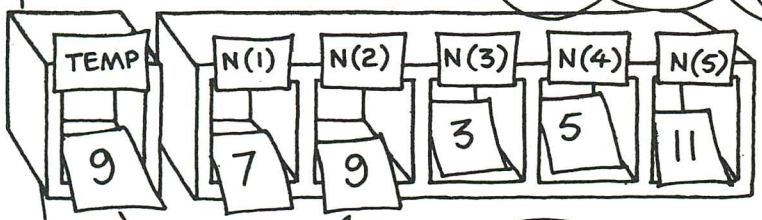


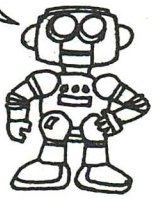

```
200 IF S1 <= S2 THEN GOTO 250
```




```
210 LET POM=N(C)
220 LET N(C)=N(C+1)
230 LET N(C+1)=POM
```

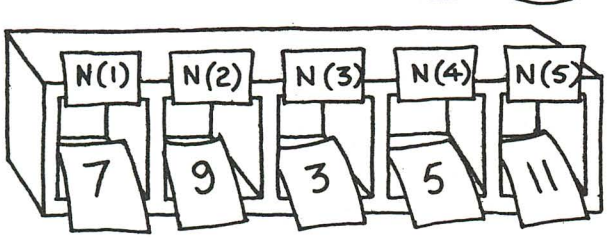
Pri každom opakovaní cyklu sa číslo posúva doprava, až kým sa nedostane na správne miesto.

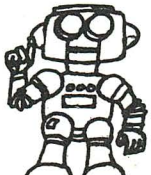




```
240 LET X=X+1
250 NEXT C
```

Pri druhom opakovaní cyklu je C=2, takže N(C)=N(2) a N(C+1)=N(3).





```
260 IF X > 0 THEN LET T=T-1: GOTO 175
270 PRINT "UTRIEDENÁ POSTUPNOSŤ"
280 FOR I=1 TO MAX
290 PRINT N(I)
300 NEXT I
```

Ak príkaz GOTO z riadka 260 dávaš do nového riadka, použi príkaz IF/THEN.

Počítač zopakuje cyklus T-1 krát, čo je počet dvojíc za sebou idúcich členov postupnosti.

V premenných S1, S2 sú uložené práve porovnávané čísla. Pri prvom opakovaní cyklu je C=1 a počítač do S1 a S2 uloží N(1) a N(2)

Porovnávanie dvoch čísel: Ak S1 je menšie než S2, čísla sú v správnom poradí, počítač začne znovu vykonávať prvý príkaz cyklu a porovná ďalšiu dvojicu. Ak S1 je väčšie než S2, vykoná nasledujúce príkazy — v postupnosti vymení pozície čísel S1 a S2.

Hodnotu premennej N(C) uloží do POM (skratka pomocnej premennej).

Hodnotu premennej N(C+1) presunie do N(C).

Hodnotu premennej POM uloží do N(C+1).

Po každej výmene zvýši hodnotu X.

Návrat — porovná ďalšiu dvojicu čísel. Po T-1 zopakovaniach cyklu porovnal všetky dvojice a pokračuje riadkom 260.

Keď X je väčšie ako 0 (teda počítač aspoň raz menil poradie čísel), zníži hodnotu T o 1, pretože jeden člen postupnosti je na správnom mieste. Znovu zopakuje celý cyklus. Ak X=0, čísla sú v správnom poradí a pokračuje riadkom 270.

MAX je počet triedených čísel.

Bublinové triedenie slov

Program Bublinové triedenie slov je podobný predchádzajúcemu. Rozdiel je iba v tom, že premenné na uchovávanie údajov (N, S1, S2, POM) sú reťazcové.*

Na porovnávanie slov sa používa tá istá metóda ako na porovnávanie čísel. Vnútri počítača sú písmená a iné symboly kódované číslami, takže pri porovnávaní znakov počítač vlastne porovnáva ich číselné kódy. Pri porovnávaní slov skontroluje najprv ich prvé písmená. Ak sú rovnaké, porovná druhé, potom tretie atď. Do reťazcových premenných môžeš ukladať nielen písmená, ale aj čísla. Program Bublinové triedenie slov preto môžeš použiť pre údaje s číslami aj s písmenami (napríklad pre adresy).

```
KOĽKO ÚDAJOV CHCEŠ UTRIEDIŤ? 4
1. ÚDAJ
?ELEKTRÓN
2. ÚDAJ
?MLADÉ ROZLETY
3. ÚDAJ
?KRÁSY SLOVENSKA
4. ÚDAJ
?ROHÁČ
UTRIEDENÝ ZOZNAM
ELEKTRÓN
KRÁSY SLOVENSKA
MLADÉ ROZLETY
ROHÁČ
```

```
KOĽKO ÚDAJOV CHCEŠ UTRIEDIŤ? 4
1. ÚDAJ
?MIŠÍKOVÁ MÁRIA  OKRUŽNÁ 12
2. ÚDAJ
?KOVÁČ PETER  ZÁHRADNÁ 3
3. ÚDAJ
?RAK JURAJ  HLAVNÁ 5/III
4. ÚDAJ
?BENKOVÁ KATKA  PÚPAVOVÁ 172
UTRIEDENÝ ZOZNAM
BENKOVÁ KATKA PÚPAVOVÁ 172
KOVÁČ PETER ZÁHRADNÁ 3
MIŠÍKOVÁ MÁRIA OKRUŽNÁ12
RAK JURAJ HLAVNÁ 5/III
```

Program ti utriedi časopisy alebo adresár. Údaje sú napísané bez čiarok, pretože na väčšine počítačov čiarky oddeľujú rôzne časti údajov. Ak chceš mať vo vstupných reťazcoch čiarky, použi úvodzovky.

Program

```
100 REM BUBLINOVÉ TRIEDENIE SLOV
110 INPUT "KOĽKO ÚDAJOV CHCEŠ UTRIEDIŤ? ";T
120 DIM N$(T)

130 FOR I=1 TO T
140 PRINT I;". ÚDAJ"
150 INPUT N$(I)
160 NEXT I

170 LET MAX=T
175 LET X=0

180 FOR C=1 TO T-1
190 LET S1$=N$(C): LET S2$=N$(C+1)
200 IF S1$ <= S2$ THEN GOTO 250
210 LET POM$=N$(C)
220 LET N$(C)=N$(C+1)
230 LET N$(C+1)=POM$
240 LET X=X+1
250 NEXT C

260 IF X > 0 THEN LET T=T-1: GOTO 175

270 PRINT "UTRIEDENÝ ZOZNAM"

280 FOR I=1 TO MAX
290 PRINT N$(I)
300 NEXT I
```

Deklarácia poľa N$ s T položkami.

Skús ako vstupy zadať reťazce začínajúce sa číslami, malými a veľkými písmenami či inými symbolmi a sleduj, ako ich počítač utriedi.

Prvé dva údaje počítač uloží do S1$ a S2$.

Porovná S1$ a S2$.

Vymení miesta prvých dvoch údajov v N$.

Ak X>0, zníži hodnotu X o 1 a vráti sa na začiatok cyklu.

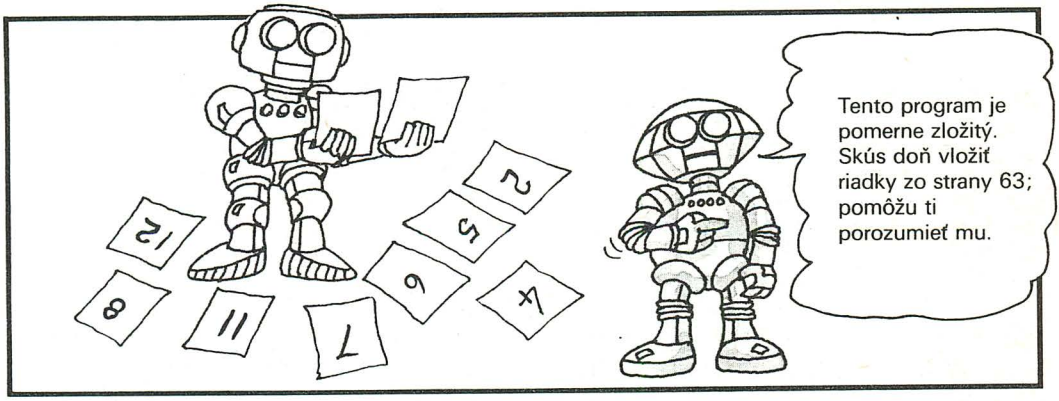
Ak príkaz GOTO z riadka 260 dávaš do nového riadka, použi príkaz IF/THEN.

*Pre počítače Sinclair zmeň riadok 120 na DIM N$(T,N), kde N je počet znakov najdlhšieho triedeného slova.

Shellovo triedenie

Bublinové triedenie je pre väčšie množstvo údajov veľmi pomalé. Program Shellovo triedenie čísel je asi trikrát rýchlejší ako program Bublinové triedenie.

Pri Shellovom triedení počítač neporovnáva prvky vedľa seba, ale prvky vzdialené od seba o hodnotu C (krok). Na začiatku sa C zvolí rovné polovici počtu triedených čísel. Po prejdení celej postupnosti s krokom C sa pokračuje s menším krokom. Program sa končí, keď sa utriedi postupnosť s krokom C = 1.



Program

```
100 REM SHELLOVO TRIEDENIE ČÍSEL

110 INPUT "KOĽKO ČÍSEL CHCEŠ
    UTRIEDIŤ? ";T
120 DIM N(T)
130 FOR I=1 TO T
140 PRINT "ČÍSLO ";I
150 INPUT N(I)
160 NEXT I

170 LET C=T

180 LET C=INT(C/2)

190 IF C = 0 THEN GOTO 330

200 LET D=T-C
```

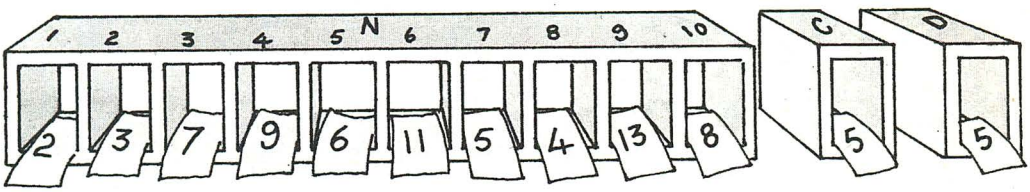
Tieto riadky majú taký istý význam ako v programe Bublinové triedenie čísel. T je počet triedených čísel. V riadkoch 130 – 160 si počítač pýta čísla, ktoré bude triediť, a uloží ich do poľa.

Premennej C priradí počet triedených čísel.

Vydelením čísla C dvoma zistí počet členov v prvej časti postupnosti. INT znamená, že sa zanedbajú číslice za desatinnou čiarkou (C bude celé číslo).

Počítač postupne delí C dvoma. Ak C = 0, pokračuje riadkom 330 a napíše utriedenú postupnosť.

D je počet členov v druhej časti postupnosti.




```
210 LET E=1

220 LET F=E
230 LET G=F+C
```

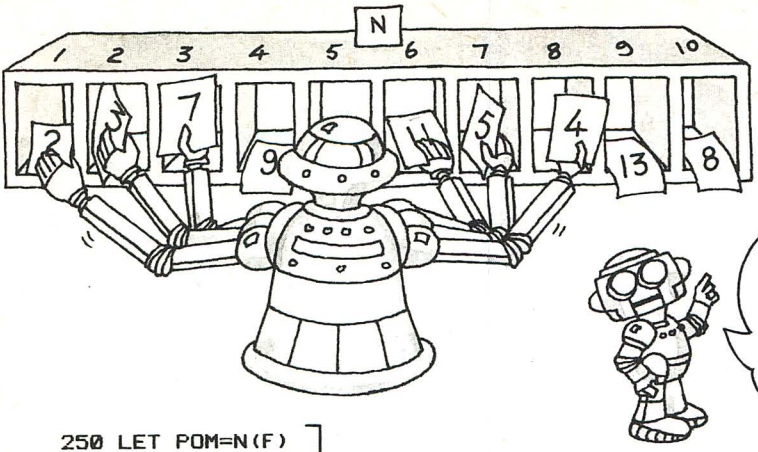
E je počítadlo.

Premenné F a G sa používajú ako indexy poľa N, v ktorom sú uložené triedené čísla.

PROGRAM POKRAČUJE
NA NASLEDUJÚCEJ STRANE.

```
240 IF N(F) <= N(G) THEN GOTO 300
```

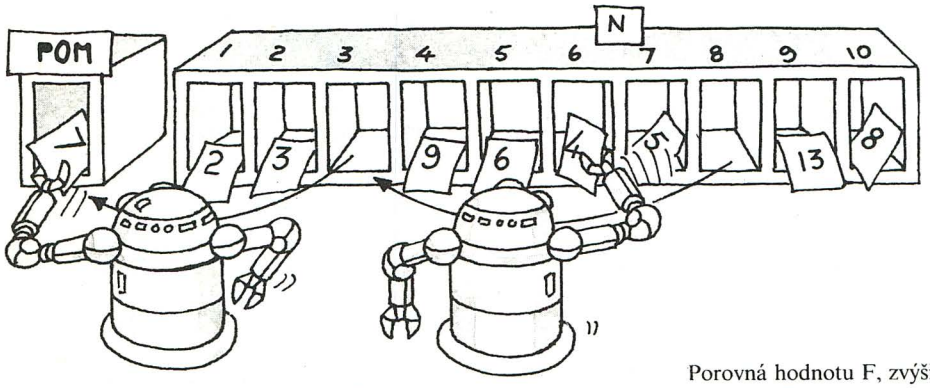
Ak N(F) je menšie ako N(G), počítač pokračuje riadkom 300, zvýši hodnotu E o 1 a priradí premenným F a G hodnoty indexov nasledujúcej dvojice členov.



Tu treba utriediť 10 čísel. Pri prvom opakovaní je C=5, takže počítač porovná čísla s indexmi 1 až 5 s číslami s indexmi 6 až 10.

```
250 LET POM=N(F)
260 LET N(F)=N(G)
270 LET N(G)=POM
```

Ak N(F) je väčšie ako N(G), vymení ich.




```
280 LET F=F-C
290 IF F > 0 THEN GOTO 230
300 LET E=E+1
```

Porovná hodnotu F, zvýši počítadlo E o 1, aby sa zmenili hodnoty indexov F a G v riadkoch 220 a 230.

```
310 IF E > D THEN GOTO 180
```

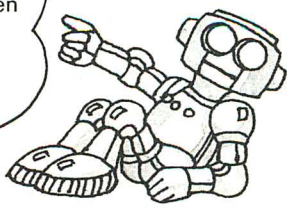
Zistí, či E je väčšie ako D. Ak áno, pokračuje riadkom 180 a znovu delí postupnosť.

```
320 GOTO 220
330 PRINT "UTRIEDENÁ POSTUPNOSŤ"
340 FOR I=1 TO T
350 PRINT N(I)
360 NEXT I
```

Napíše utriedenú postupnosť.

Skús do programu doplniť riadky tak, aby počítač spolu s utriedenou postupnosťou napísal počet porovnaní a počet výmen (riešenie je na strane 143). Zadaj počítaču rôzne postupnosti a sleduj, ako sa zmenia sledované počty.

Program uprav tak, aby triedil slová — premennú N zmeň na N$ (v riadkoch 120, 150, 240–270, 350), POM zmeň na POM$ (v riadkoch 250 a 270) a zmeň príkazy PRINT. Ak pracuješ na Sinclairi, zmeň riadok 120 na DIM N$(T,N), kde N je počet znakov najdlhšieho zo zadávaných reťazcov.



Chceš vedieť niečo viac o programe?

Lepšiu predstavu o práci programu Shellovo triedenie získaš, ak doň vložíš nasledujúce riadky. Vypíšu hodnoty premenných, aby si videl, ktoré čísla počítač porovnáva.

```
233 PRINT
235 PRINT "F=";F;",G=";G
237 PRINT "POROVNÁVAM N(";F;") A N(";G;")"
245 PRINT "VYMIEŇAM N(";F;") A N(";G;")"
274 PRINT "POSTUPNOSŤ=";
275 FOR J=1 TO T
276 PRINT N(J);" ";
277 NEXT J
278 PRINT
279 INPUT "STLAČ ĽUBOVOĽNÝ KLÁVES";Z$
```

F a G sú indexy porovnávaných členov.

Indexy členov, ktoré sa zamieňajú.

Počítač napíše, ako postupnosť práve vyzerá.

Porovnanie triedení

Ak si skúšal oba spôsoby triedenia na malom množstve údajov, možno si ani nespozoroval, o koľko je Shellovo triedenie rýchlejšie. Nechaj obidva programy vygenerovať náhodnú postupnosť čísel a pozoruj, ako dlho ju budú triediť. Čím dlhšia je postupnosť čísel, tým väčšmi bude narastať rozdiel medzi oboma triedeniami. Programy na ďalšej strane nakreslia grafy, ktoré znázorňujú rozdiely medzi týmito spôsobmi triedenia.

Generovanie čísel

```
140 LET N(I)=INT(RND(1)*200+1)
150 PRINT N(I)
165 INPUT "ZOBER SI STOPKY A KEĎ
    BUDEŠ PRIPRAVENÝ, STLAČ
    RIADKOVAČ";Z$
```

Toto číslo môžeš zmeniť, na aké chceš.

Aby programy generovali náhodné postupnosti čísel, musíš v oboch nahradiť riadky 140 a 150 a vložiť riadok 165, aby si mohol riadiť spustenie triedenia. V riadku 140 sa generujú náhodné čísla od 1 do 200 a ukladajú sa do poľa N. V riadku 150 sa čísla napíšu a riadok 165 spôsobí, že program bude pokračovať až po stlačení riadkovača.

Testovanie

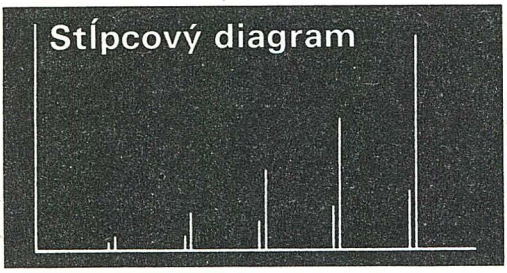
Každý program by si mal spustiť niekoľkokrát, aby si videl, za aký čas utriedi počítač rôzny počet čísel. Najprv vezmi povedzme 10 čísel, potom 20, 30, 40 atď. Rôzne typy počítačov budú triediť čísla rôzne rýchlo. Niektoré (napr. ZX81) majú pomalý a rýchly režim. V tabuľke vidíš časy, ktoré dosiahol počítač Apple II.

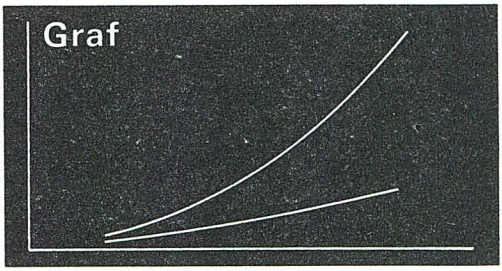
Testovanie triedení					
Počet triedených čísel	10	20	30	40	50
Bublinové triedenie	2 s	5 s	11 s	18 s	29 s
Shellovo triedenie	1 s	2 s	4 s	6 s	8 s

Kreslenie grafov

Výsledky výpočtov sú zrozumiteľnejšie a ľahšie čitateľné, ak ich graficky znázorníš a doplníš slovným vysvetlením. Pozri si nasledujúci program, ktorý nakreslí stĺpcový diagram, znázorňujúci rozdiely medzi bublinovým a Shellovým triedením. Na ďalšej strane ti poradíme, ako program zmeniť, aby sa namiesto stĺpcového diagramu kreslil graf.

Program je pomerne jednoduchý, ľahko ho môžeš zmeniť tak, aby spracúval iné druhy informácií. A dal by sa rozšíriť aj o príkazy na kreslenie rôznymi farbami.





Takto vyzerajú obrázky nakreslené dvoma grafickými programami. Na obidvoch sa porovnáva čas potrebný na utriedenie postupnosti 10, 20, 30, 40 a 50 čísel. Čas je znázornený na osi y, teda zvislej osi, počet triedených čísel na osi x. Ak tvoj počítač vie napísať text na určené miesto, doplň grafy pre obe triedenia vysvetlivkami. Ako programy upraviť pre PMD-85, je na nasledujúcej strane.

Program Stĺpcový diagram

```
100 DIM B(5): DIM S(5)
110 LET N=0
120 FOR I=10 TO 50 STEP 10
130 LET N=N+1
140 PRINT "KOĽKO SEKÚND TRVÁ
    TRIEDENIE ";I;" ČÍSEL:"
150 INPUT "BUBLINOVÝM
    TRIEDENÍM ?";B(N)
160 INPUT "SHELLOVÝM
    TRIEDENÍM ?";S(N)
170 NEXT I
180 INPUT "AKÁ JE VÝŠKA OBRAZOVKY? ";V
190 INPUT "A ŠÍRKA? ";S
200 REM DOPLŇ POTREBNÉ GRAFICKÉ PRÍKAZY
```

V poliach S a B sú zaznamenané údaje o bublinovom a Shellovom triedení.

Počítač v cykle načíta vstupné údaje do polí. N je počítadlo, ktoré používa ako index polí.

Všimni si, ako sa premenná cyklu I využíva aj pri výpise.

Čísla v príkazoch PLOT a DRAW udávajú vzdialenosť od okraja obrazovky. Ak na tvojom počítači musia čísla udávať vzdialenosť od posledného kresleného bodu, zmeň program podľa návodu na strane 149.

```
210 REM KRESLENIE OSÍ
220 PLOT 1,V: DRAW 1,1: DRAW S,1
230 REM KRESLENIE GRAFU
240 LET X=(S*0.75)/5
250 LET Y=(V*0.75)/B(5)
```

Nakreslí súradnicové osi s počiatkom v bode 1,1. V a S sú výška a šírka obrazovky.

Čísla X a Y sú mierky na x-ovej a y-ovej osi. X je rovné trom štvrtinám šírky delenej piatimi (päť je počet testov každého z triedení). Y je rovné trom štvrtinám výšky delenej najväčším časom, teda B(5).

PROGRAM POKRAČUJE NA NASLEDUJÚCEJ STRANE.

```
260 FOR I=1 TO 5
270 PLOT INT(I*X),I
280 DRAW INT(I*X),INT(B(I)*Y)
290 PLOT INT(I*X-4),1
300 DRAW INT(I*X-4),INT(S(I)*Y)
310 NEXT I
```

Vieš zmeniť program tak, aby kreslil hrubšie stĺpce? Riešenie je na strane 143.

Cyklus, v ktorom sa kreslia stĺpce. Počítač pri každom opakovaní nakreslí stĺpec pre obe triedenia.

Nakreslí bod so súradnicami I*X,1. Pri prvom prechode cyklom je I rovné 1 a X jednotke na osi X.

Nakreslí čiaru do bodu B(I)*Y. Pri prvom opakovaní cyklu je I = 1, B(I) je teda dĺžka bublinového triedenia desiatich čísel, Y je mierka na osi Y.

Ak zameníš číslo −4 za iné, zmení sa vzdialenosť medzi stĺpcami.

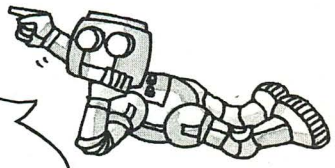
Nakreslí bod I*X−4, teda 4 body vľavo od stĺpca pre bublinové triedenie.

Nakreslí čiaru do bodu S(I)*Y.

Vráti sa na začiatok cyklu a kreslí ďalšiu dvojicu stĺpcov.

Ak tvoj počítač vie napísať text na určené miesto, pridaj riadky, ktoré napíšu názov grafu. Pre počítač PMD-85 by vyzerali takto:

```
305 MOVE I*X,INT(B(I)*Y)+50:LABEL 1,1;"B"
308 MOVE I*X-70,INT(S(I)*Y)+50:LABEL 1,1;"S"
```



Program Graf

Musíš nakresliť prvý bod grafu a spojiť ho s ďalšími bodmi. Grafy pre obe triedenia sa preto kreslia vo zvláštnych cykloch. Aby program namiesto stĺpcového diagramu kreslil graf, nahraď všetky riadky od 270.

```
270 PLOT INT(X),INT(B(1)*Y)
280 FOR N=2 TO 5
290 DRAW INT (N*X),INT(B(N)*Y)
300 NEXT N
310 PLOT INT(X),INT(S(1)*Y)
320 FOR N=2 TO 5
330 DRAW INT(N*X),INT(S(N)*Y)
340 NEXT N
```

Počítač nakreslí prvý bod krivky bublinového triedenia, ktorý má súradnice X,B(1)*Y.

Cyklus, v ktorom kreslí graf bublinového triedenia. Pri každom opakovaní cyklu zvýši hodnotu N o 1 a nakreslí ďalšiu časť čiary.

Nakreslí prvý bod krivky Shellovho triedenia, ktorý má súradnice X,S(1)*Y.

Cyklus, v ktorom kreslí graf Shellovho triedenia.

Konverzačný program

Na niekoľkých nasledujúcich stranách je program, vďaka ktorému sa môžeš s počítačom rozprávať. Samozrejme, počítač je iba taký rozumný, aký rozumný je program, ktorý ho riadi. Všetky slová, ktorými počítač zareaguje na tvoje slová, sú uložené v reťazcových poliach.*

Hlavnou úlohou programu je vybrať vhodnú odpoveď na tvoje slová. Skladá sa z niekoľkých podprogramov, ktoré spracúvajú reťazce. Úspech programu závisí nielen od jeho výstavby, ale aj od slov a viet, ktoré sú v ňom uložené. Slovník počítača môžeš meniť — napríklad tak, aby sa s tebou rozprával o inom, alebo aby odpovede boli vtipnejšie.

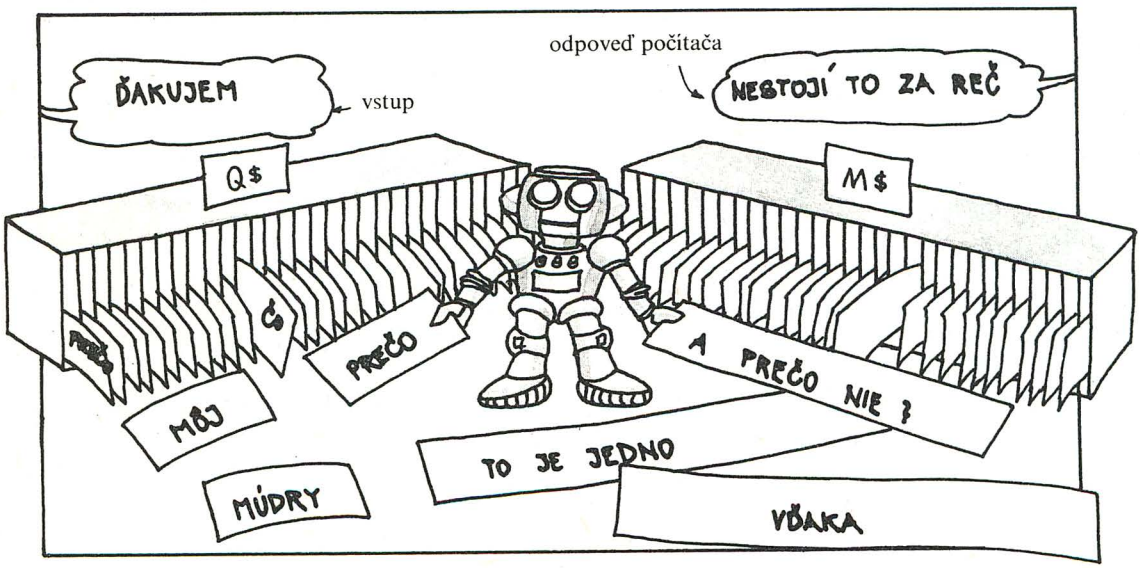
Krátke príklady

```
AKO SA VOLÁŠ? KATKA
POROZPRÁVAJ SA SO MNOU, KATKA
?DOBRE
KATKA, HÁDAJ, ČO SI MYSLÍM!
?VZDÁVAM SA, NEVIEM
NEVYJADRUJ SA ZÁPORNE!
?MYSLÍŠ NA PRÁZDNINY?
TO JE TAJNÉ
?MÁŠ CHUŤ NA ZMRZLINU?
TO HÁDAM NIE JE MOŽNÉ, TY SI
ZÁZRAČNÉ DIEŤA...
?ĎAKUJEM
NESTOJÍ TO ZA REČ
```

```
?A O ČOM SA CHCEŠ SO MNOU
ROZPRÁVAŤ?
ROZPRÁVAJME SA RADŠEJ
O SKLADACÍCH DAŽĎNIKOCH
?ALE JA NIJAKÝ NEMÁM
ÁÁÁCH
?MÔŽEM SA ŤA NIEČO OPÝTAŤ?
TO JE DOBRÁ OTÁZKA
?KTO SI?
SOM LEN POČÍTAČ
?ALE SI CELKOM MÚDRY
VĎAKA
```

Ako program pracuje

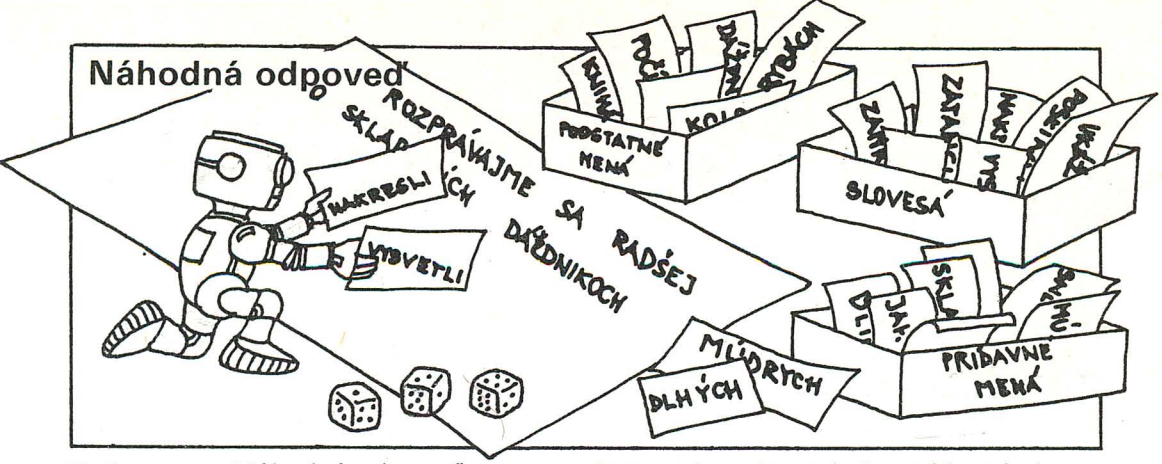
Počítač môže odpovedať dvoma spôsobmi. Buď odpovedá podľa toho, čo si napísal, alebo odpoveď vytvára náhodne.



Podprogram Rozumná odpoveď má v poli Q$ uložený zoznam najčastejšie používaných slov. Vhodná odpoveď na každé z nich je uložená v poli M$.

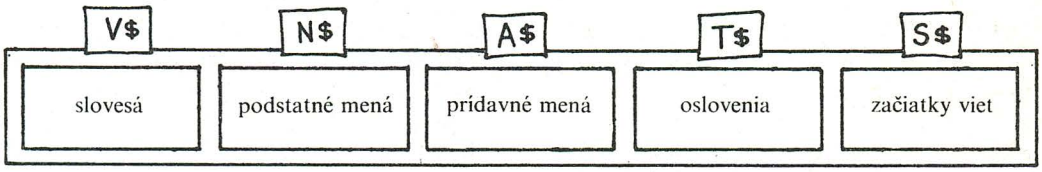
Keď počítaču niečo napíšeš, skontroluje, či si použil niektoré zo slov v Q$. Ak áno, napíše zodpovedajúcu odpoveď z poľa M$.

* Návod, ako program upraviť pre počítače Sinclair, nájdeš na strane 149.

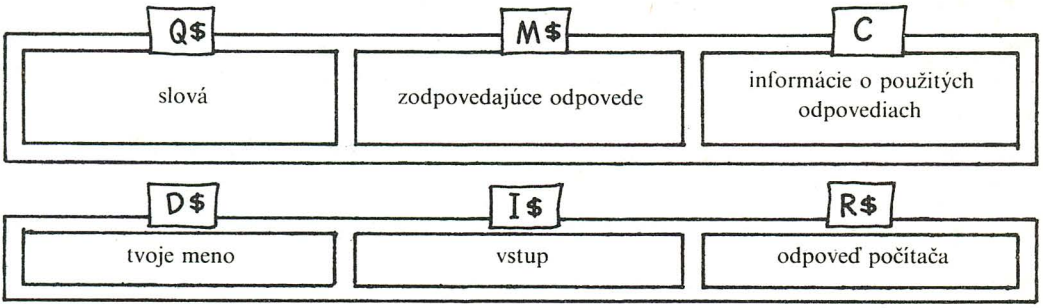


Podprogram Náhodná odpoveď obsahuje nedokončené vety, ktoré počítač doplní náhodne vybranými slovesami, podstatnými a prídavnými menami. Všetky slová sú uložené v poliach a vyberajú sa tak, aby výsledná veta dávala zmysel.

Aké premenné používame



V týchto poliach sú uložené slová používané pri náhodnom vytváraní odpovedí.



Program

```
100 GCLEAR
110 DIM V$(10),N$(10),A$(10)
120 DIM T$(10),S$(10)
130 DIM M$(30),Q$(30),C(30)
140 REM NAČÍTANIE ÚDAJOV
150 GOSUB 1000
200 REM HRÁČOVA ODPOVEĎ
210 INPUT "AKO SA VOLÁŠ? ";D$
220 PRINT
230 PRINT "POROZPRÁVAJ SA
    SO MNOU, ";D$
240 INPUT I$
250 IF I$ = "" THEN GOTO 220
260 IF I$ = "TAK AHOJ" THEN GOTO 900
```

Deklarácia polí. Na niektorých počítačoch polia, ktoré majú menej ako 10 prvkov, nemusíš deklarovať.

Počítač pokračuje podprogramom, v ktorom načíta údaje.

Tvoju odpoveď uloží do premennej I$.

Ak si namiesto odpovede iba stlačil riadkovač, pokračuje riadkom 220.

Ak napíšeš TAK AHOJ, pokračuje riadkom 900 a spýta sa ťa, či chceš skutočne skončiť.

PROGRAM POKRAČUJE
NA NASLEDUJÚCEJ STRANE.

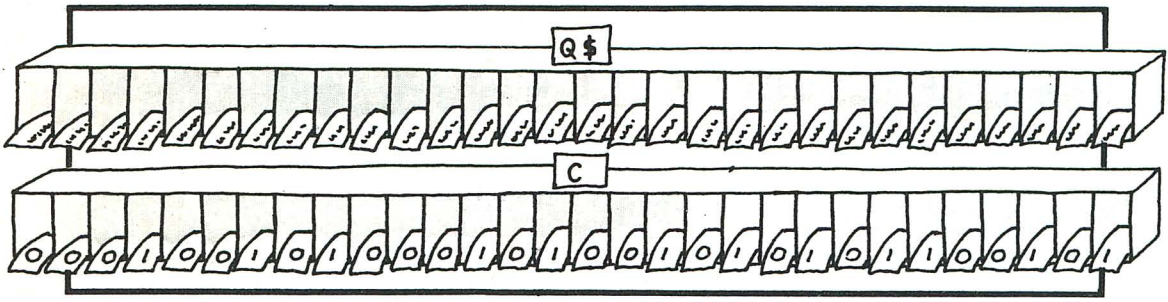
```
300 REM POČÍTAČ ODPOVEDÁ

310 LET ODPOVED=INT(RND(1)*8+1)
320 IF ODPOVED < 6 THEN GOTO 490

330 GOTO 600

340 PRINT

350 PRINT R$
360 PRINT

400 REM KOCKO ODPOVEDÍ SA POUŽILO
```

Podľa hodnoty náhodne vybraného čísla v premennej ODPOVED sa počítač rozhodne, ako bude odpovedať. Ak hodnota premennej ODPOVED je menšia ako 6, pokračuje podprogramom Rozumná odpoveď, ktorý sa začína v riadku 490.

Ak hodnota premennej ODPOVED je väčšia alebo rovná 6, odpoveď vytvorí náhodne.

Vybranú odpoveď uloží do premennej R$ a v riadku 350 ju napíše na obrazovku.

PROGRAM POKRAČUJE DOLU.



V riadkoch 400—470 počítač kontroluje, koľko odpovedí z M$ už použil. Zakaždým, keď v tvojej odpovedi nájde niektoré zo slov z Q$, hodnotu zodpovedajúceho prvku v poli C zvýši o 1. To mu zabráni vybrať neskôr tú istú odpoveď druhý raz. Ak zistí, že už použil viac než 12 odpovedí, priradí všetkým prvkom v poli C hodnotu 0.

```
405 LET T=0
410 FOR K=1 TO 30
420 LET T=T+C(K)
430 NEXT K

440 IF T < 12 THEN GOTO 460

450 FOR K=1 TO 30: LET C(K)=0: NEXT K

460 LET T=0

470 GOTO 240

490 REM ROZUMNÁ ODPOVEĎ
500 FOR SLOVO=1 TO 30

510 LET L1=LEN(Q$(SLOVO))

520 LET L2=LEN(I$)

530 FOR TEST=1 TO 12

540 IF MID$(I$,TEST,L1) = Q$(SLOVO)
    THEN GOTO 560

550 NEXT TEST: NEXT
    SLOVO: GOTO 600
```

Cyklus, v ktorom počítač zisťuje počet nenulových prvkov v poli C(T).

Ak T je menšie než 12, použil menej než 12 odpovedí a nemusí meniť prvky poľa C.

V cykle nastaví prvky poľa C na 0.

Nastaví premennú T na 0.

Pokračuje riadkom 240 a čaká na tvoju odpoveď.

Cyklus sa zopakuje toľkokrát, koľko je slov v poli Q$.

Pri každom opakovaní cyklu počítač uloží do L1 počet znakov porovnávaného slova.

Do L2 uloží dĺžku tvojej odpovede.

Vnorený cyklus opakuje L2-krát. Pri každom opakovaní porovnáva znaky reťazca I$ so znakmi jedného slova z Q$. Ak sú rovnaké, pokračuje riadkom 560.

Ak sa v I$ nenachádza žiadne zo slov z Q$, pokračuje riadkom 600 a odpoveď vytvára náhodne.

```
560 IF C(SLOVO) > 0 THEN GOTO 550
570 LET C(SLOVO) = C(SLOVO)+1
580 LET R$ = M$(SLOVO)
590 GOTO 340
```

Ak počítač v I$ nájde niektoré zo slov, pokračuje riadkom 560. Skontroluje, či zodpovedajúcu odpoveď už použil, teda či C sa nerovná 0. Ak áno, hľadá v I$ ďalšie slovo. Ak premenná C je rovná 0, zmení ju na 1. V riadku 580 potom priradí odpoveď z M$ do R$, aby ju v riadku 350 mohol napísať.

PROGRAM POKRAČUJE DOLU.

Pri prvom opakovaní vonkajšieho cyklu je SLOVO = 1 a počítač vyberie prvé slovo z Q$. Pri prvom opakovaní vnoreného cyklu je TEST = 1 a počítač vybrané slovo porovnáva s prvými piatimi znakmi vstupu (5 je počet znakov vybraného slova). Ak sa nerovnajú, opakuje vnorený cyklus: TEST = 2 a s vybraným slovom z Q$ porovnáva 2. až 6. znak reťazca I$. Ak sa vybrané slovo v I$ nenachádza, zopakuje vonkajší cyklus, pričom v I$ hľadá ďalšie slovo z Q$.

```
600 REM NÁHODNÁ ODPOVEĎ
610 LET E=INT(RND(1)*10+1)
620 LET F=INT(RND(1)*10+1)
630 LET G=INT(RND(1)*10+1)
640 LET H=INT(RND(1)*10+1)
650 LET K=INT(RND(1)*10+1)
660 LET L=INT(RND(1)*10+1)
```

Podľa náhodne zvolených čísel počítač vyberie vetu a doplní ju. Číslo E určuje, ktorú vetu vyberie. F určuje sloveso, G prídavné meno, H podstatné meno, K začiatok vety a L oslovenie.

```
670 ON E GOTO 700,720,740,760,
    780,800,820,840,860,880
```

Ďalej pokračuje podľa hodnoty premennej E. Ak E = 1, pokračuje riadkom 700, ak E = 2, pokračuje riadkom 720 atď. Význam slova ON je vysvetlený na strane 48.

```
700 LET R$=D$+", TY "+T$(L)+
    ", ČO SI TO SPRAVIL!"
710 GOTO 340
720 LET R$="ROZPRÁVAJME SA
    RADŠEJ O "+A$(G)+" "+N$(H)+"."
730 GOTO 340
740 LET R$=D$+", "+V$(F)+
    " MI NIEČO PEKNÉ!"
750 GOTO 340
760 LET R$=S$(K)+", ČO SI
    TO NAPÍSAL, TY "+T$(L)+"!"
770 GOTO 340
```

V riadkoch 700 až 900 sú napísané nedokončené vety, ktoré počítač doplní slovami z N$, V$, A$, T$ a S$. Znamienko + označuje spojenie reťazcov. Pri písaní reťazcov v úvodzovkách nesmieš vynechať nijakú medzeru, ináč by boli slová nesprávne oddelené. Úplnú odpoveď počítač uloží do R$ a v riadku 350 ju napíše.

PROGRAM POKRAČUJE NA NASLEDUJÚCEJ STRANE.

```
780  LET R$=D$+", HÁDAJ, ČO SI MYSLÍM!"
790  GOTO 340
800  LET R$="CÍTIM SA AKO "+T$(L)+", A TY?"
810  GOTO 340
820  LET R$=S$(K)+", TY SI "+T$(L)+"..."
830  GOTO 340
840  LET R$="TO ISTÉ SI MYSLÍM AJ JA O "+
     A$(G)+" "+N$(H)+"."
850  GOTO 340
860  LET R$="VIEŠ, ŽE VYNAŠLI "+
     LEFT$(A$(G),LEN(A$(G))-2)+" POČÍTAČ?"
870  GOTO 340
880  LET R$=S$(K)+", RADŠEJ MI NIEČO "+
     V$(F)+"."
890  GOTO 340
900  REM KONIEC

910  PRINT "TY SA LÚČIŠ?"

920  PRINT "UŽ SA SO MNOU NECHCEŠ ROZPRÁVAŤ?"
930  INPUT Z$: IF Z$ = "CHCEM" GOTO 210
940  PRINT "TAK TEDA AHOJ!"
950  END
```

Ak si odpovedal TAK AHOJ, počítač pokračuje týmto príkazom.

```
1000 REM ÚDAJE PRE PODPROGRAM ROZUMNÁ ODPOVEĎ
1010 FOR I=1 TO 30: READ Q$(I): NEXT I
1020 DATA "PREČO","ČO ","KTO SI","?","ZNAMENÁ"
1030 DATA "TVOJ"," MI","JA ","TO ","POVEDZ"
1040 DATA "NIE ","MÔJ","?"," SI","?"
1050 DATA "ÁNO","TY ","MÚDRY","MYSLÍŠ","?"
1060 DATA "VIEŠ","NEVYCHOVANÝ","ĎAKUJEM","AKO","TY "
1070 DATA " NE"," JE "," K ","?","POZNÁŠ"
```

Slová, ktoré počítač hľadá v tvojej odpovedi. Odpíš ich pozorne, medzery vnútri úvodzoviek sú súčasťou reťazcov.

```
1100 REM ODPOVEDE K SLOVÁM Z Q$
1110 FOR I=1 TO 30 : READ M$(I): NEXT I
1120 DATA "A PREČO NIE?","TO JE JEDNO","SOM LEN POČÍTAČ"
1130 DATA "TO JE DOBRÁ OTÁZKA!","NEVIEM","AKO TO MYSLÍŠ?"
1140 DATA "A KTO SI TY?","ÁÁÁCH","ČO ZNAMENÁ TO?"
1150 DATA "NEPOVIEM","NEVYJADRUJ SA ZÁPORNE!","ACH JAJ!"
1160 DATA "KEĎ TO TY HOVORÍŠ...","ČO TÝM MYSLÍŠ?",
     "PREČO?"
1170 DATA "TAKŽE SÚHLASÍŠ?","VARI MA NEMÁŠ RÁD?","VĎAKA"
1180 DATA "ROZHODNI SA SÁM!","AKÁ HLÚPA OTÁZKA!"
1190 DATA "MÁŠ NÍZKE IQ!","A TO SI EŠTE NIČ NEVIDEL!"
1200 DATA "NESTOJÍ TO ZA REČ","JEDNODUCHO","AJ TY!"
1210 DATA "HLÚPOSŤ!","ODKIAĽ TÁ ISTOTA?","CHOĎ PREČ!"
1220 DATA "TO JE TAJNÉ","VEDOMOSTI NIE SÚ MOJOU
     SILNOU STRÁNKOU"
1300 REM PODSTATNÉ MENÁ
1310 FOR I=1 TO 10: READ N$(I): NEXT I
```

Odpovede počítača, ak nájde niektoré zo slov z Q$. Sú v takom istom poradí ako slová. Napríklad ak nájde piate slovo z Q$, napíše piate slovo z M$.

```
1320 DATA "KNIHÁCH","POČÍTAČOCH"
1330 DATA "DÁŽDNIKOCH","RYBÁCH"
1340 DATA "KOLOTOČOCH","TENISKÁCH"
1350 DATA "LIETADLÁCH","ŠKATULIACH"
1360 DATA "DOMOCH","SKRINIACH"
```

Podstatné mená, ktoré dopĺňa do viet.

```
1400 REM SLOVESÁ
1410 FOR I=1 TO 10: READ V$(I): NEXT I
1420 DATA "ZAHRAJ","ZATANCUJ"
1430 DATA "PREZRAĎ","NAPÍŠ"
1440 DATA "NAKRESLI","VYSVETLI"
1450 DATA "POŠEPKAJ","UKÁŽ"
1460 DATA "VYPOČÍTAJ","NAPROGRAMUJ"
```

Slovesá, ktoré dopĺňa do viet.

Slová môžeš zmeniť, ale dávaj pozor, aby vytvárané vety dávali zmysel.

```
1500 REM PRÍDAVNÉ MENÁ
1510 FOR I=1 TO 10: READ A$(I): NEXT I
1520 DATA "MÚDRYCH","PÁSIKOVANÝCH"
1530 DATA "DLHÝCH","PAPIEROVÝCH"
1540 DATA "JARNÝCH","SKLADACÍCH"
1550 DATA "SNEHOVÝCH","ROZPRÁVKOVÝCH"
1560 DATA "HLADNÝCH","MOKRÝCH"
```

Prídavné mená.

```
1600 REM ZAČIATKY VIET
1610 FOR I=1 TO 10: READ S$(I): NEXT I
```

Údaj nesmie obsahovať čiarku, pretože pre počítač by to bol signál, že jeden údaj sa končí a začína sa druhý.

```
1620 DATA "HM","NO TEDA","ACH JAJ"
1630 DATA "DOČERTA","MILÝ PRIATEĽ"
1640 DATA "ACH NIE","VARI LEN NIE"
1650 DATA "TO HÁDAM NIE JE MOŽNÉ"
1660 DATA "POČÚVAJ","ACH JAJ"
1700 REM OSLOVENIA
1710 FOR I=1 TO 10: READ T$(I): NEXT I
1720 DATA "HVIEZDA","ZÁZRAČNÉ DIEŤA"
1730 DATA "GÉNIUS","PRÍŠERA"
1740 DATA "POČÍTAČOVÝ BLÁZON","LOTOR"
1750 DATA "ŠPEKULANT","MUDRLANT"
1760 DATA "STRAŠIDLO","KÚZELNÍK
1770 RETURN
```

Oslovenia a začiatky viet.

Ako zmeniť program

1. Najjednoduchšie je zmeniť slová a vety. Dávaj pozor, aby nové vety mali zmysel. Podstatné mená sú v množnom čísle, a tak ak ich napíšeš v jednotnom čísle, musíš podobne zmeniť aj prídavné mená. Do programu môžeš slová aj pridať — musíš však zväčšiť polia, zmeniť čísla, ktoré určujú počet opakovaní cyklov, a čísla v riadkoch 610 – 660.
2. Skús zmeniť aj slová z Q$. Ku každému novému slovu vymysli vhodnú odpoveď a ulož ju na správne miesto do M$.
3. Ak chceš, aby počítač častejšie vytváral odpovede náhodne, zmenši číslo 6 v riadku 320. Môžeš zmeniť aj číslo 12 v riadku 440, ktoré určuje, ako často sa priraďuje všetkým prvkom poľa C hodnota 0.

Rozšírenie programu

Ak k programu pridáš nasledujúce riadky, počítač sa bude rozprávať aj sám so sebou.

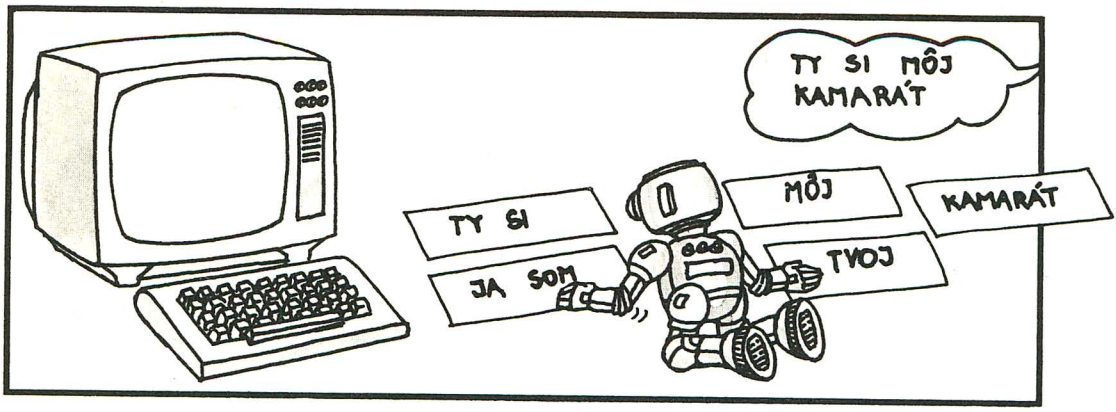
```
160 PRINT "POROZPRÁVAŠ SA SO MNOU?
    (ODPOVEDZ A ALEBO N)"
170 INPUT K$
180 IF K$ = "N" THEN LET D$=
    "ROM": GOTO 600
470 IF K$ = "A" THEN GOTO 240
475 IF K$ = "N" THEN LET I$=R$
480 LET R$="": GOTO 310
```

V D$ je uložené meno hráča. Ak sa počítač rozpráva sám so sebou, do D$ uloží svoje meno (ROM). Výpočet pokračuje riadkom 600 a odpoveď sa vytvára náhodne.

Odpoveď počítača R$ sa stane novým vstupom I$. Výpočet pokračuje riadkom 310, kde sa vyberie spôsob odpovedania.

Podprogram Opakovanie odpovede

Na týchto dvoch stranách nájdeš ďalší podprogram, ktorý môžeš pripojiť ku Konverzačnému programu. Rozšírený program umožní počítaču odpovedať ti tvojimi vlastnými slovami. Návod, ako program upraviť pre počítače Sinclair, nájdeš na nasledujúcej strane.



Podprogram Opakovanie odpovede pracuje podobne ako podprogram Rozumná odpoveď. Používajú sa v ňom polia U$ a W$. V U$ sú slová, ktoré bude počítač hľadať v tvojej vete, a vo W$ sú odpovede počítača. Ak použiješ niektoré zo slov z U$, počítač ho nahradí príslušným slovom z W$. Zvyšok vety napíše bezo zmeny.

```
135  DIM U$(9),W$(9)
138  DIM Z$(5)
```
Deklarácia polí U$, W$ a poľa Z$, v ktorom si počítač zaznamenáva nahradené slová.

```
155  GOSUB 2200
```
Podprogram, v ktorom načíta údaje.

```
325  IF ODPOVED = 7 THEN GOTO 2000
```
Ak ODPOVED = 7, počítač vykoná podprogram Opakovanie odpovede.

```
2000 REM OPAKOVANIE ODPOVEDE
```

```
2010 LET Z=0
```
Z je počítadlo.

```
2020 LET P=LEN(I$)
```
Do P uloží dĺžku vstupu.

```
2030 FOR A=1 TO P
```
Cyklus sa zopakuje P-krát.

```
2040 FOR B=1 TO 9
```
Cyklus sa zopakuje toľkokrát, koľko je položiek v U$.

```
2050 LET L=LEN(U$(B))
```
Pri každom opakovaní vnoreného cyklu počítač vezme nasledujúce slovo z U$.

```
2060 IF MID$(I$,A,L) = U$(B) THEN
     GOTO 2140
```
Porovná znaky A až L s B-tým prvkom poľa U$. Ak sa rovnajú, pokračuje riadkom 2140.

```
2070 NEXT B: NEXT A
```
Pri každom opakovaní cyklu B počítač porovnáva ďalší prvok z U$. Ak opakuje cyklus A, berie do úvahy len neporovnávanú časť vstupu.

```
2080 IF Z$(1) = "" THEN GOTO 600
```
Ak počítač vo vstupe nenašiel ani jedno slovo z U$, odpoveď vytvorí náhodne.

```
2090 FOR J=1 TO Z
2100 PRINT Z$(J);
2110 LET Z$(J)=""
2120 NEXT J
```

Napíše všetky slová zo Z$.

```
2130 LET R$=I$: GOTO 350
```

Zvyšok vstupu uloží do R$ a v riadku 350 ho napíše.

```
2140 LET Z=Z+1
```

V Z si zaznamená počet slov v Z$.

```
2150 IF A > 1 THEN LET Z$(Z)=LEFT$(I$,A-1)
     +W$(B)+" "
```

A je premenná z cyklu, ktorý sa začína v riadku 2030. Jej hodnota je pozíciou prvého znaku slova v I$, ktoré počítač nahrádza. Ak A > 1, do Z$ uloží začiatok reťazca I$ a nové slovo.

```
2160 IF A < 2 THEN LET Z$(Z)=W$(B)+" "
```

Ak A < 2, nahrádzané slovo je hneď na začiatku, a do Z$ uloží len nové slovo.

```
2170 LET I$=MID$(I$,A+L,P)
```

Zvyšok vstupu uloží do I$ a v cykle v ňom hľadá ďalšie slovo, ktoré sa dá nahradiť.

```
2180 GOTO 2020
2200 REM ÚDAJE PRE PODPROGRAM OPAKOVANIE ODPOVEDE

2210 FOR I=1 TO 9
2220 READ U$(I),W$(I)
2230 NEXT I

2240 DATA "JA SOM ","TY SI ","TY SI ","JA SOM "
2250 DATA "JA ","TY ","TY ","JA "
2260 DATA "MÔJ ","TVOJ ","TVOJE ","MOJE "
2270 DATA "TVOJ ","MÔJ ","MNE ","TEBE "
2280 DATA "TY ","POČÍTAČE "
2290 RETURN
```

Údaje uloží do U$ a W$.

Podprogram Opakovanie odpovede na počítačoch Sinclair

Ak chceš program spustiť na počítači Spectrum alebo ZX81, pridaj k nemu nasledujúce riadky. Na načítanie údajov na počítači ZX81 použi metódu zo strany 149. Cykly na načítanie údajov na ZX81 vlož medzi riadky 1000 až 1720 a príkazy DIM pred riadok 1000.

```
135  DIM U$(9,7)
136  DIM W$(9,8)
137  DIM Z$(5,20)
2042 LET P$=""
2044 FOR I=1 TO LEN (U$(B))
2046 IF U$(B)(I TO I) <> "*" THEN LET P$=P$+U$(B)(I TO I)
2048 NEXT I
2050 LET L=LEN(P$)
2055 IF L+A-1 > P THEN GOTO 2070
2060 IF I$(A TO A+L-1) = P$ THEN GOTO 2140
2080 IF Z = 0 THEN GOTO 600
```

Všetky znaky reťazca U$, až po prvú *, počítač uloží do P$.

Ak reťazec P$ je dlhší než reťazec I$, začne hľadať ďalšie slovo.

Ak I$ = P$, pokračuje riadkom 2140.

Ak v I$ nenašiel žiadne slovo, odpoveď vytvorí náhodne.

```
2150 IF A > 1 THEN LET Z$(Z)=I$(TO A-1)+" "+W$(B)+" "
2160 IF A < 2 THEN LET Z$(Z)=W$(B)+" "
2170 LET I$=I$(A+L TO)
```

Ak A > 1, začiatok reťazca I$ spolu s reťazcom W$ uloží do premennej Z$. Ak A < 2, uloží do Z$ iba W$.

```
2240 DATA "JA SOM ","TY SI **","TY SI *","JA SOM *"
2250 DATA "JA ****","TY *****","TY ****","JA *****"
2260 DATA "MÔJ ***","TVOJ ****","TVOJE *","MOJE ***"
2270 DATA "TVOJ **","MÔJ ****","MNE ***","TEBE ***"
2280 DATA "TY ****","POČÍTAČE"
2290 RETURN
```

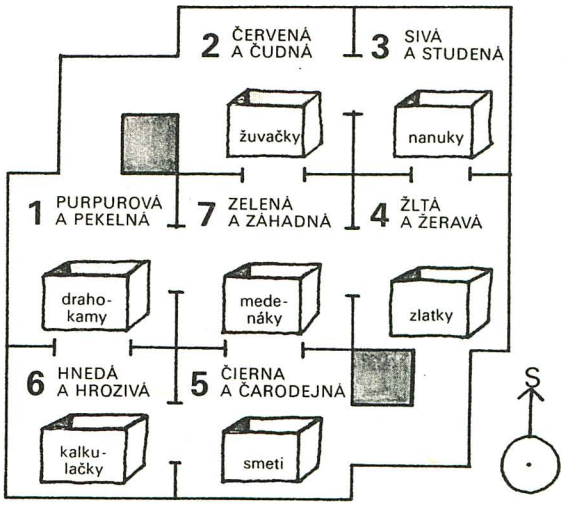
Reťazec doplň na potrebnú dĺžku hviezdičkami (nemôžeš použiť medzery, pretože tie sú v reťazcoch dôležité a nedajú sa vynechať).

Ako napísať program Poklad

Na nasledujúcich stranách nájdeš krok po kroku vysvetlený program pre hru Poklad. Úlohou hráča je prejsť bludiskom siedmich komnát a z každej získať poklad. Program je dosť zložitý, preto buď pri jeho vytváraní pozorný. Každú časť si najprv otestuj, porovnaj s výpisom programu na stranách 80–81, a až potom pokračuj vo vytváraní ďalšej časti. Skôr ako začneš program písať, mali by ti byť jasné pravidlá hry.

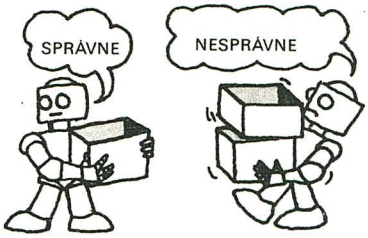
Pravidlá hry

Na obrázku vpravo je bludisko so siedmimi tajomnými komnatami. Každá má meno a je v nej truhlica s pokladom. Mapa bludiska je uložená v pamäti počítača, ale nie je nakreslená na obrazovke. Hráč musí zistiť, ako sú komnaty navzájom pospájané. Na začiatku hry sa ocitne v náhodne vybranej komnate. Jeho úlohou je zhromaždiť všetky truhlice v jednej komnate, pričom môže vykonať iba ohraničený počet krokov.




```
POČÍTAČ ROZUMIE TÝMTO PRÍKAZOM
S,V,J,Z    :POHYB NA SEVER,VÝCHOD,JUH,ZÁPAD
ZDVIHNI    :ZDVIHNE TRUHLICU
POLOŽ      :POLOŽÍ TRUHLICU
KDE        :INFORMUJE O POLOHE TRUHLÍC
SOS        :VYPÍŠE PRAVIDLÁ HRY
```

Hráč môže počítaču zadávať takéto príkazy.



Aby to nemal hráč také ľahké, naraz môže prenášať len jednu truhlicu.

Príklady

1
```
SI V 4. KOMNATE
JE ŽLTÁ A ŽERAVÁ
SÚ V NEJ: ZLATKY
ČO CHCEŠ ROBIŤ?
?ZDVIHNI
ZDVIHOL SI TRUHLICU, V KTOREJ
SÚ ZLATKY
```

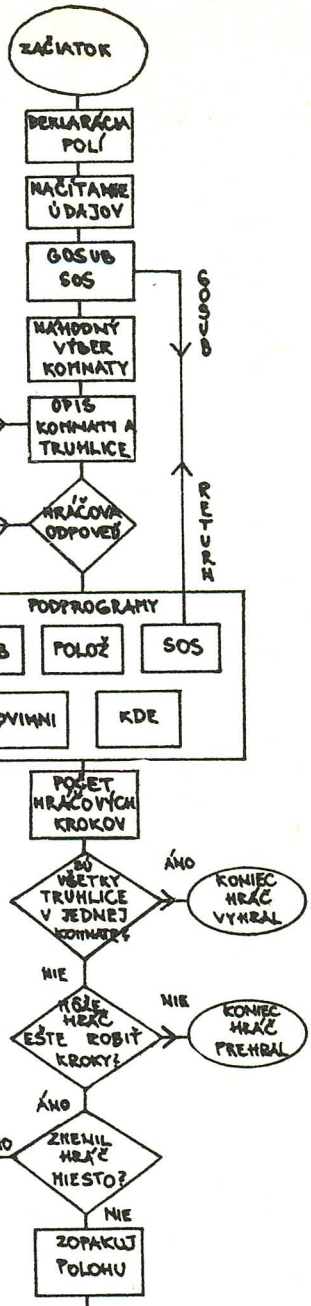
2
```
EŠTE STÁLE SI V 4. KOMNATE
ČO CHCEŠ ROBIŤ?
?Z
```

3
```
SI V 7. KOMNATE
JE ZELENÁ A ZÁHADNÁ
SÚ V NEJ: MEDENÁKY
ČO CHCEŠ ROBIŤ?
?POLOŽ
ULOŽIL SI ZLATKY DO 7. KOMNATY
```

4
```
EŠTE STÁLE SI V 7. KOMNATE
ČO CHCEŠ ROBIŤ?
?J
```

Píšeme program

Vpravo vidíš vývojový diagram. Je v ňom zachytená štruktúra programu a poradie, v akom sa vykonávajú jednotlivé kroky. Na niekoľkých nasledujúcich stranách sú postupne vysvetlené všetky kroky programu. Nech ťa netrápi, že niektoré časti nenasledujú priamo za sebou. Ak dodržíš navrhnuté číslovanie riadkov, kroky sa vykonajú v správnom poradí.



1 Deklarácia polí a načítanie údajov (riadky 100–260)

Všetky údaje o bludisku z predchádzajúcej strany musíš uložiť do pamäte počítača. Budeš na to potrebovať niekoľko polí.

Polia S, V, J, Z

Polia S, V, J, Z deklaruj ako prvé. Budú v nich uložené informácie o tom, do ktorej miestnosti sa hráč dostane, ak sa pohne smerom na sever, východ, juh a západ. Deklaráciu polí napíš do prvého riadka (100)*. Každé pole má 7 prvkov.

	S	V	J	Z
1	2	7	6	0
2	0	3	7	1
3				
4				
5				
6				
7				

Zostav si najprv takúto tabuľku. Čísla v prvom stĺpci zľava sú čísla komnát. Údaje do riadka doplníš tak, že sa pozrieš, ktoré komnaty sa nachádzajú severne, východne, južne a západne od komnaty, ktorej číslo je v prvom stĺpci daného riadka. Prvé dva riadky sme už vyplnili.

Načítanie údajov

```
110  FOR K=1 TO 7
120  READ S(K),V(K),J(K),Z(K)
130  NEXT K

2000 DATA 2,7,6,0
```

Za deklaráciou polí bude nasledovať cyklus, v ktorom sa do polí načítajú údaje. Údaje odpíš z tabuľky do riadkov s DATA.

* Ak máš ZX81, v jednom riadku môžeš deklarovať len jedno pole, preto začni program riadkom 10.

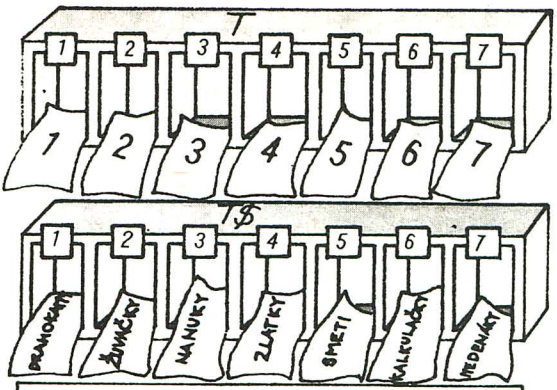
Pole D$

```
2100 DATA PURPUROVÁ A PEKELNÁ,ČERVENÁ A ČUDNÁ
```

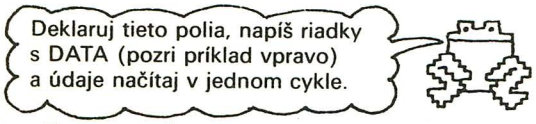
Ďalej budeš potrebovať pole D$, v ktorom budú opisy komnát. Deklaruj ho na začiatku programu a pomocou mapy bludiska napíš riadky s DATA.

Polia T$ a T

Nakoniec budeš potrebovať ešte 2 polia. Do poľa T$ uložíš obsahy truhlíc a do T údaje o tom, v ktorej komnate sa truhlice práve nachádzajú. Pritom číslo komnaty s truhlicami je uložené v T na tom istom mieste, ako je uložený obsah truhlice v T$. Napr. na začiatku hry je T(2) rovné 2, a to je číslo miestnosti, v ktorej je truhlica s T$(2), teda so žuvačkami. Pri hre sa čísla v T menia podľa toho, ako hráč prenáša truhlice.




```
2200 DATA DRAHOKAMY,1,ŽUVAČKY,2
2210 DATA NANUKY,3,ZLATKY,4
```



Premenné

M	C	F	WXY
Počet krokov, ktoré urobil hráč.	Ukazuje, či hráč nesie truhlicu.	Podmienková premenná (prepínač). Jej význam vysvetlíme neskôr.	Premenné na uchovávanie dočasných údajov.

Tabuľka vysvetľuje, na čo sa premenné v programe používajú. Každej premennej priraď počiatočnú hodnotu 0. Začni riadkom 300.

2 Podprogram SOS (riadky 1000–1120)

```
V BLUDISKU JE 7 KOMNÁT A V KAŽDEJ
JE TRUHLICA S POKLADOM. TVOJOU
ÚLOHOU JE ZHROMAŽDIŤ VŠETKY
TRUHLICE V JEDNEJ KOMNATE.
```

Tento podprogram vysvetlí hráčovi pravidlá hry. Príkaz GOSUB umiestni do riadka 300 a podprogram nech sa začína riadkom 1000. Vypíšu sa v ňom pravidlá (vľavo) a príkazy, ktorým počítač rozumie (strana 74).

3 Náhodný výber komnaty (riadok 350)

Číslo komnaty, v ktorej sa hráč nachádza, bude zaznamenané v premennej R. Na začiatku sa poloha hráča určí náhodne. V riadku 350 ulož do premennej R náhodne vybrané číslo od 1 do 7.

4 Opis komnaty a truhlice (riadky 400–470)

```
SI V 7. KOMNATE
JE ZELENÁ A ZÁHADNÁ
SÚ V NEJ: SMETI
```

```
SI V 4. KOMNATE
JE ŽLTÁ A ŽERAVÁ
JE PRÁZDNA
```

```
SI V 1. KOMNATE
JE ČERVENÁ A ČUDNÁ
SÚ V NEJ: DRAHOKAMY
```

Najprv musíš hráčovi oznámiť, v ktorej komnate sa nachádza. Číslo komnaty R použi ako index poľa D$. Hore vidíš, čo by sa malo objaviť na obrazovke.

Opis truhlice

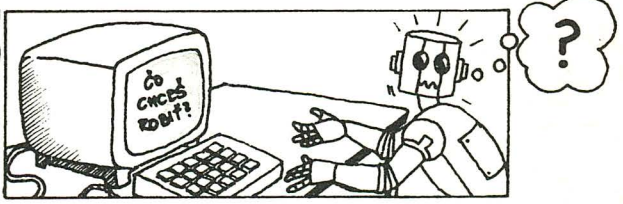
```
430 FOR K=1 TO 7
440 IF T(K) = R THEN PRINT T$(K):
    LET F=1
450 NEXT K
460 IF F = 0 THEN PRINT "JE PRÁZDNA"
470 LET F=0
```

Čo sa nachádza v komnate číslo R, zistíš porovnaním čísla R s prvkami poľa T (v poli T sú informácie o tom, kde sa nachádzajú truhlice). Na porovnávanie použi cyklus (riadky 430–470). Príkaz IF/THEN prikáže počítaču porovnať prvok poľa T s číslom R. Ak sú rovnaké, napíše obsah truhlice a nastaví premennú F na 1.

Podmienková premenná

Podľa obsahu premennej F počítač po skončení cyklu vie, či je miestnosť prázdna, alebo nie. V angličtine sa takejto premennej hovorí zástavová (flag — čítaj fleg), lebo je ako signalizačná zástavka. Ak je podmienka v príkaze IF/THEN splnená, priradí sa premennej F hodnota 1 (zástava je vytiahnutá), ak nie je splnená, F zostáva rovné 0. Nezabudni premennej F priradiť v riadku 470 hodnotu 0, aby si ju mohol znovu použiť.

5 Hráčova odpoveď (riadky 500–560)



V programe sa ďalej zisťuje, čo chce hráč urobiť. Počítač pomocou INPUT načíta jeho príkaz (napr. ZDVIHNI, POLOŽ) do premennej A$. Napíš päť riadkov s príkazom GOSUB, ktoré v závislosti od hráčovho príkazu pošlú počítač do zodpovedajúceho podprogramu. Čísla riadkov, v ktorých sa začínajú podprogramy, doplníš, keď ich budeš mať hotové.

6 Podprogram Pohyb
(riadky 1200–1260)

Pri písaní podprogramu použi tento vývojový diagram.

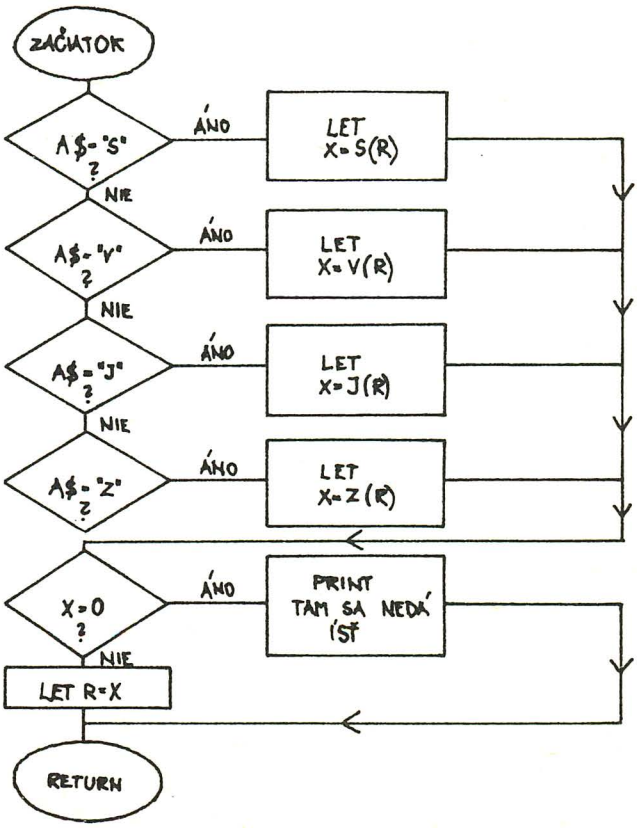
1. Ak chce hráč prejsť do inej komnaty, napíše S, V, J alebo Z. Kam chce ísť, zisti pomocou príkazu IF/THEN.

2. R použi ako index zodpovedajúceho poľa (S, V, J alebo Z) a zisti číslo komnaty, do ktorej sa hráč dostane. Číslo si zatiaľ ulož do premennej X.

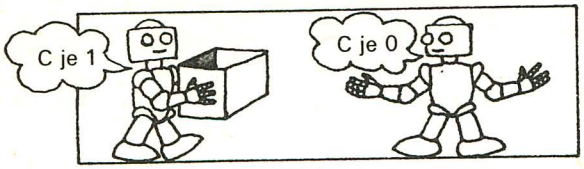
3. Podľa hodnoty X zisti, či hráč môže urobiť požadovaný pohyb. Ak X=0, znamená to, že v určenom smere nie je komnata. Napíš správu o chybe a pomocou GOTO pošli počítač na príkaz RETURN na konci podprogramu.

4. Ak sa X nerovná nule, ulož nové číslo komnaty do R (R=X).

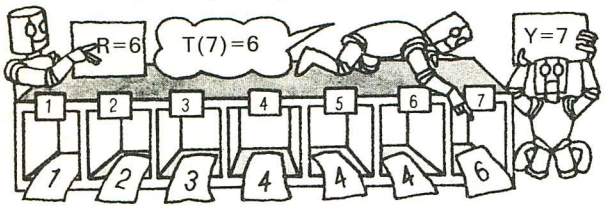
5. Na koniec podprogramu napíš RETURN.



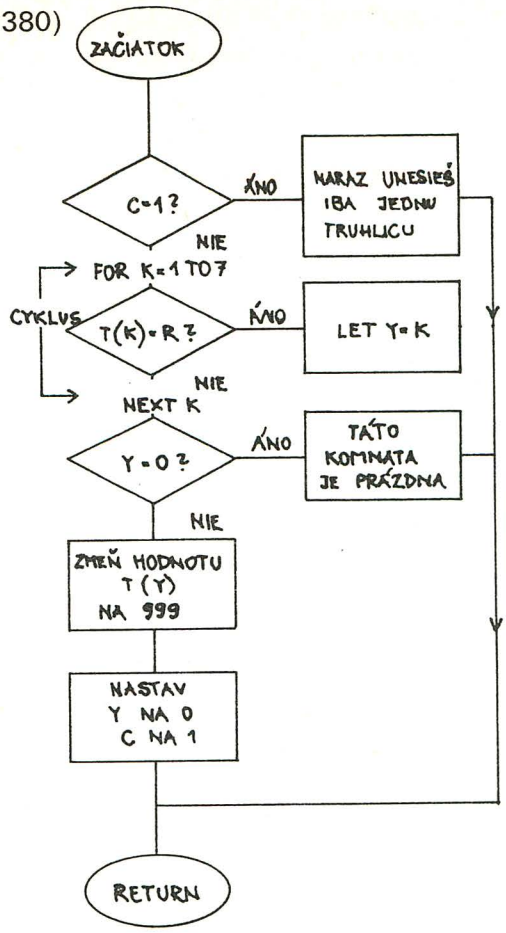
7 Podprogram Zdvihni (riadky 1300–1380)



1. Hráč môže naraz prenášať len jednu truhlicu. Preto musíš skontrolovať, či už niečo nenesie. Urobíš to tak, že zistíš hodnotu premennej C. Ak C=1, hráč nesie truhlicu. Vtedy napíš správu o chybe a pošli počítač na RETURN.

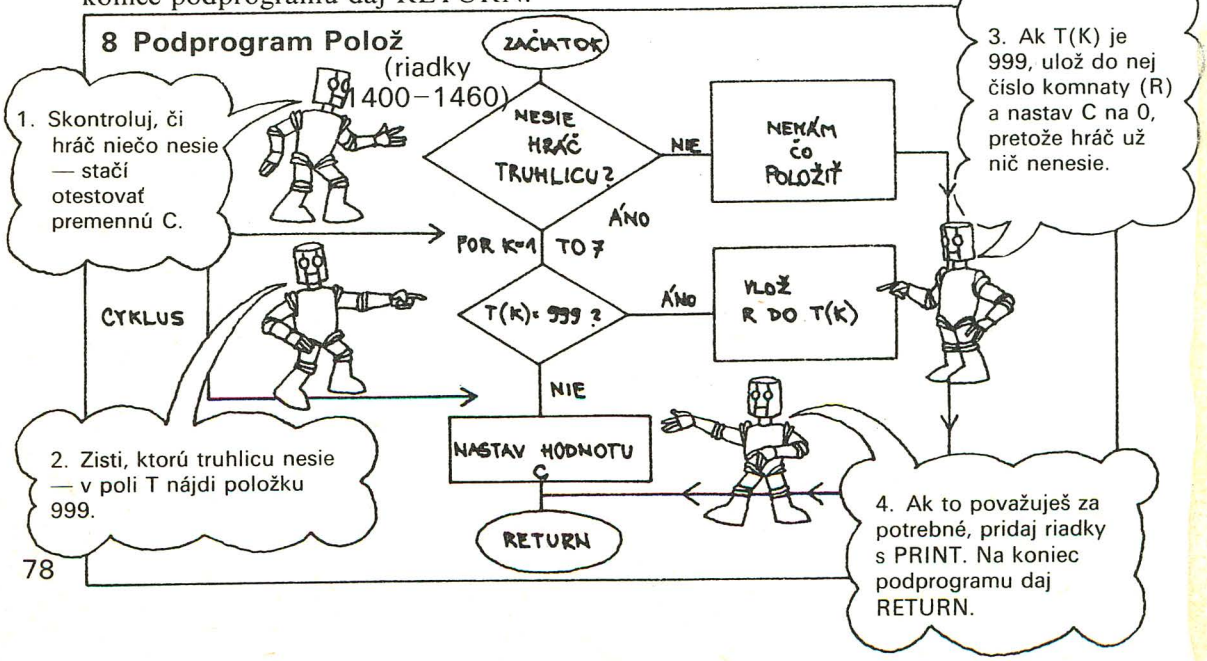


2. Ďalej musíš zistiť, či sa v komnate nachádza truhlica — urob to v cykle príkazom IF/THEN. Ak niektorá položka v T je rovná R, zaznamenaj si príslušný index poľa T, teda hodnotu K v dočasnej premennej Y. Môže sa stať, že v komnate sa nachádza viac truhlíc. V takom prípade bude po skončení cyklu v premennej Y uložený posledný z indexov poľa T.



3. Ak Y=0, v komnate nie je truhlica a môžeš poslať počítač na RETURN. V opačnom prípade je v Y číslo prenášanej truhlice. Zmeň hodnotu T(Y) na 999, aby si vedel, že táto truhlica sa práve premiestňuje. Nastav C na 1, Y na 0 a na koniec podprogramu daj RETURN.

8 Podprogram Polož (riadky 1400–1460)



9. Podprogram Kde (riadky 1500–1590)

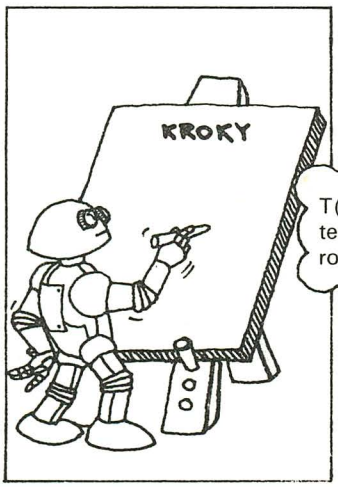
Tento podprogram poskytuje hráčovi informáciu o polohe všetkých truhlíc. Skús podprogram napísať tak, aby písal nasledujúce správy. Najprv si nakresli vývojový diagram.

```
NESIEŠ DRAHOKAMY
ROZMIESTNENIE TRUHLÍC
   2: ŽUVAČKY
   3: NANUKY
   4: ZLATKY
   7: SMETI
   6: KALKULAČKY
   7: MEDENÁKY
```

```
NENESIEŠ NIČ
ROZMIESTNENIE TRUHLÍC
   7: DRAHOKAMY
   7: ŽUVAČKY
   7: NANUKY
   7: ZLATKY
   7: SMETI
   6: KALKULAČKY
   7: MEDENÁKY
```

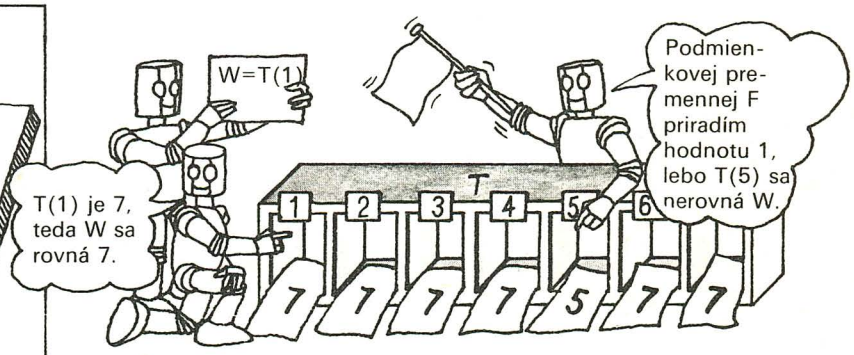
Najprv zisti podľa hodnoty premennej C, či hráč niečo nesie. Ak áno, v cykle zisti index tej položky poľa T, ktorá má hodnotu 999. Obsah nesenej truhlice nájdeš v poli T$ pod tým istým indexom. Rozmiestnenie ostatných truhlíc napíše v ďalšom cykle.

10 Počet hráčových krokov (riadok 600)



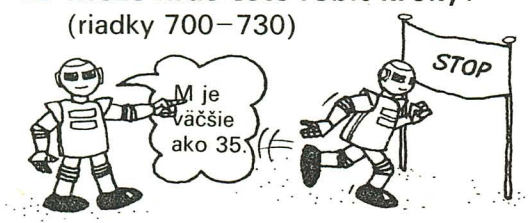
V riadku 600 zvýš obsah premennej M o 1, aby si vedel, koľko krokov hráč spravil.

11 Sú všetky truhlice v jednej komnate? (riadky 610–690)



Hráč vyhrá, ak zhromaždí všetky truhlice v jednej komnate. Či sa mu to už podarilo, zistíš podľa toho, či sú všetky položky poľa T rovnaké. Vezmi jednu položku poľa T, napríklad T(1), a ulož ju do dočasnej premennej W. Potom v cykle porovnaj každú položku poľa T s W. Pomocou podmienkovej premennej si zaznamenaj, či vždy nastala rovnosť alebo nie.

12 Môže hráč ešte robiť kroky? (riadky 700–730)



Rozhodni sa, koľko krokov dovolíš hráčovi vykonať. Ak M je väčšie než toto číslo, hráč prehráva. Napíš správu a ukonči program.

13 Zmenil hráč miesto? (riadky 800–850)

```
EŠTE STÁLE SI V 7. KOMNATE
ČO CHCEŠ ROBIŤ?
?
```

Ak hráč napísal príkaz S, V, J alebo Z, pošli počítač na riadok 400. Tu sa napíše, do ktorej komnaty hráč prešiel a čo sa v nej nachádza. V opačnom prípade pripomeň hráčovi jeho polohu a pošli počítač na riadok 500. Tu sa načíta ďalší hráčov príkaz.

Program Poklad

Na tejto dvojstrane nájdeš úplný výpis programu Poklad v poradí správnom z hľadiska programu.

1 Deklarácia polí a načítanie údajov

```
&100 DIM S(7),V(7),J(7),Z(7),D$(7),
     T$(7),T(7)
 110 FOR K=1 TO 7
@120 READ S(K),V(K),J(K),Z(K)
 130 NEXT K
 140 FOR K=1 TO 7
@150 READ D$(K)
 160 NEXT K
 170 FOR K=1 TO 7
@180 READ T$(K),T(K)
 190 NEXT K
 200 LET M=0
 210 LET C=0
 220 LET F=0
 230 LET W=0
 240 LET X=0
 250 LET Y=0
 260 GCLEAR
```

Údaje sú v riadku 2000 a ďalej.

2 Podprogram SOS

```
300 GOSUB 1000
```

Podprogram sa začína v riadku 1000.

3 Náhodný výber komnaty

```
350 LET R=INT(RND(1)*7+1)
```

4 Opis komnaty

```
400 PRINT "SI V ";R;". KOMNATE"
410 PRINT "JE ";D$(R)
```

Opis truhlice

```
420 FOR K=1 TO 7
430 IF T(K) = R THEN F=1
440 NEXT K
445 IF F = 1 THEN PRINT "SÚ V NEJ: "
450 FOR K=1 TO 7
460 IF T(K) = R THEN PRINT TAB(13);T$(K)
470 NEXT K
480 IF F = 0 THEN PRINT "JE PRÁZDNA"
490 LET F=0: PRINT: PRINT
```

F je podmienková premenná. Priradí sa jej hodnota 1, ak niektorý z prvkov poľa T sa rovná číslu komnaty R (riadok 430).

5 Hráčova odpoveď

```
500 PRINT "ČO CHCEŠ ROBIŤ?"
510 INPUT A$
520 IF A$ = "SOS" THEN GOSUB 1000
530 IF A$ = "S" OR A$ = "V" OR A$ = "J" OR
    A$ = "Z" THEN GOSUB 1200
540 IF A$ = "ZDVIHNI" THEN GOSUB 1300
550 IF A$ = "POLOŽ" THEN GOSUB 1400
560 IF A$ = "KDE" THEN GOSUB 1500
```

Podľa hráčovho príkazu (riadok 510) sa v riadkoch 520 – 560 rozhoduje, ktorý podprogram sa má vykonať.

10 Počet hráčových krokov

```
600 LET M=M+1
```

11 Sú všetky truhlice v jednej komnate?

```
610 LET W=T(1)
620 FOR K=2 TO 7
630 IF W <> T(K) THEN LET F=1
640 NEXT K
655 IF F = 1 THEN GOTO 690
660 PRINT "VÝBORNE. ZHROMAŽDIL SI VŠETKY
    TRUHLICE"
670 PRINT "V ";R;". KOMNATE ZA ";M;"
    KROKOV!"
680 STOP
690 LET F=0
```

Obsah premennej T(1) sa v riadku 610 uloží do W. V cykle sa W porovná so všetkými ostatnými číslami uloženými v T. Ak niektoré z porovnávaných čísel je iné ako W, zdvihne zástavu (priradí F hodnotu 1). Ak sú všetky čísla z poľa T rovnaké, zostáva F rovné 0. To znamená, že všetky truhlice sú v jednej komnate.

12 Môže hráč ešte robiť kroky?

```
700 IF M <= 35 THEN GOTO 730
710 PRINT "ĽUTUJEM. VYČERPAL SI VŠETKY
    KROKY."
720 STOP
730 PRINT
```

Ak M je väčšie ako 35, hráč vyčerpal všetky kroky a program sa zastaví. Počet povolených krokov môžeš zmeniť.

13 Zmenil hráč miesto?

```
800 IF A$ = "S" OR A$ = "V" OR A$ = "J" OR
    A$ = "Z" THEN GOTO 400
810 PRINT "EŠTE STÁLE SI V ";R;". KOMNATE"
850 GOTO 500
```

2 Podprogram SOS

```
1000 PRINT "V BLUDISKU JE
     7 KOMNÁT A V KAŽDEJ "
1010 PRINT "JE TRUHLICA
     S POKLADOM. TVOJOU"
1020 PRINT "ÚLOHOU JE
     ZHROMAŽDIŤ VŠETKY"
1030 PRINT "TRUHLICE
     V JEDNEJ KOMNATE"
1040 PRINT
1050 PRINT "POČÍTAČ ROZUMIE
     TÝMTO PRÍKAZOM:"
1060 PRINT "SOS       : NAPÍŠE
     PRAVIDLÁ HRY"
1070 PRINT "S,V,J,Z   : POHYB NA
     SEVER,VÝCHOD,JUH,ZÁPAD"
1080 PRINT "ZDVIHNI   : ZDVIHNE
     TRUHLICU"
1090 PRINT "POLOŽ     : POLOŽÍ
     TRUHLICU"
1100 PRINT "KDE       : INFORMUJE
     O POLOHE TRUHLICE"

@1110 PRINT: PRINT
 1120 RETURN
```

Počítač napíše pravidlá hry a príkazy, ktorým rozumie. Zakaždým keď hráč napíše SOS, vykoná tento podprogram.

6 Podprogram Pohyb

```
 1200 IF A$ = "S" THEN LET X=S(R)
 1210 IF A$ = "V" THEN LET X=V(R)
 1220 IF A$ = "J" THEN LET X=J(R)
 1230 IF A$ = "Z" THEN LET X=Z(R)
@1240 IF X = 0 THEN PRINT "TAM SA NEDÁ
      ÍSŤ": GOTO 1260
 1250 LET R=X
 1260 RETURN
```

Číslo komnaty, ktorá sa nachádza určeným smerom, uloží do X. Použije pri tom R ako index zodpovedajúceho poľa. Ak X je rovné 0 (riadok 1240), v určenom smere nie je komnata.

7 Podprogram Zdvihni

```
@1300 IF C = 1 THEN PRINT "NARAZ MÔŽEŠ
      NIESŤ IBA JEDNU TRUHLICU": GOTO 1370
 1310 FOR K=1 TO 7
 1320 IF T(K) = R THEN LET Y=K
 1330 NEXT K
 1340 IF Y = 0 THEN PRINT "TÁTO KOMNATA
      JE PRÁZDNA": GOTO 1370
 1350 LET T(Y)=999
 1360 PRINT "ZDVIHOL SI TRUHLICU,
      V KTOREJ SÚ ";T$(Y)
@1370 LET C=1:LET Y=0
 1380 RETURN
```

V riadku 1320 počítač postupne porovná čísla uložené v poli T s R. Ak sú rovnaké, index prvku, ktorý sa rovná R, uloží do premennej Y. Po skončení cyklu vyberie pomocou Y prvok poľa T a zmení jeho hodnotu na 999 (riadok 1350). Ak Y = 0 (riadok 1340), znamená to, že komnata je prázdna.

8 Podprogram Polož

```
@1400 IF C = 0 THEN PRINT "NEMÁM
      ČO POLOŽIŤ": GOTO 1450
 1410 FOR K=1 TO 7
 1420 IF T(K) = 999 THEN PRINT "ULOŽIL
      SI ";T$(K);" DO ";R;". KOMNATY"
 1430 IF T(K) = 999 THEN LET T(K)=R
 1440 NEXT K
 1450 LET C=0
 1460 RETURN
```

Nájde prvok poľa T, ktorý má hodnotu 999 (riadok 1420), a zmení jeho hodnotu na R (riadok 1430).

9 Podprogram Kde

```
@1500 IF C = 0 THEN PRINT
      "NENESIEŠ NIČ": GOTO 1550
 1510 PRINT "NESIEŠ "
 1520 FOR K=1 TO 7
 1530 IF T(K) = 999 THEN
      PRINT TAB(10);T$(K)
 1540 NEXT K
 1550 PRINT "ROZMIESTNENIE TRUHLÍC:"
 1560 FOR K=1 TO 7
 1570 IF T(K) <> 999 THEN PRINT
      "     ";T(K);" : ";T$(K)
 1580 NEXT K
 1590 RETURN
```

1 Údaje

```
@2000 DATA 2,7,6,0
 2010 DATA 0,3,7,1
 2020 DATA 0,0,4,2
 2030 DATA 3,0,5,7
 2040 DATA 7,4,0,6
 2050 DATA 1,5,0,0
 2060 DATA 2,4,5,1
```

Toto sú údaje pre polia S, V, J a Z.

```
2100 DATA PURPUROVÁ A PEKELNÁ,ČERVENÁ
     A ČUDNÁ
2110 DATA SIVÁ A STUDENÁ,ŽLTÁ A ŽERAVÁ,
     ČIERNA A ČARODEJNÁ
2120 DATA HNEDÁ A HROZIVÁ,ZELENÁ
     A ZÁHADNÁ
```

Údaje pre pole D$.

```
2200 DATA DRAHOKAMY,1,ŽUVAČKY,2
2210 DATA NANUKY,3,ZLATKY,4
2220 DATA SMETI,5,KALKULAČKY,6
2230 DATA MEDENÁKY,7
```

Údaje pre polia T$ a T.

Zmeny pre počítače Sinclair

Zmeny platia pre ZX81. Pre ZX Spectrum treba meniť len dimenzie reťazcových polí. Každý príkaz DIM treba oddeliť dvojbodkou, alebo ho dať na nový riadok tak ako pre ZX81.

Riadok 100: Príkazy DIM rozpíš do viacerých riadkov po jednom a deklaruj D$(7,20) a T$(7,23).
Riadok 120: Údaje pre každé pole načítaj príkazom INPUT, údaje zadaj po spustení programu.

```
120 INPUT S(K)
123 INPUT V(K)
125 INPUT J(K)
127 INPUT Z(K)
```

Riadok 150: Použi príkaz INPUT a údaje zadaj po spustení programu.

```
150 INPUT D$(K)
180 INPUT T$(K)
```

Riadok 180: Príkazom INPUT načítaj údaje do poľa T$. Premennej T priraď hodnotu príkazom LET v cykle, aby si po každom spustení programu nemusel zadávať počiatočné rozmiestnenie truhlíc (riadky 193–197).

```
193 FOR K=1 TO 7
195 LET T(K)=K
197 NEXT K
```

Riadok 350: Použi RND.

```
350 LET R=INT(RND*7+1)
```

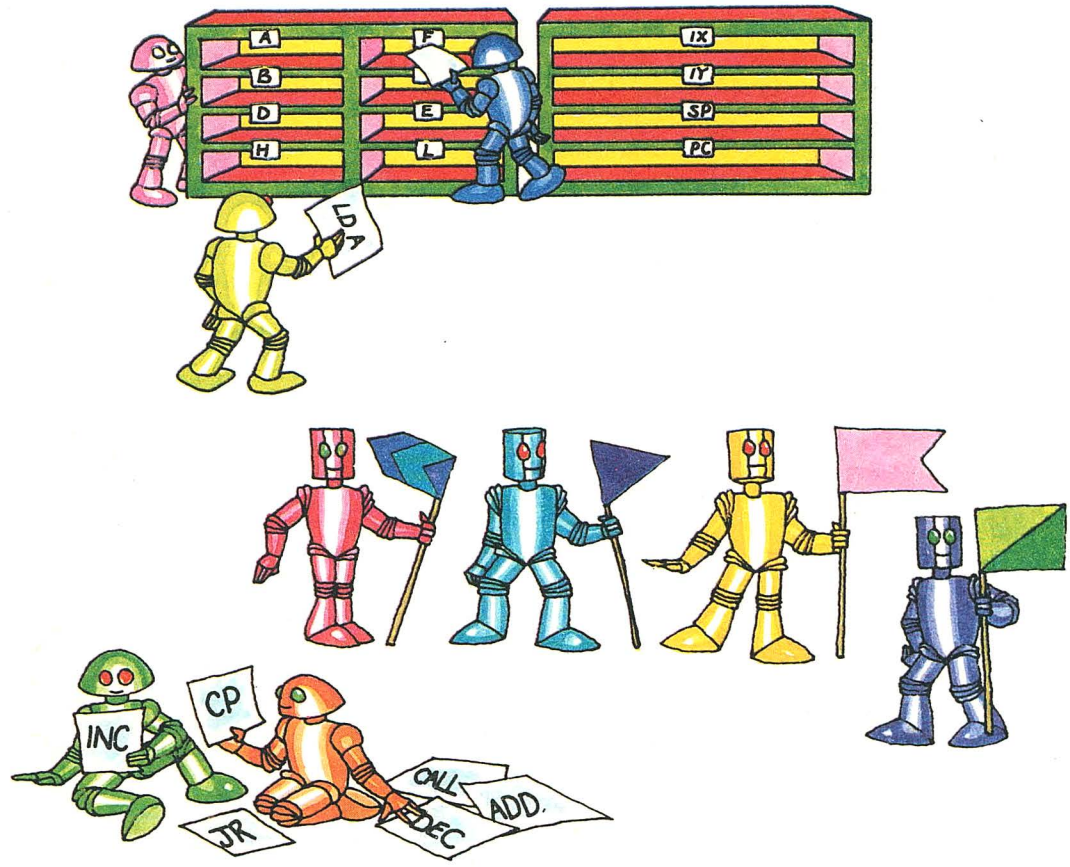
Riadky 490 a 1110: Do ďalšieho riadka napíš príkaz PRINT.
Riadky 1240–1510: Každý príkaz napíš do zvláštneho riadka, ak treba, použi príkaz IF/THEN.

```
1240 IF X = 0 THEN PRINT "TAM SA NEDÁ ÍSŤ"
1245 IF X = 0 THEN GOTO 1260
```

Riadok 2000: Aby si po každom spustení hry nemusel zadávať programu vstupné údaje, nahraď riadky 2000–3000 nasledujúcimi riadkami. Ak budeš chcieť hrať hru odznova, napíš príkaz GOTO 2000 (keby si hru spustil príkazom RUN, počítač by vymazal obsah všetkých premenných).

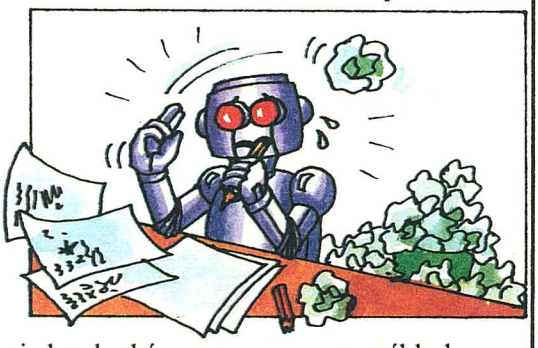
```
2000 PRINT "CHCEŠ HRAŤ HRU EŠTE RAZ?"
2010 INPUT R$
2020 IF R$ = "A" THEN GOTO 193
2030 IF R$ = "N" THEN STOP
2040 GOTO 2000
```

Úvod
do strojového kódu



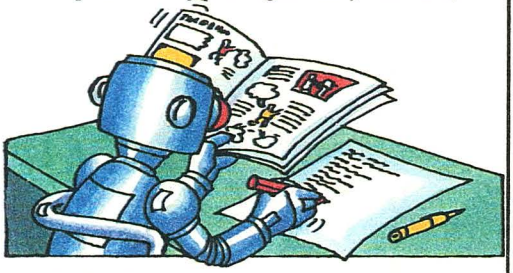
Všetka činnosť počítača sa vlastne vykonáva v strojovom kóde. Výhodou programov v strojovom kóde je to, že sú oveľa rýchlejšie a zaberajú menej pamäte než programy v basicu. Na druhej strane programy v strojovom kóde, ktoré vytvoril niekto druhý, pochopíš oveľa ťažšie než programy v basicu, a strojové programy aj ťažšie vytvoríš.

V tejto časti knižky ti krok za krokom ukážeme, ako sa programuje v strojovom kóde. Naučíš sa písať



jednoduché programy — napríklad na sčítanie dvoch čísel alebo na rozblikanie správy na obrazovke — aj to, ako ich máš na svojom počítači spúšťať.

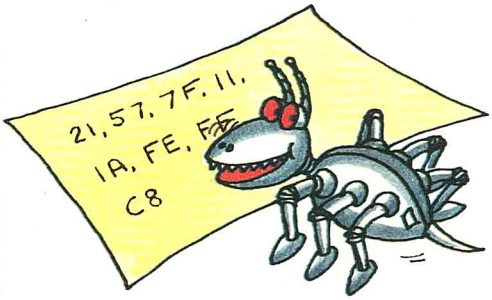
Programy v strojovom kóde sú napísané špeciálne pre počítače s mikroprocesorom Z80 alebo 6502.* Mikroprocesor je integrovaný obvod,



ktorý obsahuje základnú jednotku počítača. Počítače s rôznymi mikroprocesormi rozumejú odlišnému strojovému kódu.

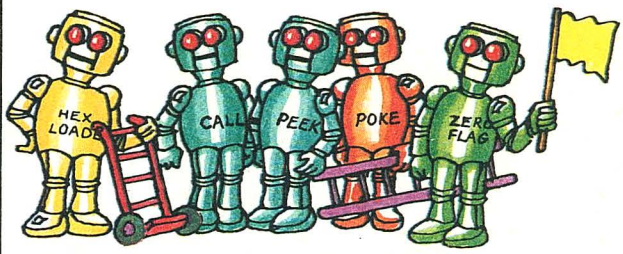
Strojový kód je zložitý a programovanie v ňom je prácne. Musíš dodržať mnoho pravidiel a zapamätať si veľa podrobností. Okrem toho programy v strojovom kóde sa veľmi ťažko čítajú, lebo na prvý pohľad vyzerajú ako postupnosť písmen a čísel bez zjavného zmyslu. Aj chyby sa v nich prejavujú ničivo a ťažko sa hľadajú. Keď budeš programovať v strojovom kóde, musíš byť preto pozorný a všetko si overovať dvakrát alebo aj trikrát. Dúfame však, že sa napriek všetkému hneď na začiatku nezľakneš a nedáš sa odradiť.

Zvyčajne niet dôvodu písať v strojovom kóde dlhé programy — takmer všetko sa dá rovnako dobre naprogramovať aj v basicu. Strojový



kód je však niekedy nenahraditeľný: pomôže zrýchliť hru alebo vytvoriť na obrazovke zvláštne efekty. Na stranách 122—123 sa dozvieš, ako môžeš svoje programy v basicu zlepšiť krátkymi podprogramami v strojovom kóde.

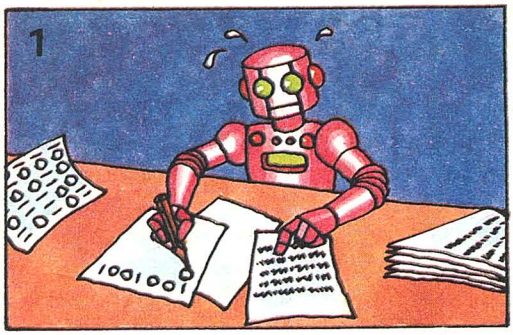
Na konci tejto časti nájdeš tabuľky, ktoré ti pomôžu pri programovaní v strojovom kóde. Zhŕňajú inštrukcie strojového kódu a pravidlá práce s číslami v šestnástkovej sústave.



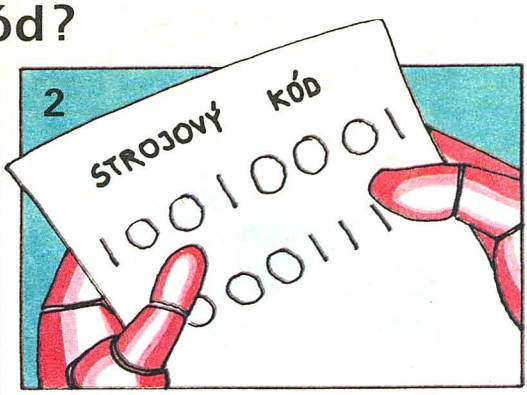
Rovnako dobre ti poslúži aj slovník výrazov, ktoré sa používajú pri programovaní v strojovom kóde. Zoznamovanie so strojovým kódom ti spestria návrhy na písanie krátkych programov a hádanky. (Riešenia sú na strane 128.)

*ZX Spectrum a ZX81 používajú mikroprocesor Z80. Počítače Atari 800XL, Atari 130XE, Commodore 16, 64, 116 používajú mikroprocesor 6502. V československých počítačoch je mikroprocesor 8080, ktorý pozná veľa inštrukcií pre Z80 (takmer všetky programy pre Z80 v tejto knižke pobežia aj na 8080).

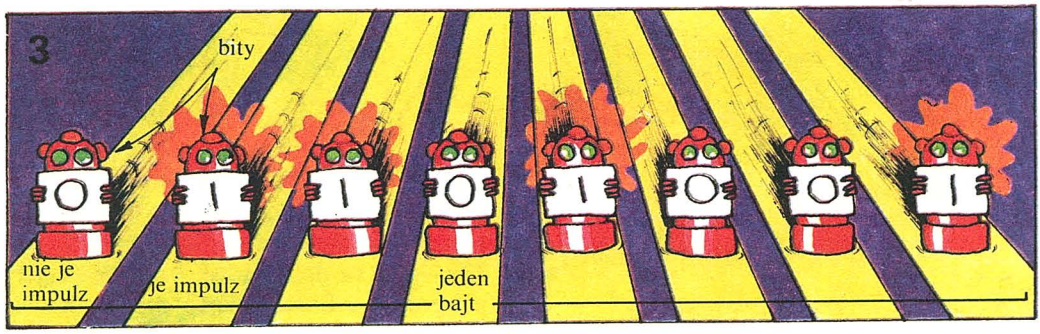
Čo je to strojový kód?



Strojový kód je jazyk, v ktorom pracuje počítač. Ak chceš programovať počítač v basicu, musí mať v pamäti interpreter, ktorý mu hovorí, ako sa príkazy basicu vykonávajú.

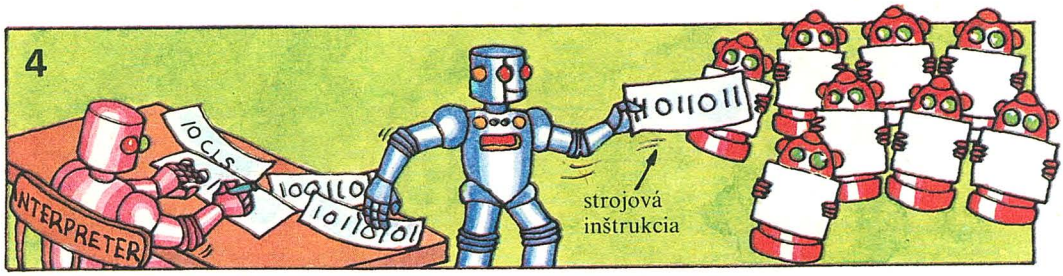


V strojovom kóde je každá inštrukcia aj každý údaj vyjadrený číslom v dvojkovej sústave. Dvojková sústava používa len dve číslice: 1 a 0. Pomocou nich možno zapísať ľubovoľné číslo.*



V počítači sú čísla znázornené elektrickými impulzmi. Ak je impulz, znamená to jednotku, ak nie je, tak nulu. Jednotlivé signály „je impulz" a „nie je impulz" sa nazývajú bity.

Bity sa v počítači pohybujú v skupinách po osem — takúto skupinu nazývame bajt. Každý bajt znázorňuje jednu inštrukciu v strojovom kóde alebo kúsok informácie (údaj).



Počítač „sám od seba" pozná jediný programovací jazyk — strojový kód. Keď napíšeš príkaz v basicu, vykoná ho špeciálny program — interpreter — vyvolaním postupnosti strojových inštrukcií.

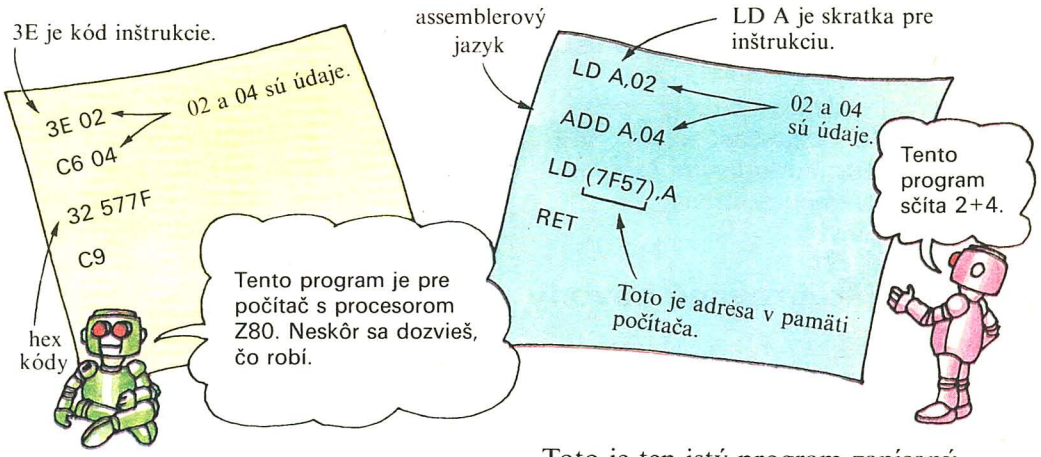
Termín strojový kód sa teda používa aj na označenie programovacieho jazyka, v ktorom sa píšu programy na nižšej úrovni, ako je basic, čiže v tvare oveľa bližšom počítaču než basic. V strojovom kóde musíš rozpísať na jednoduchšie operácie aj také činnosti, pre ktoré v basicu stačí jediný príkaz, napr. zmazanie obrazovky.

* O dvojkovej sústave je viac na strane 108.

Programovanie v strojovom kóde

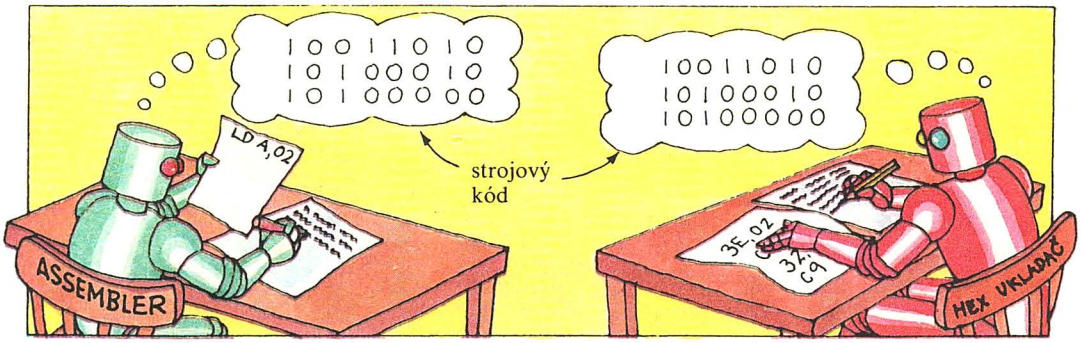
Program v strojovom kóde sa dá zapísať niekoľkými spôsobmi. Inštrukcie môžeš napísať ako čísla v dvojkovej sústave — to je však prácne. Namiesto dvojkovej sústavy môžeš použiť šestnástkovú sústavu (hexadecimálnu, skrátene hex). Keď si na ňu zvykneš, zistíš, že je oveľa výhodnejšia ako dvojková.

Strojové programy sa dajú zapísať aj v assemblerovom jazyku. V ňom sú číselné kódy inštrukcií nahradené niekoľkopísmenovými anglickými skratkami, ktoré sa dajú ľahšie zapamätať. Hovorí sa im mnemonické (ľahko zapamätateľné) mená.



Toto je časť strojového programu zapísaná v šestnástkovej sústave. Táto sústava má šestnásť číslic označených symbolmi 0 až 9 a A až F (neskôr sa o tom dozvieš viac). Číslo na začiatku každého riadka je inštrukcia (napríklad 3E). Je to vlastne šestnástkový zápis jej kódu.

Toto je ten istý program zapísaný v assemblerovom jazyku. Každý riadok sa začína skratkou inštrukcie, ktorá zodpovedá šestnástkovému číslu na ľavom obrázku. Napríklad skratka LD A znamená to isté ako šestnástkové číslo 3E. V oboch programoch je v jednom riadku zapísaná jedna strojová inštrukcia.

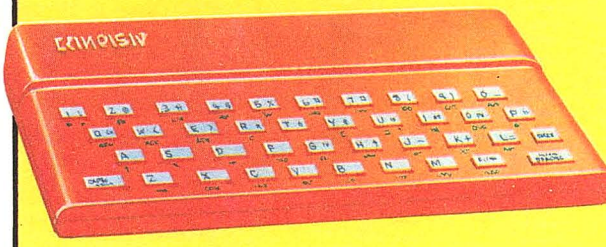


Programu v assemblerovom jazyku počítač priamo neporozumie. Musí mať špeciálny program — assembler, ktorý mu skratky preloží do strojového kódu. Niektoré počítače majú assembler vstavaný, do iných ho treba nahrávať z kazety (slovom assembler sa niekedy označuje aj prekladač aj assemblerový jazyk). I v prípade, že assembler nemáš, program môžeš zapísať pomocou skratiek (tie si ľahšie zapamätáš), no potom ho musíš ručne preložiť do šestnástkovej sústavy. Niektoré počítače takto zapísaný program už vedia prijať, iné opäť potrebujú špeciálny krátky program — šestnástkový ukladač, ktorý preloží čísla zo šestnástkovej sústavy do strojového tvaru. (Na strane 104 nájdeš takýto program.)

Spoznaj svoj počítač

Ak programuješ v strojovom kóde, musíš počítaču presne vysvetliť všetky základné kroky každej činnosti: odkiaľ vybrať a kam uložiť údaje, ako ich napísať na obrazovku a podobne. Pri programovaní v basicu tvoje príkazy rozkladá na základné kroky interpreter. Aby si teda mohol na svojom počítači programovať v strojovom kóde, musíš vedieť viac o jeho činnosti. Táto dvojstrana ti ukáže, z čoho sa počítač skladá a na čo každá časť slúži. Podrobnejšie sa jednotlivými časťami zaoberajú nasledujúce strany.

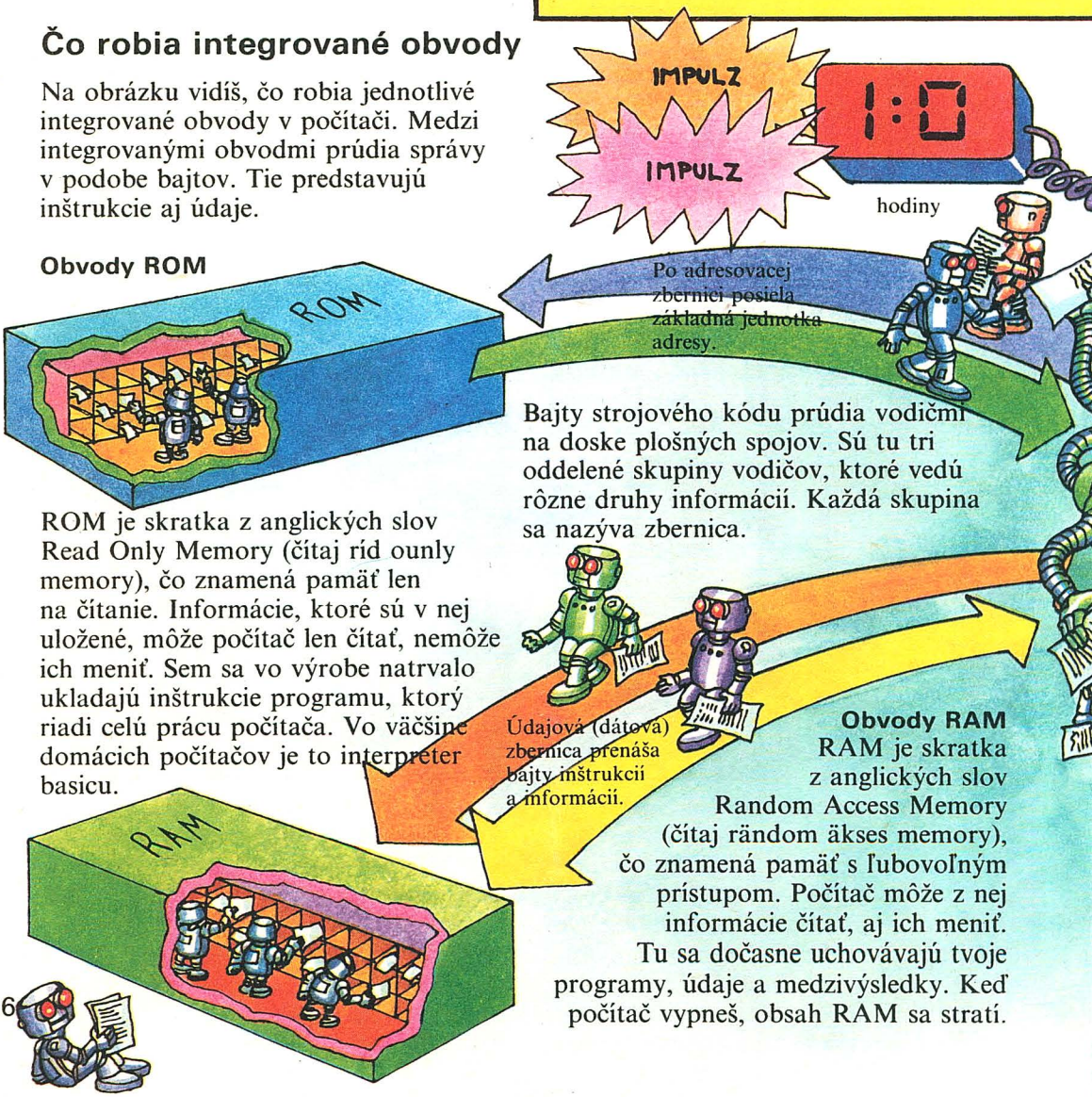
Vnútro počítača

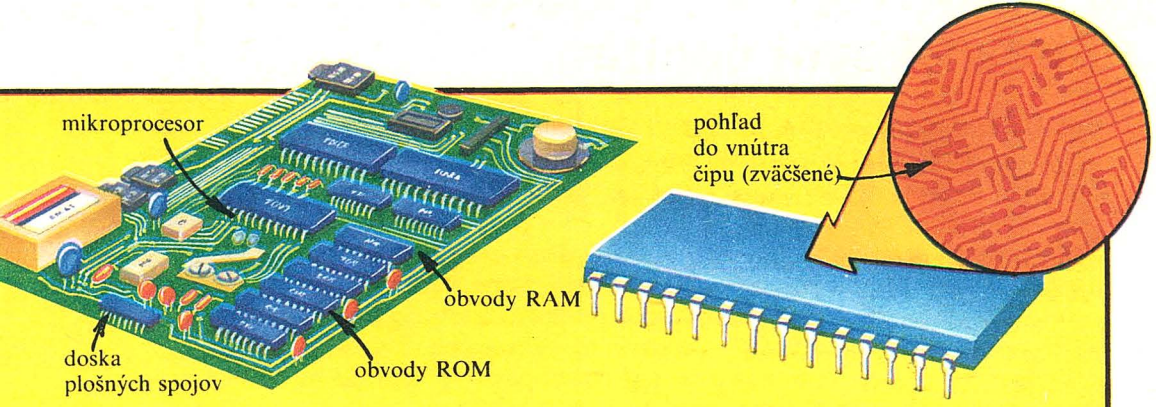


V klávesnici počítača je doska plošných spojov, na ktorej kovové vodiče vedú elektrický prúd. Na ne je pripojené množstvo integrovaných obvodov.

Čo robia integrované obvody

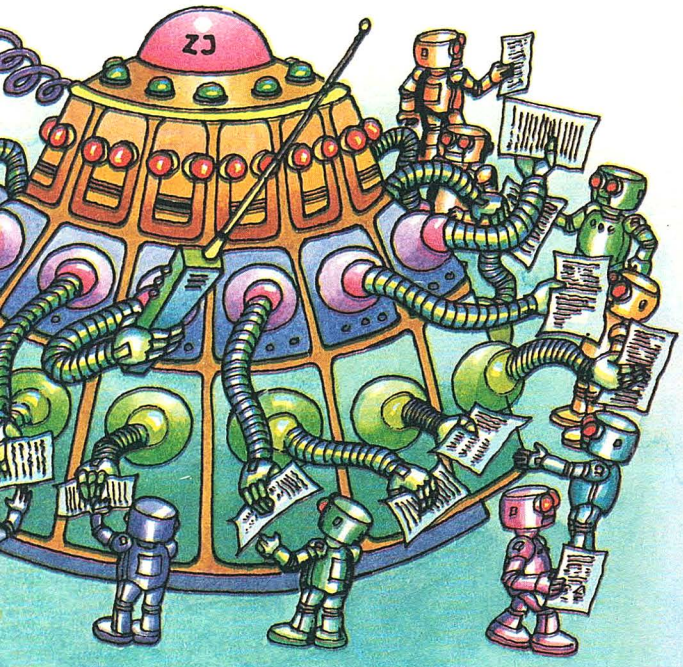
Na obrázku vidíš, čo robia jednotlivé integrované obvody v počítači. Medzi integrovanými obvodmi prúdia správy v podobe bajtov. Tie predstavujú inštrukcie aj údaje.





Vnútro integrovaného obvodu je čip. Obsahuje mikroskopické elektrické obvody, cez ktoré prechádza v rýchlych impulzoch elektrický prúd. Postupnosť signálov „je impulz" a „nie je impulz" tvorí strojový kód počítača. V každom počítači nájdeš niekoľko druhov čipov. Čo ktorý robí, dozvieš sa z obrázka dolu.

Hodiny
Kryštálom riadené hodiny „tikajú" niekoľko miliónkrát za sekundu a riadia tok impulzov v počítači.



Mikroprocesor
Obvod mikroprocesora obsahuje základnú jednotku počítača. Mikroprocesor sčituje čísla, porovnáva informácie, rozhoduje sa a riadi činnosť ostatných obvodov v počítači. Program, ktorý riadi mikroprocesor, je uložený v ROM.

Mikroprocesor tvojho počítača

Existuje niekoľko rôznych druhov mikroprocesorov. Najpoužívanejšie sú Z80 (v ZX Spectrum, ZX81, Sord M5, Sharp MZ811, Sharp MZ821) a 6502 (jeho rôzne modifikácie sú v Atari 800XL a 130XE, Commodore 64 a 128). Československé počítače používajú procesor 8080, ktorý je predchodcom procesora Z80. Niektoré mikroprocesory majú niekoľko modifikácií: napríklad Z80A je rýchlejšia verzia Z80. 6502A a 6510 sú verzie procesora 6502.

Počítače s rovnakým mikroprocesorom rozumejú rovnakému strojovému kódu. Ukážkové programy v tejto knihe sú pre Z80 a 6502.

Pamäť počítača

Pamäť počítača si možno predstaviť ako množstvo škatuliek, z ktorých každá obsahuje jeden bajt informácií, teda inštrukciu alebo údaj v strojovom kóde. Škatuľka sa nazýva pamäťové miesto a má svoje číslo — adresu — podľa nej ju počítač môže nájsť.

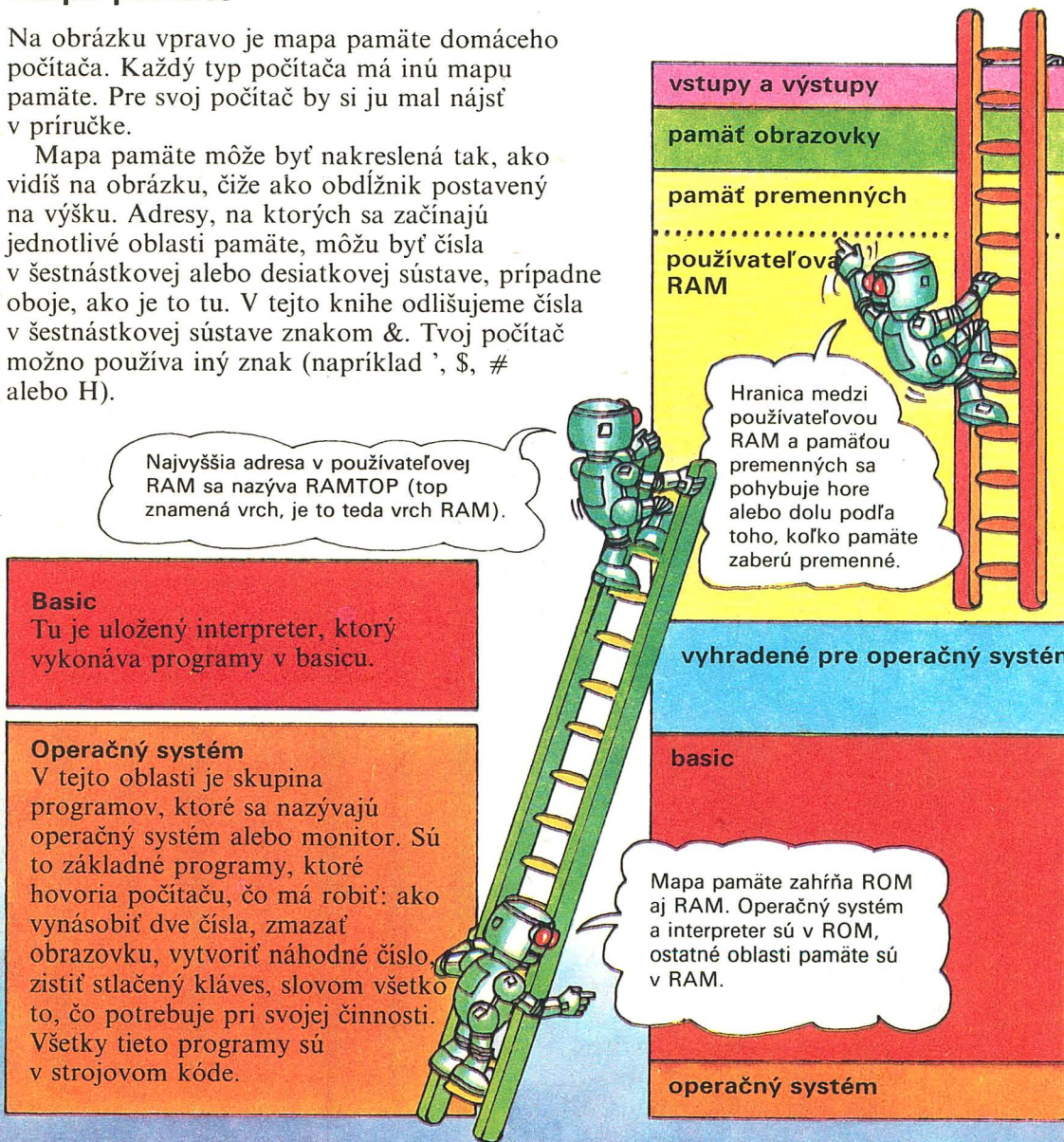
V každej oblasti pamäte je uložený iný druh informácií. Obrázok s adresami a veľkosťami jednotlivých oblastí sa nazýva mapa pamäte.

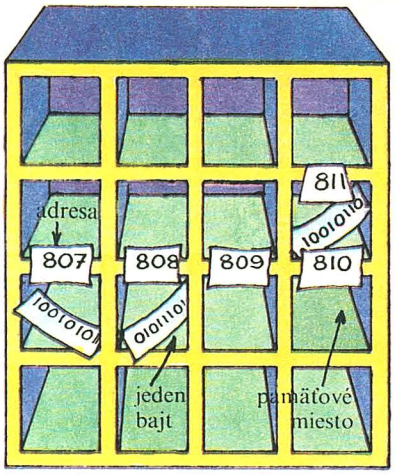
Keď programuješ v strojovom kóde, počítaču musíš presne povedať, kde sa ktoré informácie nachádzajú — spravíš to jednoducho: napíšeš adresu pamäťového miesta. A musíš povedať dokonca aj to, kam sa má uložiť program, potrebuješ teda poznať mapu pamäte svojho počítača.

Mapa pamäte

Na obrázku vpravo je mapa pamäte domáceho počítača. Každý typ počítača má inú mapu pamäte. Pre svoj počítač by si ju mal nájsť v príručke.

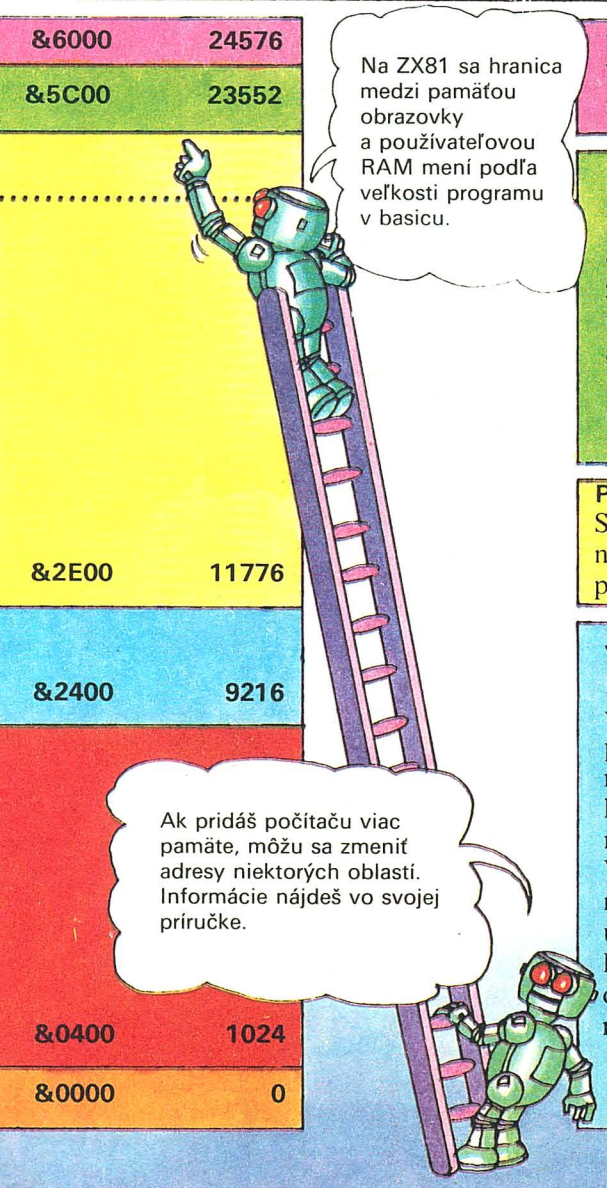
Mapa pamäte môže byť nakreslená tak, ako vidíš na obrázku, čiže ako obdĺžnik postavený na výšku. Adresy, na ktorých sa začínajú jednotlivé oblasti pamäte, môžu byť čísla v šestnástkovej alebo desiatkovej sústave, prípadne oboje, ako je to tu. V tejto knihe odlišujeme čísla v šestnástkovej sústave znakom &. Tvoj počítač možno používa iný znak (napríklad ', $, # alebo H).





Adresy v pamäti

V počítači sú adresy zvyčajne uložené v dvoch bajtoch, čiže v 16 bitoch. Najväčšia pamäť, ktorá sa dá použiť pre počítač s mikroprocesorom Z80 alebo 6502, je 64 kilobajtov (ROM aj RAM spolu), lebo najväčšie číslo, ktoré sa dá napísať šestnástimi ciframi v dvojkovej sústave, je 65 535. Toto je teda najvyššia možná adresa. Keďže pamäťové miesta majú adresy od 0 do 65 535, je ich spolu 65 536. Na každom pamäťovom mieste je uložený 1 bajt. 1024 bajtov je kilobajt (KB), 65 536 bajtov je 64 KB (65 536 : 1024 = 64).



Vstupy a výstupy
Táto časť pamäte je spojená so vstupnými a výstupnými zásuvkami počítača.

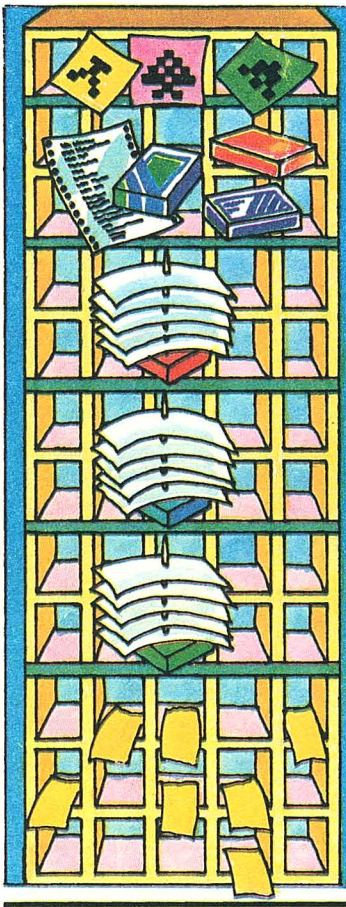
Pamäť obrazovky
Nazýva sa aj videostránka. V tejto oblasti sú uložené informácie práve zobrazované na obrazovke. Všetky informácie, ktoré sa sem uložia, sa automaticky zobrazujú na obrazovke. Každé pamäťové miesto zodpovedá nejakej časti obrazovky.

Používateľova RAM
Sem sa uložia programy v basicu, ktoré napíšeš do počítača. Údaje pre premenné sú v hornej časti tejto oblasti.

Vyhradené pre operačný systém
Do tejto časti RAM počítač ukladá všetky informácie, ktoré potrebuje pre správnu činnosť. Napríklad miesto, kde je kurzor, akou farbou sa kreslí, ktorý kláves bol stlačený, ktorý riadok programu sa práve vykonáva. Väčšinou je táto oblasť rozdelená na ešte menšie úseky — každý je určený na jedno špeciálne použitie. Počítače mávajú pre túto časť pamäte osobitnú mapu. Viac sa o tom dozvieš na ďalšej strane.

Podrobnejší pohľad do pracovnej pamäte počítača

Obrázok v modrej ploche ukazuje podrobnejšie rozdelenie pamäte vyhradenej pre operačný systém. Vo svojej príručke možno nájdeš podrobnejšiu mapu tejto oblasti, prípadne zoznam niektorých adries a ich použitia. Niektoré počítače (napríklad Sinclair) nemajú pamäťové miesta pre operačný systém sústredené v jednej oblasti, ale ich majú rozhádzané po celej pamäti.



Grafické symboly vytvorené používateľom
Ak si vytvoríš vlastné grafické znaky, budú uložené tu.

Vyrovnávacie pamäte
Sú to dočasné pamäte pre informácie, ktoré prichádzajú z klávesnice, alebo sú posielané do tlačiarne či magnetofónu. Často sa označujú anglickým termínom buffer (čítaj bafr, nárazník).

Strojový zásobník
Nazýva sa aj zásobník procesora. Základná jednotka sem ukladá pomocné informácie počas vykonávania programu v strojovom kóde.

Zásobník basicu
Nazýva sa aj zásobník pre GOSUB. Sem sa ukladajú čísla riadkov, kam sa má počítač vrátiť z podprogramu vyvolaného príkazom GOSUB.

Zásobník výpočtov
Sem sa ukladajú medzivýsledky výpočtov.

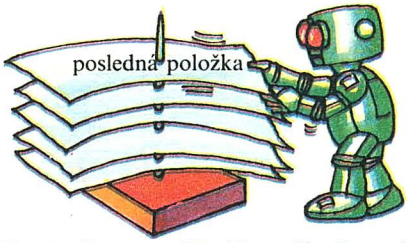
Systémové premenné
Sem sa ukladajú informácie o tom, čo sa deje v počítači. Napríklad je tu uložené číslo riadka a stĺpca, kde sa nachádza kurzor, kód stlačeného klávesu a adresa oblasti, v ktorej sú uložené premenné.

Stránky pamäte

Pamäť počítača sa niekedy rozdeľuje ešte aj na stránky. Na mikropočítačoch býva jedna stránka 256 bajtov, 4 stránky teda zaplnia kilobajt ($4 \times 256 = 1024$).

Pamäťové miesta 0 až 255 sa nazývajú nultá stránka. Oblasti pamäte sa často začínajú na začiatku stránky: napríklad na predchádzajúcej strane sa začína používateľova RAM na stránke 46 (ak počítame stránky od nuly).

Viac o zásobníkoch



Zásobníky umožňujú počítaču uchovávať informácie zvláštnym spôsobom. Informácia uložená do zásobníka ako posledná sa z neho vyberie ako prvá. Tejto pamäti sa hovorí aj „posledný dnu — prvý von“ (po anglicky last in — first out, čítaj last in föst aut, skratka LIFO).

Čísla v šestnástkovej sústave

V strojovom programe zapisujeme čísla a adresy v šestnástkovej sústave. Ukážme si, ako sa čísla z desiatkovej sústavy prevádzajú do šestnástkovej a naopak.

desiatková s.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
šestnástková s.	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

V tabuľke sú šestnástkové čísla (0 až 9 a A až F) a ich hodnoty v desiatkovej sústave. Čísla nad 15 budú mať viac cifier; podobne ako čísla nad 9 v desiatkovej sústave. Hodnota číslice závisí od miesta v čísle.

desiatková s.			
1000	100	10	1
1	2	2	6

šestnástková s.		
256	16	1
4	C	A

V desiatkovej sústave prvá číslica sprava určuje počet jednotiek, druhá desiatok, tretia stovák ($100 = 10 \times 10$) atď.

V šestnástkovom zápise čísla prvá číslica sprava určuje tiež počet jednotiek, ale druhá šestnástok, tretia dvestopäťdesiat šestiek ($256 = 16 \times 16$) atď.

Prevod zo šestnástkovej sústavy do desiatkovej

Číslo 4CA prevedieme do desiatkovej sústavy tak, že vyjadríme v desiatkovej sústave hodnotu každej jeho číslice. Potom hodnotu každej číslice vynásobíme hodnotou prislúchajúcou jej miestu v čísle a výsledky sčítame.

256	16	1
4	C	A
4	12	10 hodnota v desiatkovej s.
×256	×16	×1
1024 +	192 +	10 = 1226

&4CA je 1226.

Vieš previesť &A7 do desiatkovej a 124 do šestnástkovej sústavy? (Riešenie je na s. 124.)

Prevod z desiatkovej sústavy do šestnástkovej

Ak chceš previesť číslo, napr. 1226, do šestnástkovej sústavy, vydeľ ho najprv číslom 256. Tak zistíš, koľkokrát sa 256 nachádza v tvojom čísle. Zvyšok po delení vydeľ 16 — výsledok delenia určuje počet šestnástok v tvojom čísle a zvyšok počet jednotiek. Nakoniec číslo zapíš šestnástkovými číslicami.*

1226:256 = 4 . . . . . 4 je &4
zvyšok 202

202:16 = 12 . . . . . 12 je &C
zvyšok 10 . . . . . 10 je &A

1226 je &4CA

Prevod adries v šestnástkovej sústave

V adrese, napr. &5C64, dve ľavé cifry určujú číslo stránky (pozri predchádzajúcu stranu), ďalej dve cifry určujú umiestnenie v stránke.

Zo šestnástkovej sústavy do desiatkovej

adresa &5C64
číslo stránky = &5C = 92
umiestnenie v stránke = &64 = 100
92 × 256 = 23552 + 100
= 23652
Adresa &5C64 je v desiatkovej sústave 23652.

Pri prevode adresy zo šestnástkovej sústavy do desiatkovej preveď najprv každú dvojicu cifier. Potom vynásob číslo stránky číslom 256 (v jednej stránke je 256 pamäťových miest) a pripočítaj umiestnenie v stránke.

Z desiatkovej sústavy do šestnástkovej

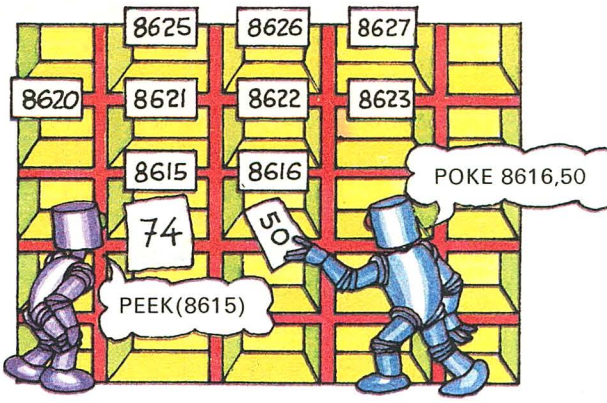
adresa v desiatkovej s. 23652
23652:256 = 92 (číslo stránky)
zvyšok 100 (umiestnenie v stránke)
92:16 = 5 zvyšok 12 = &5C
100:16 = 6 zvyšok 4 = &64
Adresa 23652 je &5C64.

Pri prevode adresy z desiatkovej sústavy do šestnástkovej ju najprv vydeľ číslom 256. Podiel je číslo stránky, zvyšok je umiestnenie v stránke. Potom preveď obe čísla do šestnástkovej sústavy a napíš vedľa seba.

* Na strane 125 sa dozvieš, ako to vypočítaš na kalkulačke.

Príkazy PEEK a POKE

Ak chceš zistiť alebo zmeniť obsah pamäťového miesta, pomôžu ti slová z basicu PEEK (čítaj pík, nazri) a POKE (čítaj pouk, strč). Používajú sa s desiatkovými (na niektorých počítačoch aj so šestnástkovými) zápismi adries pamäťových miest. Nezabudni pred šestnástkovým zápisom čísla napísať špeciálny znak (&, ', $ alebo #). Podrobnosti si zisti v svojej príručke.



PEEK môžeš použiť s ľubovoľnou adresou v pamäti, adresa v POKE však musí byť v RAM, lebo len tam sa dajú informácie meniť.

Príkaz PEEK

```
PRINT PEEK(12345)
46
PRINT PEEK(720)
240
PRINT PEEK(8643)
0
LET A=PEEK(1024)
PRINT A
176
```

```
10 FOR J=700 TO 725
20  PRINT PEEK(J);",";
30 NEXT J
```

```
RUN
36,27,234,56,21,0,0,
0,0,45,32,67,121,45,
47,89,63,21,0,87,241,
202,225,63,87,16,
```

Toto sú obsahy bajtov pamäte zapísané v desiatkovej sústave.

Do pamäte nazrieš, keď napíšeš PEEK a adresu pamäťového miesta. Ak chceš vidieť obsah pamäťového miesta na obrazovke, použi PRINT PEEK alebo výsledok funkcie PEEK priraď pomocou LET do premennej a nechaj vypísať jej obsah. Príklady sú na obrázku.

Skús napísať krátky program pomocou cyklu FOR/NEXT, ktorý ti napíše na obrazovku hodnoty niekoľkých po sebe idúcich pamäťových miest. Pozri si mapu pamäte svojho počítača a experimentuj s adresami v rôznych oblastiach pamäte.

Príkaz POKE

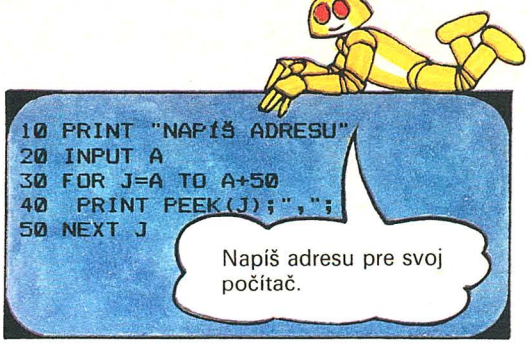
Na obrázku vidíš, ako použiť POKE. Takto môžeš zmeniť hodnotu ľubovoľného pamäťového miesta. Ak však zmeníš hodnoty v oblasti vyhradenej pre operačný systém, môžeš tým prácu počítača narušiť natoľko, že na to, aby pracoval zase normálne, budeš ho musieť vypnúť a znovu zapnúť. Skús napísať krátky program podobný nášmu, ktorým uložíš niekoľko hodnôt do pamäťových miest v používateľovej RAM.

Čísla, ktoré sa ukladajú v POKE, musia byť od 0 do 255, lebo v jednom bajte môžeš uložiť len takéto čísla.

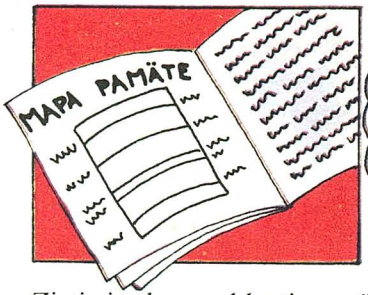
Čo znamenajú čísla

Keď si dáš zobraziť obsah nejakého pamäťového miesta na obrazovke, uvidíš vždy číslo od 0 do 255, keďže v jednom bajte môžu byť uložené len čísla z tohto intervalu. Bajt má len 256 možných hodnôt, každá hodnota však môže mať pre počítač veľa významov podľa toho, v ktorej oblasti pamäte je uložená alebo v ktorom okamihu ju bude počítač spracúvať.

Napríklad dvojkové číslo 00110000 (v desiatkovej s. 48) môže byť kód niektorej inštrukcie strojového kódu alebo niektorého klávesu, či časť adresy iného pamäťového miesta (adresa pozostáva z dvoch bajtov).

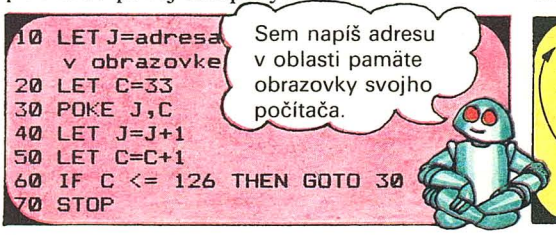


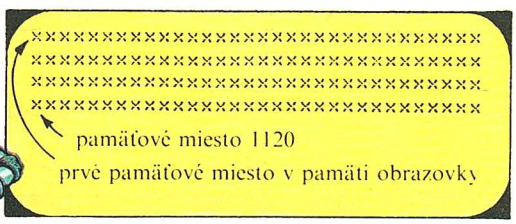
Nájdi vo svojej príručke adresu v ROM, kde je uložený interpreter, a spusti tento program. Na obrazovke sa zjaví 51 čísel. Sú to desiatkové zápisy hodnôt bajtov, ktoré obsahujú kódy inštrukcií nejakej časti interpretera.



Zisti si adresu oblasti pamäte obrazovky a ulož do nej nejaké čísla — tie sa zobrazia na obrazovke. Niektoré počítače pochopia čísla ako kódy písmen, ktoré majú zobraziť, iné ako popis kombinácie bodov, ktoré sa majú rozsvietiť. Zvyšok strany platí len pre počítače prvej skupiny.

Väčšina počítačov používa na kódovanie znakov kód ASCII. Niektoré (napríklad ZX81) používajú osobitný kód. Iné (napríklad Commodore 64 alebo Atari 130XE) poznajú zvláštne grafické znaky, ktoré majú špeciálne kódy. Tabuľky týchto kódov nájdeš vo svojej príručke.





Na počítačoch, ktoré majú odlíšený textový a grafický režim (alebo majú len textový režim), môžeš vypísať všetky znaky pomocou programu z obrázka. Program uloží do pamäte obrazovky kódy od 33, čo je znak "!", po 126, čo je znak "~". Ostatné čísla z intervalu 0–255 sa používajú pre zvláštne klávesy, niekedy aj pre zvláštne grafické symboly (viac sa dozvieš z príručky).

Ak chceš, aby sa zobrazil znak na tom mieste obrazovky, ktoré si vyberieš, musíš vypočítať adresu pamäťového miesta, ktoré mu zodpovedá. Je to jednoduché. Napr. ak sa pamäť obrazovky začína na adrese 1024 a počítač zobrazuje v riadku 32 znakov, prvé písmeno v 4. riadku bude uložené na adrese $1024+(32\times 3)$, čo je 1120 (adresu 1024 považujeme za nulu).

Základná jednotka zvnútra

Vykonávanie všetkých činností v počítači má na starosti základná jednotka (ZJ): vyberá z pamäte inštrukcie a údaje a vykonáva inštrukcie.

ZJ má tri dôležité oblasti: registre — v nich sú uložené spracúvané údaje, aritmeticko-logickú jednotku (ALJ) — v nej sa údaje sčitujú, odčitujú a porovnávajú, a riadiacu jednotku, ktorá riadi všetky tieto činnosti.

Na obrázkoch vidíš usporiadanie registrov procesorov Z80 a 6502.

Tieto obrázky ti ukazujú, aké druhy inštrukcií môže ZJ vykonať. Vie vybrať bajt z pamäte a uložiť ho do registra, preniesť bajt z jedného registra do druhého, spracovať bajty

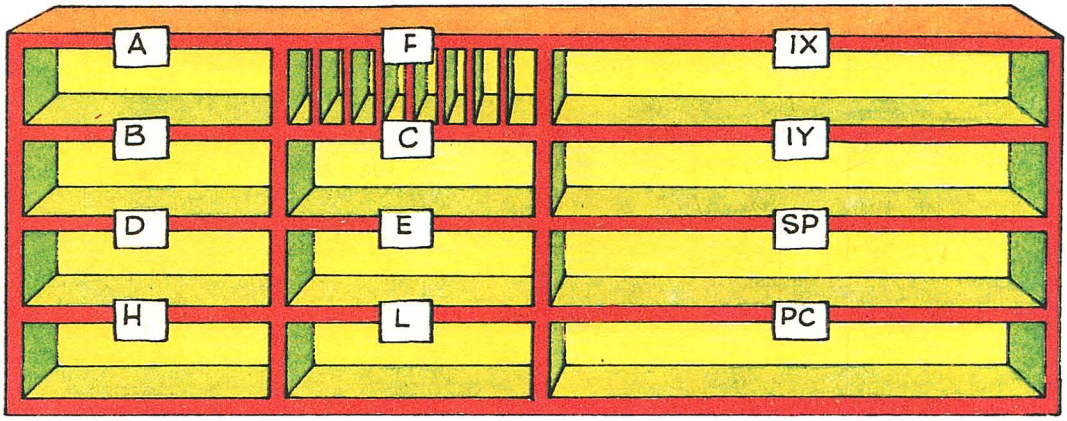
Registre Z80

Hlavný rozdiel medzi Z80 a 6502 je v počte registrov — Z80 ich má viac, môže v nich teda uchovávať niektoré údaje v ZJ, zatiaľ čo 6502 ich musí ukladať späť do pamäte.

A je akumulátor. Je to najdôležitejší register. Uchováva bajty na ich ceste do ALJ a z nej. Naraz v ňom môže byť jeden bajt.

F (flags, čítaj flegz, vlajky) je register podmienok. Z jeho ôsmich bitov sa využíva len šesť. Jednotlivé bity majú svoje osobitné mená (nepleť si ich s menami registrov) a nastavujú sa, keď sú splnené isté podmienky. Napríklad bit C (nazýva sa podmienka prenosu) sa nastaví na jednotku, ak je výsledok sčítania väčší než 255 a nezmestí sa do ôsmich bitov registra, bit S sa nastaví na jednotku, ak je výsledok záporný.

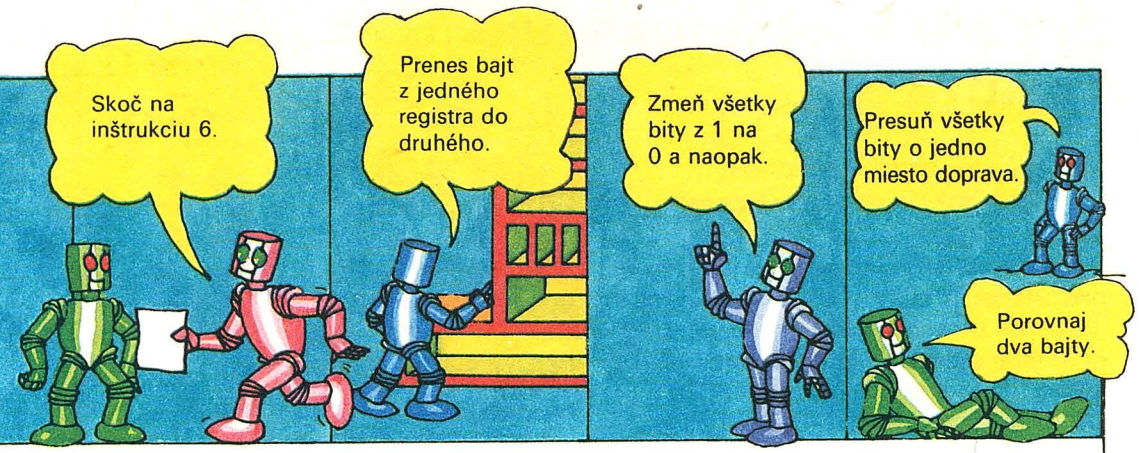
IX a **IY** sa nazývajú indexové registre (skrátene aj indexregistre). Sú šestnásťbitové. Používajú sa ako pomocné registre pri vypočítavaní adries pamäťových miest.



B, C, D, E, H, L sú všeobecné registre, v ktorých môžu byť uložené bajty na ich ceste z pamäte a do nej. V každom registri môže byť jeden bajt, ale niektoré inštrukcie pracujú so šestnásťbitovými pármi registrov BC, DE a HL.

SP (stack pointer, čítaj stek pointr, ukazovateľ zásobníka) je ukazovateľ zásobníka. Je v ňom uložená šestnásťbitová adresa posledného pamäťového miesta v strojovom zásobníku.

PC (program counter, čítaj prougrem kauntr, počítadlo programu) je počítadlo adries. Je v ňom uložená šestnásťbitová adresa pamäťového miesta, z ktorého sa bude vyberať ďalšia inštrukcia na vykonanie. Po vybratí inštrukcie z pamäte sa počítadlo zväčší.



v ALJ a uložiť ich nazad do pamäte. Aj najjednoduchšia činnosť, ako napríklad sčítanie dvoch čísel a zobrazenie výsledku na obrazovke, môže vyvolať viac než sto takýchto jednoduchých inštrukcií — ZJ ich vykonáva asi pol milióna za sekundu.

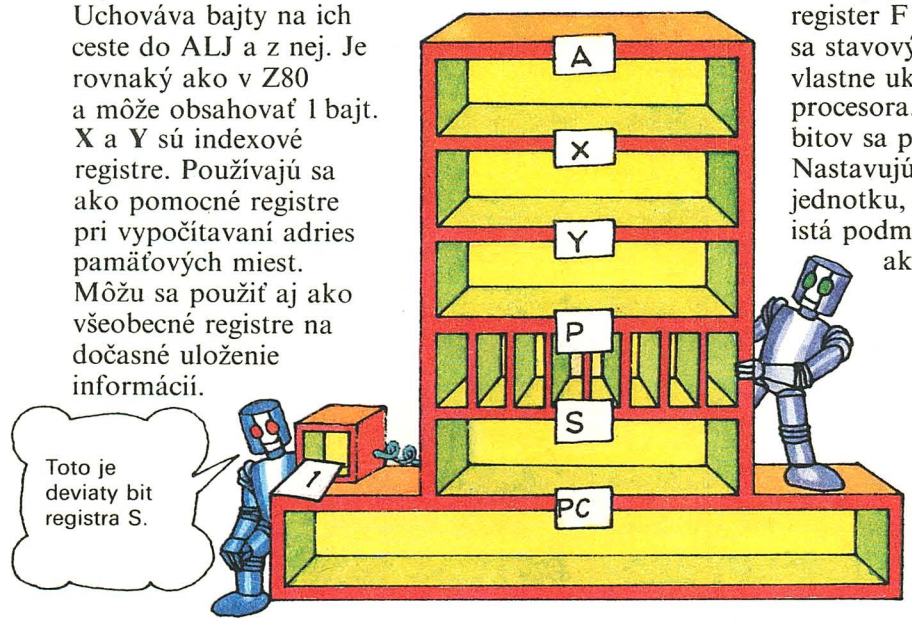
Pre každú operáciu vyberie riadiaca jednotka jeden bajt z ROM alebo RAM. Je to kód inštrukcie, ktorá sa vzápätí vykoná.

Registre 6502

Hlavné registre 6502 sú v podstate rovnaké ako registre Z80, ibaže niektoré majú odlišné mená.

A je akumulátor. Uchováva bajty na ich ceste do ALJ a z nej. Je rovnaký ako v Z80 a môže obsahovať 1 bajt.

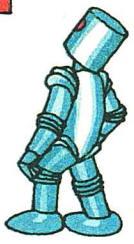
X a **Y** sú indexové registre. Používajú sa ako pomocné registre pri vypočítavaní adries pamäťových miest. Môžu sa použiť aj ako všeobecné registre na dočasné uloženie informácií.



P funguje rovnako ako register F v Z80. Nazýva sa stavový register, lebo vlastne ukazuje stav procesora. Z jeho ôsmich bitov sa používa sedem. Nastavujú sa na jednotku, ak je splnená istá podmienka, napríklad ak je číslo záporné alebo kladné.

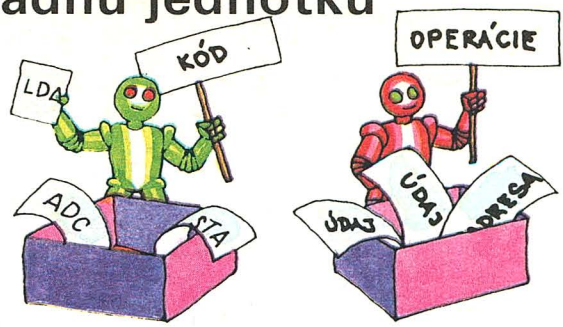
PC je počítadlo adries. Používa sa rovnako ako v Z80.

S je ukazovateľ zásobníka. Ukazuje na posledný údaj v zásobníku — zvláštnej oblasti RAM, kam ZJ ukladá pomocné údaje. V procesore 6502 má deväť bitov, pričom deviaty bit je stále nastavený na jednotku, ukazuje teda vždy do prvej stránky. Zvyšných osem bitov určuje umiestnenie v prvej stránke.

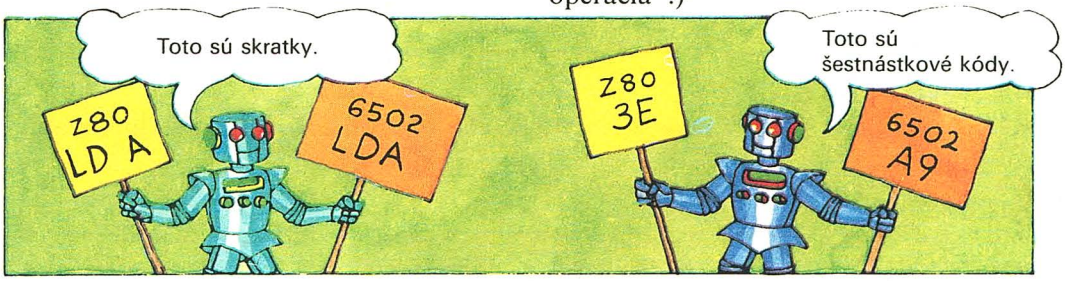


Programujeme základnú jednotku

Program v strojovom kóde je postupnosť inštrukcií, ktorá hovorí základnej jednotke, čo má presne robiť s bajtmi v registroch a pamäti. Môžeš použiť len také inštrukcie, ktoré základná jednotka pozná, teda pre počítač s mikroprocesorom Z80 alebo Z80A sú to inštrukcie Z80 a pre počítač s procesorom 6502, 6502A alebo 6510 zasa inštrukcie 6502. Na konci tejto časti knihy je zoznam inštrukcií Z80 a 6502.

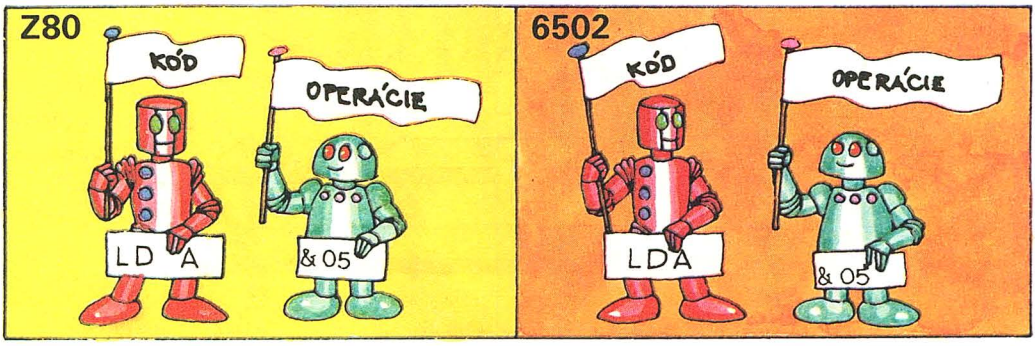


Väčšina inštrukcií strojového kódu pozostáva z dvoch častí: z kódu operácie a z jedného alebo dvoch operandov. Kód operácie hovorí, čo treba urobiť, a operandy, kde nájsť príslušné údaje. (Slovo operand znamená „to, s čím sa má urobiť operácia".)



Kód operácie môže byť zapísaný skratkou — písmenami, ktoré vyjadrujú, čo sa operáciou dosiahne — alebo šestnástkovým zápisom čísla, ktorým je operácia v počítači zakódovaná. Napr. operácia „zober bajt do akumulátora" má pre Z80 skratku LD A a pre 6502 LDA. Hex kód pre Z80 je 3E, pre 6502 je A9.

Skratky si zapamätáš ľahšie ako čísla, lebo im môžeš porozumieť (ak vieš, z akých anglických slov vznikli). Do počítača ich možno zapisovať len vtedy, ak je v ňom assembler — program, ktorý prekladá skratky do čísel kódov* Program sa zapíše pomocou skratiek a potom sa preloží do hex kódov.



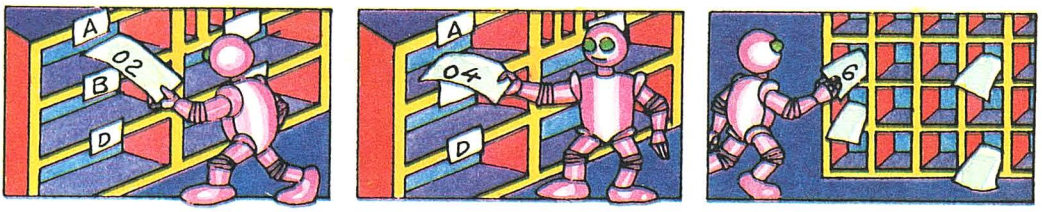
Tu vidíš zápis dvoch inštrukcií pomocou skratiek. Jedna je pre Z80, druhá pre 6502. Obe hovoria počítaču, aby do akumulátora vložil číslo 5 (& označuje, že číslo je v šestnástkovej sústave). V strojovom kóde zapisujeme všetky čísla v šestnástkovej sústave. Znak # označuje pre 6502 konštantu.

* O assembleroch sa dozvieš viac na strane 120.

Jednoduchý program

Ukážeme si dva jednoduché programy (jeden pre Z80, druhý pre 6502) zapísané skratkami, ktoré prikážu základnej jednotke sčítať dve čísla. Program zapísaný skratkami nazývame program v assemblerovom jazyku (alebo kratšie v assembleri), programu v šestnástkovej sústave hovoríme program v strojovom kóde. Na nasledujúcich stranách sa dozvieš, ako preložíš program z assemblera do strojového kódu, ako ho dostaneš do svojho počítača a ako ho spustíš.

Hoci sú inštrukcie rozdielne, oba programy pracujú na rovnakom princípe*. V procesore 6502 sú údaje, s ktorými sa vykonáva operácia, uložené v akumulátore. V Z80 sú v akumulátore alebo v registrovom páre HL (ak sa nezmestia do 1 bajtu).

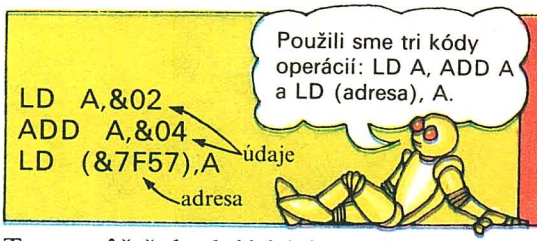


Aby počítač sčítal dve čísla, musí najprv prvé z nich uložiť do akumulátora. Potom k nemu pripočíta druhé a výsledok uloží do pamäte. Tu sú skratky pre tieto inštrukcie:

skratky pre Z80	význam
LD A,číslo	Vlož číslo do akumulátora. LD je skratka z load (čítaj loud, vlož), A je meno akumulátora.
ADD A,číslo	Pripočítaj číslo k A (k akumulátoru).
LD (adresa),A	Vlož číslo z registra A na pamäťové miesto so zadanou adresou.

skratky pre 6502	význam
LDA číslo	Vlož číslo do akumulátora.
ADC číslo	ADC je skratka z add with carry (čítaj ed vid keri, sčítaj s prenosom). Prikazuje sčítať čísla, a ak treba, nastaviť bit C, teda podmienku prenosu. O podmienkach je viac na strane 109.
STA adresa	Ulož obsah A do pamäte na zadanú adresu. ST je skratka zo store (čítaj stó, ulož).

Sčitovací program pre Z80



Teraz môžeš doplniť údaje a adresy. V našom príklade sčitujeme čísla 2 a 4 (sú rovnaké v desiatkovej aj v šestnástkovej sústave) a výsledok ukladáme na adresu &7F57.

Sčitovací program pre 6502

* Program pre iný mikroprocesor, než je v tvojom počítači, môžeš odteraz preskočiť.

Prekladáme program

Jediná možnosť, ako ručne preložiť program zo skratiek do šestnástkových kódov, je použiť tabuľku kódov. Nájdeš ju na konci tejto časti knihy. Pri prekladaní musíš byť pozorný, lebo pre jednu skratku je v nej niekoľko kódov, a to podľa toho, či je operandom číslo, adresa alebo meno registra. Tu sú napríklad rôzne kódy pre naplnenie akumulátora.

Z80		6502	
skratky	**hex kódy**	**skratky**	**hex kódy**
LD A,číslo	3E číslo	LDA číslo	A9 číslo
LD A,(adresa)	3A adresa	LDA adresa	AD adresa

Ak je operandom číslo (konštanta), hovorí sa o priamom operande. Ak je ním adresa, na ktorej je číslo uložené, hovoríme o priamej adrese. O ďalších spôsoboch adresovania sa dozvieš neskôr. Zoznam kódov na konci zahŕňa všetky inštrukcie, ktoré používame v našich programoch.

Sčitovací program pre Z80		Sčitovací program pre 6502	
skratky	**hex kódy**	**skratky**	**hex kódy**
LD A,číslo	3E číslo	LDA číslo	A9 číslo
ADD A,číslo	C6 číslo	ADC číslo	69 číslo
LD (adresa),A	32 adresa	STA adresa	8D adresa

Tu sú šestnástkové kódy pre sčitovacie programy. Postupnosť inštrukcií v skratkách nazývame niekedy zdrojový program (lebo je vstupom pre assembler) a postupnosť ich kódov nazývame cieľový program (lebo je výsledkom práce assemblera).*

Doplnený sčitovací program pre Z80		Doplnený sčitovací program pre 6502	
skratky	**hex kódy**	**skratky**	**hex kódy**

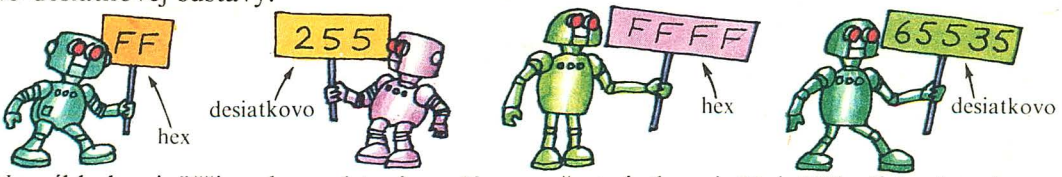
LD A,&02
ADD A,&04
LD (&7F57),A
	3E 02
	C6 04
	32 577F
		LDA #&02
		ADC #&04
		STA &7F57
			A9 02
			69 04
			8D 577F

Teraz môžeš doplniť do programov čísla a adresy. Je to veľmi jednoduché, malá zmena je pri zápise adries — musíš zameniť poradie prvej a druhej dvojice cifier. Viac je o tom na ďalšej strane.

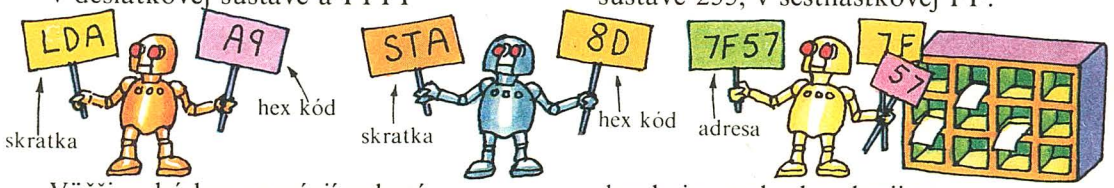
* O assembleri je viac na strane 85.

Viac o šestnástkových kódoch

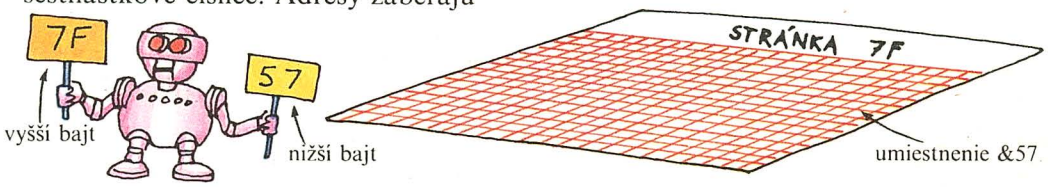
Programy v strojovom kóde sa zapisujú v šestnástkovej sústave preto, lebo prevod z dvojkovej sústavy do šestnástkovej je oveľa jednoduchší než prevod do desiatkovej sústavy.



Napríklad najväčšia adresa, ktorá sa dá napísať do šestnástich bitov, je 65535 v desiatkovej sústave a FFFF v šestnástkovej. Najväčšie číslo, ktoré sa dá vložiť do bajtu, je v desiatkovej sústave 255, v šestnástkovej FF.*



Väčšina kódov operácií zaberá v strojovom kóde jeden bajt, teda dve šestnástkové číslice. Adresy zaberajú dva bajty, teda dve dvojice šestnástkových číslic.



Prvá dvojica šestnástkových číslic sa nazýva vyšší bajt. Je to vlastne číslo stránky, v ktorej sa nachádza pamäťové miesto (pozri stranu 90). Druhá dvojica číslic sa nazýva nižší bajt a určuje umiestnenie pamäťového miesta v stránke (jedna stránka = 256 pamäťových miest). ZJ spracúva najprv nižší bajt adresy, až potom vyšší, preto ich musíš písať v tomto poradí.

Ako čítať programy v strojovom kóde

Program v strojovom kóde vyzerá na prvý pohľad zložito. Ukážeme si dva spôsoby, ktorými sa takéto programy zapisujú (ani jeden z našich ukážkových programov nie je kompletný a nebežal by na počítači).

Hex výpis

hex adresa — Každá dvojica číslic je jedna inštrukcia, údaj alebo časť adresy.

```
3A30 00 00 00 00 DB 01 CB 1F
3A38 CB 1F 30 07 3E 10 D3 00
3A40 CF 37 3F 21 2F 39 11 00
3A48 D0 01 C0 03 C5 0E 27 71
3A50 1A FE 58 28 28 CD C5 3A
3A58 CD A0 3A D3 01 4E 0D 79
3A60 FE 00 FA 88 3A F5 CD BD
3A68 0B FE 64 CA 06 3B F1 71
```

Takto vyzerá šestnástkový výpis (dump, čítaj damp). Na začiatku každého riadka je adresa, potom nasledujú hodnoty jednotlivých bajtov. Prvý bajt v každom riadku je uložený na adrese, ktorá je na začiatku riadka, ďalšie bajty sú na nasledujúcich adresách.

Výpis z assemblera

adresa — hex kódy — skratky

```
0340  A2 00     LDX #&00
0342  BD 4E 03  LDA &034E,X
0345  9D C0 83  STA &83C0,X
0348  EB        INX
0349  E0 0B     CPX #&0B
034B  D0 F5     BNE &F5
034D  00        BRK
```

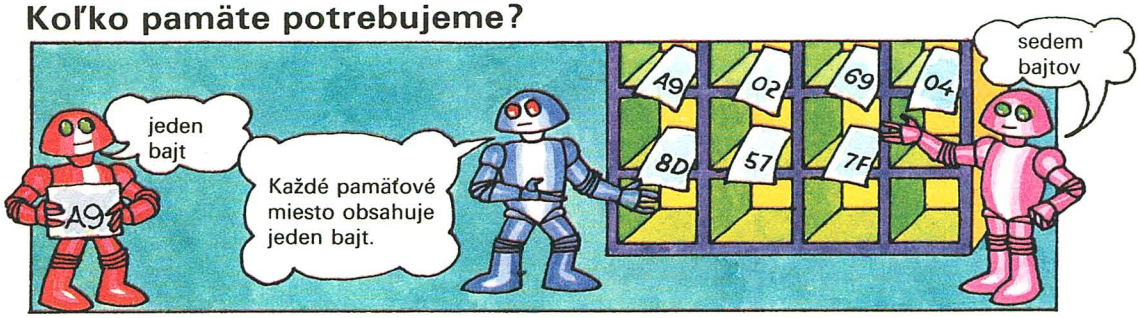
V tomto výpise sú skratky aj šestnástkové kódy. V prvom stĺpci sú adresy, kde sú uložené prvé bajty z riadkov, druhý stĺpec obsahuje šestnástkové kódy inštrukcií, v ďalšom sú skratky.

*O prevode z dvojkovej sústavy do desiatkovej hovoríme na strane 108.

Ako nájsť voľnú RAM

Skôr ako spustíš sčitovací program zo strany 98, musíš urobiť niekoľko krokov. Najprv si musíš vybrať oblasť pamäte, do ktorej program uložíš. Keď píšeš do počítača program v basicu, interpreter ho automaticky ukladá do správnej oblasti — do používateľovej RAM. Keď však píšeš program v strojovom kóde, obchádzaš interpreter, a preto musíš určiť, kam program uložiť. Treba nájsť takú oblasť pamäte, v ktorej tvoj program nebude rušiť ostatné informácie. Nesmieš ho napríklad uložiť do oblasti vyhradenej pre operačný systém alebo zásobník. Ak to urobíš, počítač zrejme prestane správne pracovať, lebo sa mu časť informácií zmaže, prepíše sa na iné. Musíš dať pozor aj na to, aby si mal program v strojovom kóde oddelený od programu v basicu. Keď počítač prestane pracovať, pretože si zle uložil program, väčšinou sa nedá robiť nič iné, len ho vypnúť a znova zapnúť. Tým sa však tvoj program z RAM stratí.

Koľko pamäte potrebujeme?

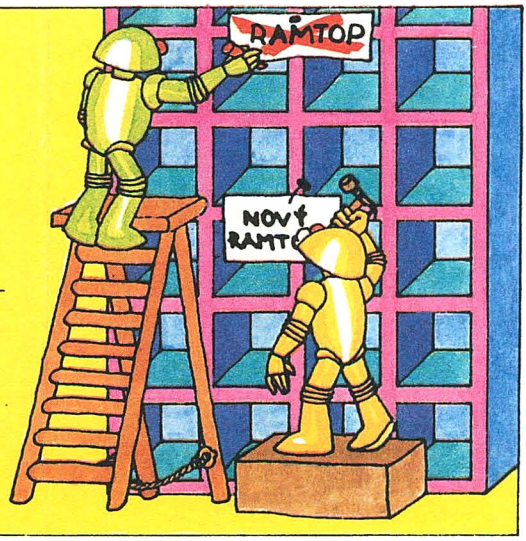


Dĺžku programu v strojovom kóde určíš ľahko — stačí spočítať dvojice šestnástkových cifier, každá znázorňuje jeden bajt. Napríklad náš sčitovací program má sedem bajtov.

Programy v strojovom kóde sú zvyčajne krátke, preto na začiatok stačí nájsť niekoľko sto bajtov voľnej pamäte.

Hľadáme voľnú pamäť

Programy v strojovom kóde sa často umiestňujú na najvyššie adresy v oblasti používateľovej RAM. Musíš však predtým zabezpečiť, aby sa strojový program nezmiešal s programom v basicu. Aby sa tak nestalo, na väčšine počítačov môžeš znížiť adresu konca používateľovej RAM (teda najvyššiu adresu v používateľovej RAM), a tak vytvoriť nad touto oblasťou „územie nikoho". Sem potom uložíš svoj strojový program. Na ďalšej strane uvidíš, ako môžeš zmeniť najvyššiu adresu používateľovej RAM. Táto hodnota sa často nazýva RAMTOP (čítaj ramtop, vrch RAM) alebo MEMEND (čítaj memend, koniec pamäte). Niekedy je to koniec celej pamäte RAM.

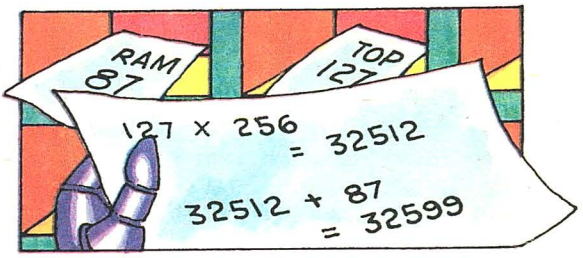


Ako znížiť najvyššiu adresu v používateľovej RAM

Počítač má adresu RAMTOP uloženú niekde v oblasti systémových premenných. Príkazy na jej zmenu sú na rôznych počítačoch rôzne, ale ich princíp býva rovnaký. Viac sa o tom dozvieš vo svojej príručke; možno v nej nájdeš aj jednoduchší spôsob, ako nájsť voľnú pamäť.

Adresa RAMTOP zaberá dva bajty v oblasti systémových premenných: jeden určuje číslo stránky, druhý umiestnenie v stránke. Adresu týchto dvoch miest nájdeš v príručke (najčastejšie pod menom RAMTOP alebo MEMEND). Počítač ukladá dva bajty adresy v obrátenom poradí, najprv na adrese 1 umiestnenie v stránke, až potom na adrese 2 číslo stránky.

```
PRINT PEEK(adresa 1)+
PEEK(adresa 2)*256
```



Ak chceš, aby počítač vypísal adresu RAMTOP, použi takýto príkaz PRINT PEEK. Do zátvoriek doplň adresy systémových premenných svojho počítača. Tento príkaz prevedie dva bajty na jednu adresu vynásobením čísla stránky číslom 256 a pripočítaním umiestnenia v stránke.

```
CLEAR adresa ramtop - 100
                        ZX Spectrum
MEMEND adresa ramtop - 100
                        PP01
```

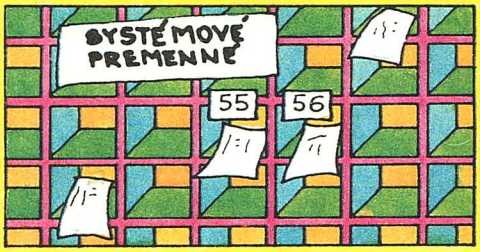
Niektoré počítače majú osobitný príkaz na zmenu hodnoty RAMTOP. Napríklad ZX Spectrum používa príkaz CLEAR, za ktorým musíš napísať novú adresu konca používateľovej RAM, teda starú hodnotu RAMTOP mínus dĺžku tvojho strojového programu. Bližšie informácie nájdeš v príručke.

Príkaz v ukážke zníži koniec používateľovej RAM o 100 pamäťových miest a tie môžeš použiť pre svoj program v strojovom kóde, ktorý sa začína na adrese o jednu väčšej ako nový RAMTOP. Číslo 100 môžeš zmeniť.

* Na nasledujúcej strane nájdeš informácie pre ďalšie počítače.

Tip pre VIC 20

Tento počítač nemá osobitný príkaz na zmenu obsahu systémovej premennej, v ktorej je uložená adresa konca používateľovej RAM. Túto premennú však môžeš zmeniť pomocou príkazov POKE a PEEK.



Adresa, ktorú treba zmeniť, je v pamäťových miestach 55 a 56. Nezabudni, že v druhom bajte je číslo stránky.

```
POKE 56,PEEK(56)-1
```

Takýmto príkazom znížiš adresu konca používateľovej RAM o 256, čiže o jednu stránku. Počítač sa pozrie na adresu 56, od čísla, ktoré tam nájde, odčíta jednotku a výsledok uloží späť na adresu 56. Inými slovami zmenší číslo stránky o jednotku. Ak chceš vidieť novú adresu, napíš:
PRINT PEEK(55)+PEEK(56)*256

Tip pre ZX81

Najlepšie miesto, kam na ZX81 môžeš uložiť program v strojovom kóde, je začiatok používateľovej RAM. Pamäť si vyhraď pomocou príkazu REM hneď na začiatku šestnástkového ukladača. Napíš doň toľko číslic, koľko bajtov potrebuješ vyhradiť pre svoj strojový program.

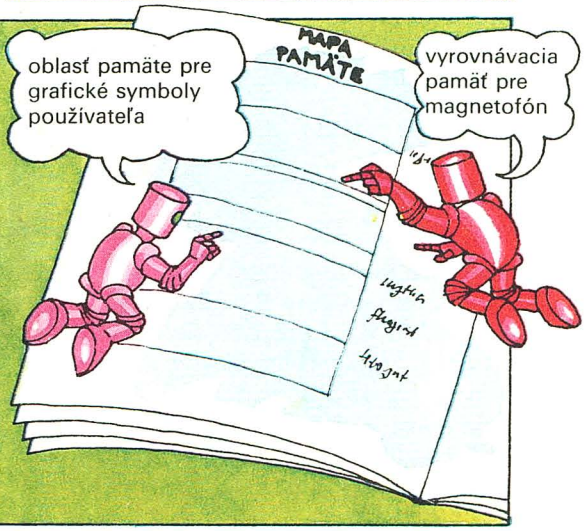
```
5 REM 1234567
```
sedem bajtov

Každá z číslic v príkaze REM zaberie jeden bajt pamäte. Na tieto miesta potom uložíš svoj strojový program. Aby si to mohol urobiť, musíš poznať adresu, na ktorej sa začína príkaz

REM. Používateľova RAM sa začína na adrese 16509. Počítač potrebuje 2 bajty pre číslo riadka, 1 bajt pre REM, 1 pre koniec riadka a 1 na uloženie dĺžky riadka. Prvá číslica v REM je teda na adrese 16514.

Iné miesta pre strojové programy

V príručke k počítaču nájdeš adresy miest, kam môžeš uložiť strojové programy. Na niektorých počítačoch sú to špeciálne adresy vyhradené na tento účel. Napr. na PMD-85-1 a PMD-85-2 je to časť pamäte od 7000 do 77FF (to je v šestnástkovej sústave, v desiatkovej je to od 28672 do 30720); na PP01 je to časť pamäte od 9800 do 9FFF. Na iných počítačoch možno strojový program uložiť napr. do vyrovnávacej pamäte pre magnetofón či do pamäte pre grafické symboly používateľa. Pre strojový program môžeš tieto oblasti pamäte použiť vtedy, ak vieš, že ich nebudeš pri práci s ním potrebovať, teda ak nebudeš volať program z modulu ROM, ukladať ho na kazetu či vytvárať vlastné grafické znaky.



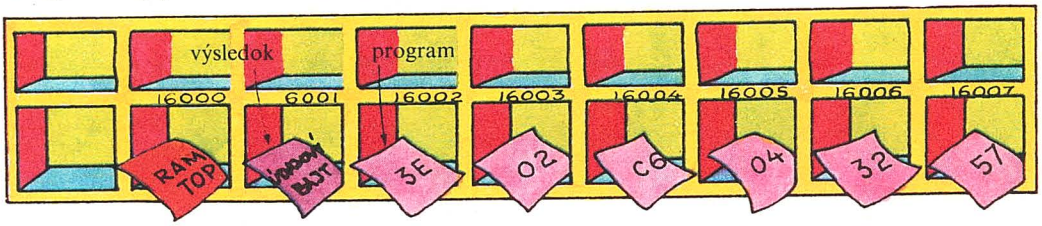
Ako uložiť a spustiť program

Na niekoľkých ďalších stranách sa dozvieš, ako uložiť do pamäte počítača sčitovací program zo strany 98 a ako ho spustiť. Program v strojovom kóde musíš pred spustením uložiť bajt po bajte do oblasti pamäte, ktorú si na to vybral (napr. nad adresou RAMTOP). Na niektorých počítačoch to urobíš ľahko pomocou programu zvaného monitor, ktorý je stále v pamäti ROM. Na iných počítačoch musíš napísať krátky program v basicu — šestnástkový ukladač. Ten preloží každý bajt programu do čísla v desiatkovej sústave a uloží ho príkazom POKE do pamäte. Šestnástkový ukladač je na nasledujúcej strane.

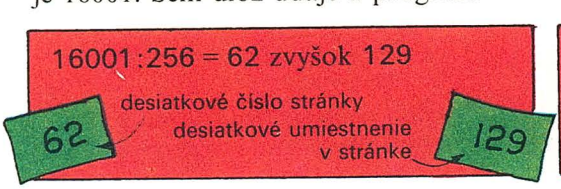
Pred uložením sčitovacieho programu do pamäte musíš najprv určiť adresu, na ktorú sa má uložiť výsledok. Všimni si, že na koniec sčitovacieho programu sme pridali ešte jednu inštrukciu.

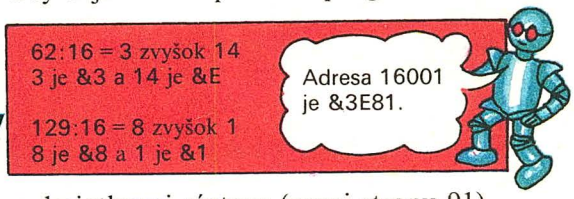
Ako určiť adresu pre výsledok

Údaje, ktoré sú výsledkom práce strojového programu, napríklad súčet v sčitovacom programe, nazývame údajové bajty. Musíš ich uložiť na také miesto v pamäti, kde sa nezmiešajú s programom. Najlepšie miesto je hneď na začiatku oblasti, ktorú si vybral pre program (program sa potom bude začínať až za nimi).



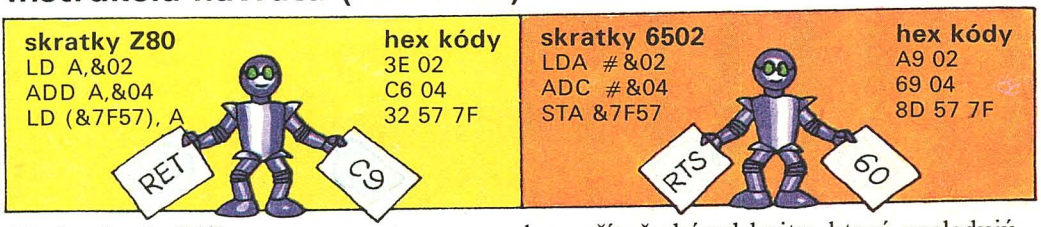
Ak si RAMTOP znížil napríklad na hodnotu 16000, najnižšia voľná adresa je 16001. Sem ulož údaje a program začni na adrese 16002. Adresu údajov musíš previesť do šestnástkovej sústavy, aby si ju mohol použiť v programe.





Adresu prevedieš do šestnástkovej sústavy tak, že ju vydelíš číslom 256. Podiel je číslo stránky a zvyšok je umiestnenie v stránke — zatiaľ v desiatkovej sústave (pozri stranu 91). Prevod dokončíš tak, že obe čísla vydelíš číslom 16 a podiely aj zvyšky prevedieš na šestnástkové cifry.

Inštrukcia návratu (RETURN)



Na koniec každého programu v strojovom kóde musíš pridať inštrukciu RET (pre Z80) alebo RTS (pre 6502), ktorá počítaču prikáže, aby sa po skončení programu vrátil tam, odkiaľ doň skočil. Bez tejto inštrukcie by počítač chápal bajty, ktoré nasledujú v pamäti za tvojím programom, ako inštrukcie a snažil by sa ich vykonať. Väčšinou by sa to skončilo zaseknutím počítača.*

*O inštrukcii RET či RTS hovoríme viac na stranách 116 a 117.

Šestnástkový ukladač

Tento program uloží bajty strojového kódu do pamäte počítača. Musíš ich napísať v šestnástkovej sústave do riadka 160 a za ne pridať slovo KONIEC, aby počítač vedel, kde sa končia. V riadku 40 počítač presunie dve šestnástkové cifry z riadka DATA do premennej H$, v riadkoch 70 až 110 ich prevedie na číslo a v riadku 130 toto číslo uloží do pamäte. Zmeny pre ZX Spectrum nájdeš v poznámke s hviezdičkou, ak máš ZX81 alebo Atari, pozri si stranu 144.

```
10 PRINT "OD AKEJ ADRESY MÁM UKLADAŤ
   STROJOVÝ PROGRAM?"
20 INPUT A
30 LET C=0
40 READ H$
50 IF H$="KONIEC" THEN GOTO 180
60 IF LEN(H$) <> 2 THEN GOTO 170
70 LET X=(ASC(H$)-48)*16
80 IF ASC(H$) > 57 THEN LET X=
   (ASC(H$)-55)*16
90 LET Y=ASC(RIGHT$(H$,1))
100 IF Y < 58 THEN LET X=X+Y-48
110 IF Y > 57 THEN LET X=X+Y-55
120 IF X < 0 OR Y > 255 THEN GOTO 170
130 POKE A+C,X
140 LET C=C+1
150 GOTO 40
155 REM TU SÚ BAJTY, KTORÉ TREBA ULOŽIŤ
160 DATA EF,F6,E2,A9,KONIEC
170 PRINT "ZLÉ ÚDAJE"
180 STOP
```

A je adresa, odkiaľ sa začnú ukladať bajty strojového kódu.

C je počítadlo.

Počítač presunie dve šestnástkové číslice z riadka 160 do premennej H$.

Ak je v H$ slovo KONIEC, znamená to koniec údajov.

Ak H$ neobsahuje dve číslice, prikáže ísť na riadok 170.

Prevedie prvú číslicu zo šestnástkovej sústavy a uloží ju do X.

Prevedie druhú číslicu zo šestnástkovej sústavy, uloží ju do Y a pripočíta k X.

Zisťuje, či sú údaje správne — či je číslo v X v rozpätí od 0 do 255.

Pri prvom prechode je C = 0, teda prvý bajt uloží na adresu A.

Pripočíta 1 k C, čiže ďalší bajt uloží na adresu A + 1.

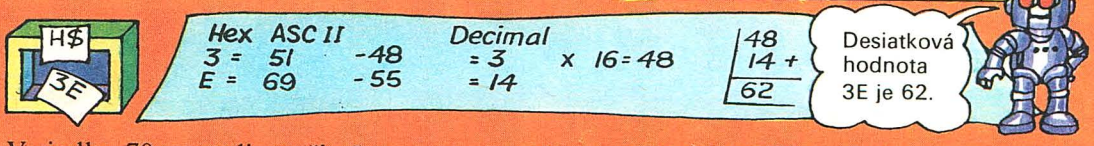
Späť na čítanie ďalšieho kódu.

Sem napíš svoj strojový program v šestnástkovej sústave.

Napíše, ak nájde nesprávne údaje v riadku 60 alebo 120, a zastaví počítač.

Ako ukladač pracuje

hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ASCII	48	49	50	51	52	53	54	55	56	57	65	66	67	68	69	70
	mínus 48										mínus 55					
desiatkovo	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15



V riadku 70 prevedie počítač pomocou funkcie ASC prvú číslicu na jej kód ASCII. Potom prevedie kód ASCII na desiatkovú hodnotu odčítaním 48 alebo 55 a túto hodnotu vynásobí šestnástimi, lebo prvá číslica určuje počet šestnástok. Výsledok uloží do X.

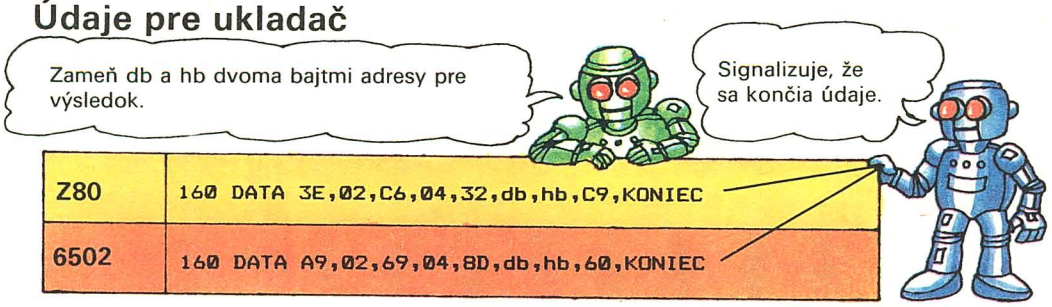
Podobne v riadku 90 prevedie druhú cifru na jej kód ASCII a uloží ho do Y. V riadkoch 100 a 110 kód ASCII zmení na desiatkovú hodnotu odčítaním 48 alebo 55 a túto hodnotu pripočíta k X. Tentokrát netreba násobiť šestnástimi, lebo druhá číslica určuje počet jednotiek. Číslo v premennej X teraz predstavuje desiatkovú hodnotu dvoch šestnástkových cifier v H$.

* Pre ZX Spectrum zmeň ASC na CODE, všetky čísla v riadku 160 daj do úvodzoviek a zmeň riadok 90 na 90 LET Y=CODE(H$(2 TO)).

Ako použiť šestnástkový ukladač

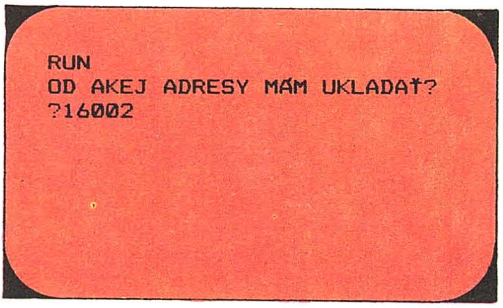
Pomocou šestnástkového ukladača môžeš vyskúšať sčitovací program. Do svojho počítača napíš najprv šestnástkový ukladač. Zameň pritom v riadku 160 naše ukážkové údaje za kódy pre sčitovací program. Sčitovací program nie je nijaký úžasný program, ale je jednoduchý a ukáže ti, ako pracujú programy v strojovom kóde.

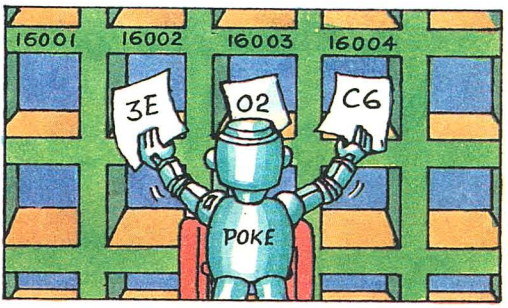
Údaje pre ukladač



V riadkoch DATA sú šestnástkové kódy pre sčitovací program. Písmená db (dolný bajt) a hb (horný bajt) nahraď dvoma bajtmi adresy, na ktorej chceš mať uložený výsledok sčítania. Nezabudni ich napísať v opačnom poradí, teda najskôr nižší bajt (umiestnenie v stránke), až potom vyšší (číslo stránky).

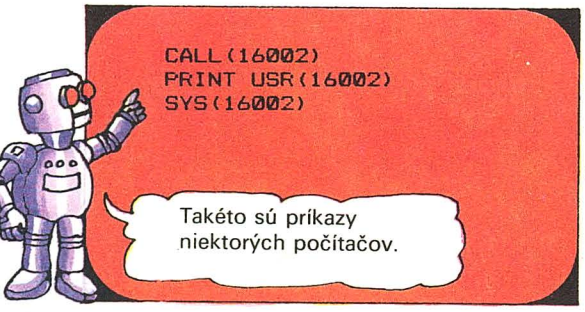
Ako spustiť ukladač

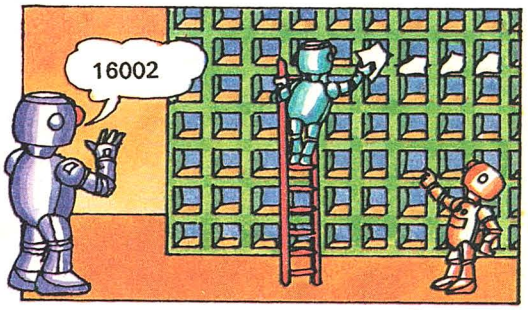




Napíš RUN — tým spustíš ukladač. Keď sa ťa spýta na adresu, napíš adresu prvého pamäťového miesta za bajtmi pre výsledok. Adresu napíš v tvare desiatkového čísla.

Ako spustiť strojový program





Príkaz, ktorým sa spustí strojový program, je na rôznych počítačoch rôzny. Niektoré používajú CALL, iné PRINT USR alebo SYS s desiatkovou adresou pamäťového miesta, na ktorom sa začína program. Zisti si v príručke, ktorý príkaz je vhodný pre tvoj počítač. Takýto príkaz vykoná počítač tak, že skočí na zadanú adresu a začne vykonávať strojové inštrukcie.

Ako si pozrieť výsledok

```
PRINT PEEK(16001)
```

```
PRINT PEEK(16001)
6
```

Počítač vykoná inštrukcie v strojovom kóde a uloží výsledok na miesto, ktoré si mu určil. Ak chceš vidieť výsledok, použi PRINT PEEK s adresou výsledku. Počítač ti ho napíše v desiatkovej sústave.

Nápady na iné programy

Teraz vieš o strojovom kóde dosť na to, aby si mohol vytvárať jednoduché programy. Na konci tejto strany je zhrnutý postup, ktorý pri písaní strojového programu treba dodržať. Riešenia nasledujúcich úloh nájdeš na strane 128.

1. Skús napísať program, ktorý sčíta 25 a 73 (v desiatkovej sústave) a výsledok uloží do pamäte.
2. Dokážeš napísať program, ktorý sčíta 64, 12 a 14 (v desiatkovej sústave) a výsledok uloží do pamäte?

Sčitovací program pracuje správne len do súčtu 255. Na strane 108 sa dozvieš, ako sčítať väčšie čísla.

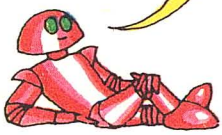
Ako písať program v strojovom kóde

1. Zapíš svoj program pomocou skratiek a všetky údaje preveď do šestnástkovej sústavy.
2. Nájdi kódy pre každú skratku. (Na konci knihy je zoznam skratiek a kódov.)

Nezabudni, že šestnástkový ukladač potrebuje na konci programu slovo KONIEC.

3. Na koniec programu pridaj inštrukciu návratu. (Pozri stranu 103.)
4. Zisti počet bajtov pre program a údaje a vyhraď pamäť RAM. (Pozri strany 100 – 102.)

Zapíš si adresy údajových bajtov, aj adresu, od ktorej sa začína program.



5. Urči adresy bajtov na uloženie údajov a preveď ich do šestnástkovej sústavy. (Pozri stranu 103.)

6. Napíš adresy údajov do programu. Nezabudni zameniť poradie bajtov. (Pozri strany 98 a 99.)

Pred spustením ukladača pozorne skontroluj šestnástkové kódy v riadku DATA.

7. Napíš do počítača šestnástkový ukladač (alebo ho doň prečítaj z pásky) a doplň v ňom šestnástkové kódy

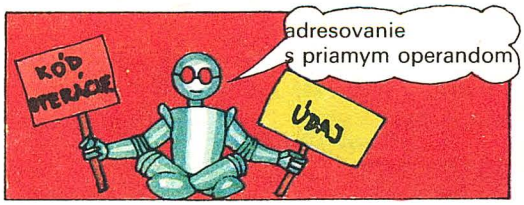
Ak tvoj program nebeží, znovu skontroluj šestnástkové kódy.

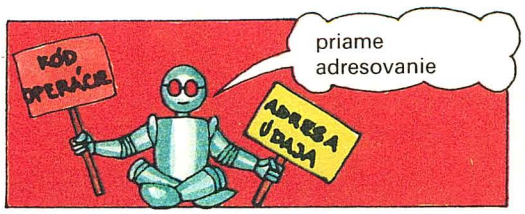
v riadku 160. Nezabudni napísať za ne slovo KONIEC. (Pozri stranu 104.)
8. Spusti ukladač a zadaj mu adresu prvého miesta, kam chceš uložiť strojový program. (Pozri stranu 105.)
9. Spusti strojový program pomocou príkazu pre svoj počítač. V ňom zadáš adresu začiatku programu. (Pozri stranu 105.)

Ak zmeníš údaje v riadku 160, musíš znovu spustiť ukladač, aby uložil do pamäte zmenený strojový program.

Vyberáme z pamäte a ukladáme do nej

Údaje, ktoré má náš sčitovací program sčítať, sú priamo v ňom — v jeho inštrukciách. Tomu hovoríme adresovanie s priamym operandom. Častejšie však potrebujeme povedať počítaču, aby urobil niečo s údajmi, ktoré sú inde v jeho pamäti. Vtedy je v inštrukcii namiesto operandu adresa, na ktorej počítač nájde operand. Tomu hovoríme priame (alebo absolútne) adresovanie.





Toto sú len dva z viacerých spôsobov adresovania, teda spôsobov, ako povedať počítaču, kde nájde údaje, s ktorými má pracovať. Jednotlivé inštrukcie majú odlišné kódy pre každý spôsob adresovania.

Program na sčitovanie čísel z pamäte

Tento program sčíta dve čísla uložené v pamäti. Porovnaj si jeho kódy inštrukcií s predchádzajúcim sčitovacím programom, ktorý používal adresovanie s priamym operandom.

Program pre Z80		
skratky	**hex kódy**	**význam**
LD A,(adresa 1)	3A adresa 1	Prenes číslo z adresy 1 do akumulátora.
LD B,A	47	Prenes číslo z akumulátora do registra B.
LD A,(adresa 2)	3A adresa 2	Prenes číslo z adresy 2 do akumulátora.
ADD A,B	80	Pripočítaj číslo z B k číslu v akumulátore.
LD (adresa 3),A	32 adresa 3	Ulož obsah akumulátora na adresu 3.
RET	C9	Vráť sa.

Ak chceš sčítať dve čísla uložené v pamäti počítača, musíš ich najprv dostať do registrov — napríklad do A a B. Keďže údaj nemožno preniesť z pamäte rovno do registra B, musíš ho najprv preniesť do registra A a odtiaľ do B.

Program pre 6502		
skratky	**hex kódy**	**význam**
LDA adresa 1	AD adresa 1	Prenes číslo z adresy 1 do akumulátora.
ADC adresa 2	6D adresa 2	Pripočítaj číslo z adresy 2 k akumulátoru.
STA adresa 3	8D adresa 3	Ulož obsah akumulátora na adresu 3.
RTS	60	Vráť sa.

Ako spustiť program

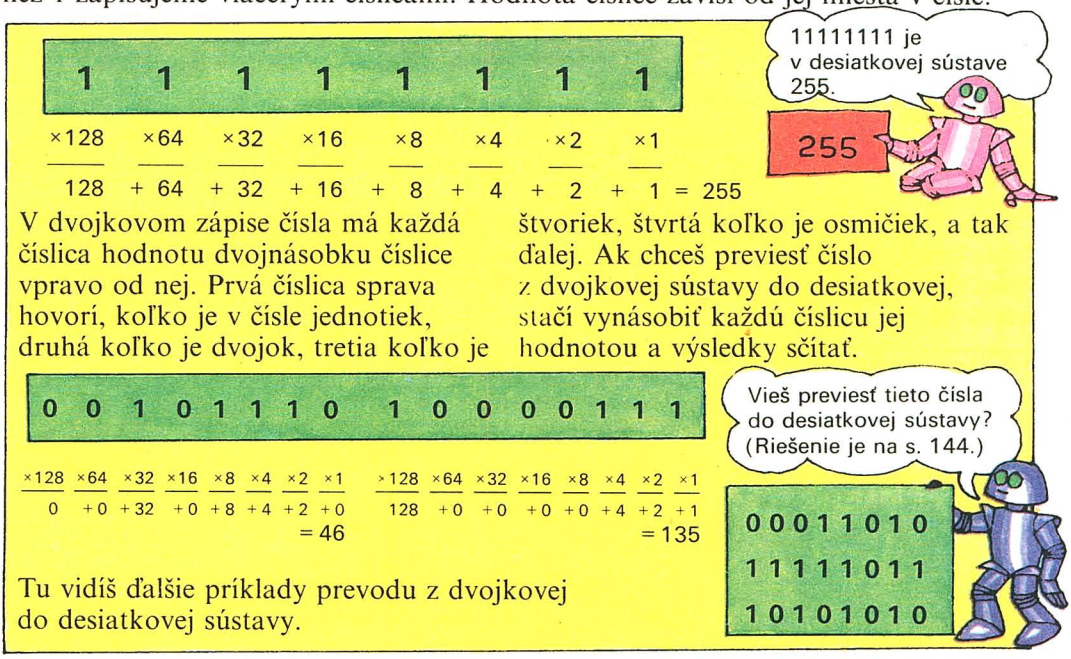
Pri spúšťaní programu postupuj podľa bodov na vedľajšej strane. Predtým však musíš uložiť do pamäte čísla, ktoré chceš sčítať. Aby sa ti nepomiešali s inštrukciami, vyhraď pre ne pamäťové miesta pred strojovým programom. Preveď adresy týchto miest do šestnástkovej sústavy a vlož ich do programu. Na uloženie výsledku potrebuješ tretiu adresu. Výsledok si pozrieš pomocou PRINT PEEK(adresa 3).

Pracujeme s veľkými číslami

Sčitovacie programy na predchádzajúcich stranách pracujú správne, len kým súčet neprevýši 255. To je najväčšie číslo, ktoré môže byť uložené v osembitovom registri alebo v pamäťovom mieste. Ak chceš pracovať s väčšími číslami, musíš vedieť viac o dvojkovej sústave a o podmienke prenosu (bite C). Na strane 110 nájdeš program na sčítanie veľkých čísel.

Dvojkové čísla

Dvojková sústava je založená na podobnom princípe ako desiatková alebo šestnástková, čísla sa však vyjadrujú len pomocou 2 číslic — 0 a 1. Čísla väčšie než 1 zapisujeme viacerými číslicami. Hodnota číslice závisí od jej miesta v čísle.

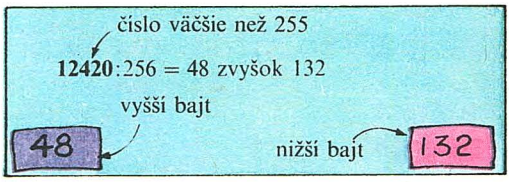


V dvojkovom zápise čísla má každá číslica hodnotu dvojnásobku číslice vpravo od nej. Prvá číslica sprava hovorí, koľko je v čísle jednotiek, druhá koľko je dvojok, tretia koľko je štvoriek, štvrtá koľko je osmičiek, a tak ďalej. Ak chceš previesť číslo z dvojkovej sústavy do desiatkovej, stačí vynásobiť každú číslicu jej hodnotou a výsledky sčítať.

Tu vidíš ďalšie príklady prevodu z dvojkovej do desiatkovej sústavy.

Ako zadať počítaču veľké čísla

V počítači sú veľké čísla uložené v dvoch bajtoch, podobne ako adresy. Aj tu hovoríme o nižšom a vyššom bajte. Vyšší bajt nám povie, koľkokrát sa 256 nachádza v čísle, a nižší obsahuje zvyšnú časť čísla, teda zvyšok po delení číslom 256. Podobne ako pri adresách, počítač ukladá do pamäte najprv nižší bajt, až potom vyšší.



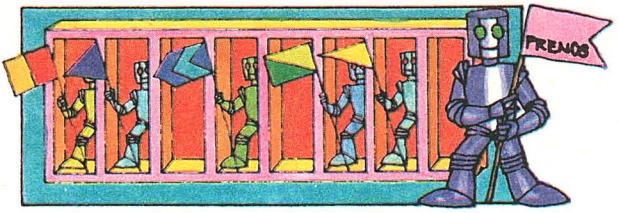
Ak chceš počítaču zadať číslo väčšie než 255, musíš poznať oba jeho bajty. Zistíš ich tak, že zadané číslo vydelíš číslom 256. Výsledok je hodnota vyššieho bajtu, zvyšok hodnota nižšieho. Obe hodnoty budú vyjadrené v desiatkovej sústave. Ak chceš použiť takéto číslo ako údaj v strojovom programe, musíš previesť každý bajt do šestnástkovej sústavy, čiže čísla vydeliť číslom 16 a podiel aj zvyšok previesť na šestnástkové cifry. (Hovoríme o tom na s. 91.)

307; 21214; 759; 1023.

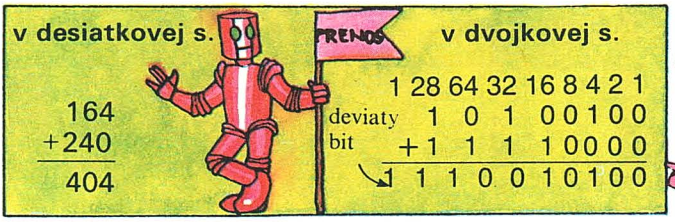
Aké sú vyššie a nižšie bajty týchto čísel v desiatkovej a šestnástkovej sústave? (Riešenie je na s. 144.)

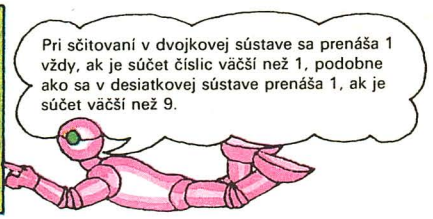
Podmienka prenosu

Podmienka prenosu (v angličtine carry flag, čítaj käri fleg, vlajka prenosu) je bit C v registri podmienok (v 6502 v stavovom registri procesora). Nastaví sa na 1 vždy, ak je výsledok sčítania väčší než 255 a nezmestí sa do ôsmich bitov registra. Ak má podmienka hodnotu 1, hovoríme, že je nastavená; ak má hodnotu 0, hovoríme, že je nulová.



Bit C môžeš považovať za deviaty bit výsledku sčítania. Pozri si ukážku 164 + 240 (v dvojkovej sústave 10100100+11110000).

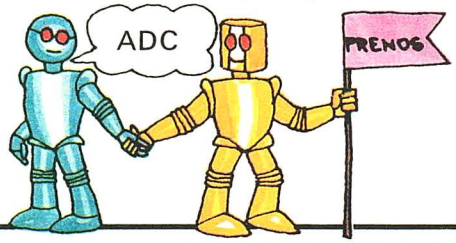


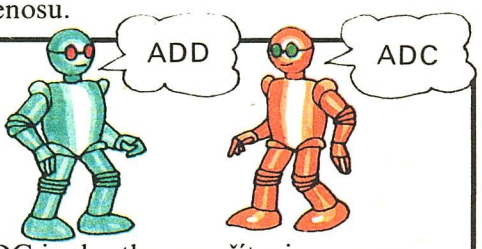


Výsledok tohto sčítania je 404. Výsledné číslo zaberie v dvojkovej sústave 9 bitov. Deviaty bit hovorí, koľkokrát sa 256 nachádza v čísle, a v počítači bude uložený v podmienke prenosu.

Prenos v Z80

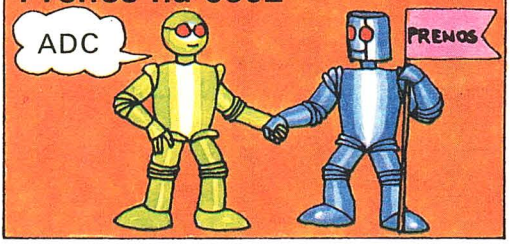
Z80 má dve rôzne inštrukcie pre sčítanie: ADD a ADC. ADD prikáže počítaču sčítať dve čísla bez toho, aby zarátal prenos z predchádzajúceho výpočtu. Ak sa pri sčítaní prenáša 1 do deviateho bitu, podmienka prenosu sa nastaví, inak sa vynuluje.





ADC je skratka pre sčítanie s prenosom (add with carry, čítaj edd uid käri). Prikazuje sčítať dve čísla a podmienku prenosu a podľa výsledku nastaviť podmienku prenosu. Ak sčituješ veľké čísla, najprv použi na sčítanie najnižších bajtov ADD a potom ADC na sčítanie vyšších bajtov so zarátaním prenosu z predchádzajúceho sčítania.

Prenos na 6502



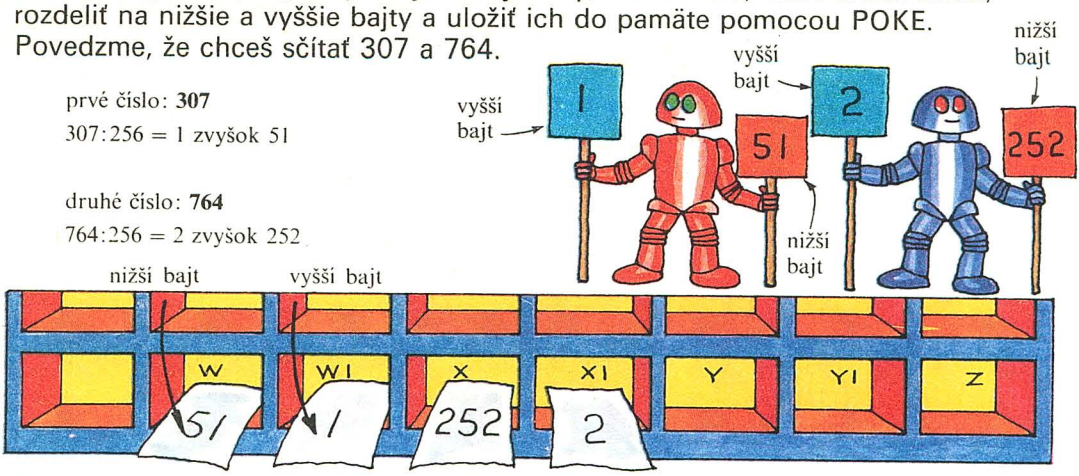
6502 má len jednu inštrukciu na sčítanie — ADC, prenos z predchádzajúcich výpočtov sa teda vždy pripočítava k výsledku. Preto je dôležité použiť pred sčítaním inštrukciu CLC (clear carry, čítaj klír käri, zmaž prenos), ktorá vynuluje podmienku.

Sčítanie veľkých čísel

Skôr ako spustíš programy z tejto dvojstrany, musíš čísla, ktoré chceš sčítať, rozdeliť na nižšie a vyššie bajty a uložiť ich do pamäte pomocou POKE. Povedzme, že chceš sčítať 307 a 764.

prvé číslo: **307**
307:256 = 1 zvyšok 51

druhé číslo: **764**
764:256 = 2 zvyšok 252



Tieto čísla musíš uložiť do pamäťových miest na začiatku oblasti pre strojový program. Nižší bajt musí byť vždy na nižšej adrese, za ním nasleduje vyšší bajt. Na našom obrázku sú dva bajty prvého čísla uložené v pamäťových miestach W a W1, bajty druhého čísla v X a X1. Na uloženie výsledku potrebuješ tri miesta: Y, Y1 a Z, teda miesta pre nižší a vyšší bajt a prípadný prenos.

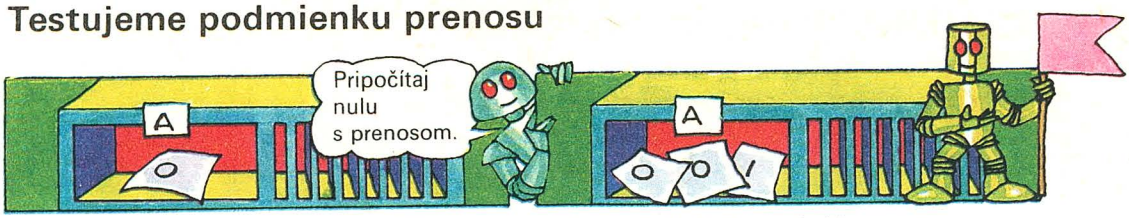
Program pre Z80

Dvojbajtové čísla sa na Z80 sčitujú pomerne ľahko, lebo registre môžeš použiť v pároch, a to tak, že každý pár obsahuje dva bajty čísla. Prvé číslo uložíme do registrového páru HL, druhé do páru BC. Registrový pár HL nám poslúži ako akumulátor pre veľké čísla. Pri čítaní programu ti iste pomôže obrázok na začiatku tejto strany.

skratky	hex kódy	význam
LD HL,(adresa W)	2A adresa W	Prenes číslo z adresy W (nižší bajt prvého čísla) do registra L a číslo z adresy W1 (vyšší bajt prvého čísla) do registra H.
LD BC,(adresa X)	ED 4B adresa X Tento kód operácie má dva bajty.	Prenes číslo z adresy X (nižší bajt druhého čísla) do registra C a číslo z adresy X1 (vyšší bajt druhého čísla) do registra B.
ADD HL,BC	09	Sčítaj obsah HL a BC a výsledok nechaj v HL. Nepripočítavaj prenos, ale ho nastav, ak sa výsledok nezmestí do dvoch bajtov.
LD (adresa Y),HL	22 adresa Y	Ulož nižší bajt výsledku na adresu Y a vyšší bajt na adresu Y1.
LD A,&0 ADC A,&0 LD (adresa Z),A RET	3E 00 CE 00 32 adresa Z C9	Na nasledujúcej strane uvidíš, ako počítač pracuje s podmienkou prenosu. Návrat.

Do tohto programu musíš pred spustením doplniť adresy pre W, X, Y a Z (nezabudni na správne poradie bajtov — nižší, vyšší). Ak používaš páry registrov, stačí zadať jednu adresu pre každý pár. Do vyššieho registra v páre počítač automaticky uloží číslo z nasledujúceho pamäťového miesta.

Testujeme podmienku prenosu

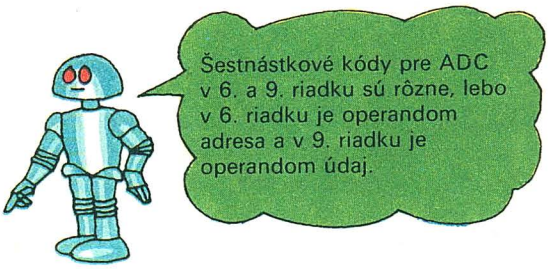


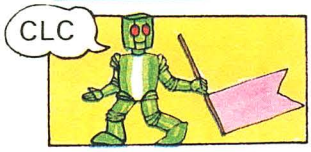
Riadky 5 až 7 v programe pre Z80 testujú podmienku prenosu. Obsah podmienky nemôžeš priamo preniesť do registra ani do pamäte. Jej obsah sa najjednoduchšie dozvieš pomocou ďalšieho sčítania. Do registra A sa najprv uloží 0 (5. riadok) a potom sa pomocou inštrukcie ADC pripočíta 0 (6. riadok). Ak bola po predchádzajúcom sčítaní podmienka prenosu nastavená, akumulátor bude po sčítaní obsahovať jednotku a tá sa uloží na adresu Z (7. riadok).

Program pre 6502

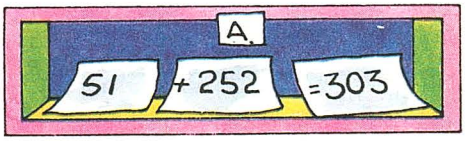
Takto sčítaš dve čísla väčšie než 255 na 6502. Skôr ako program spustíš, musíš čísla rozdeliť na vyššie a nižšie bajty tak, ako to vidíš na vedľajšej strane.

skratky	hex kódy
CLC	18
LDA adresa W	AD adresa W
ADC adresa X	6D adresa X
STA adresa Y	8D adresa Y
LDA adresa W1	AD adresa W1
ADC adresa X1	6D adresa X1
STA adresa Y1	8D adresa Y1
LDA #&0	A9 00
ADC #&0	69 00
STA adresa Z	8D adresa Z
RTS	60

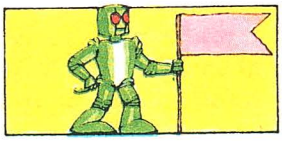




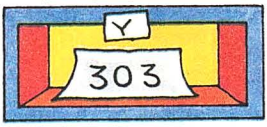
Najprv sa vynuluje prenos, ak by bol nastavený po predchádzajúcej operácii.

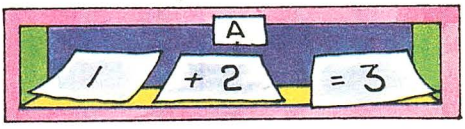


Potom sa uloží nižší bajt prvého čísla do akumulátora a pripočíta sa k nemu s prenosom nižší bajt druhého čísla (2. a 3. riadok).

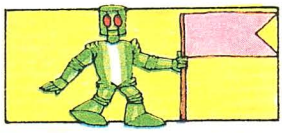


Ak je výsledok väčší než 255, nastaví sa prenos.





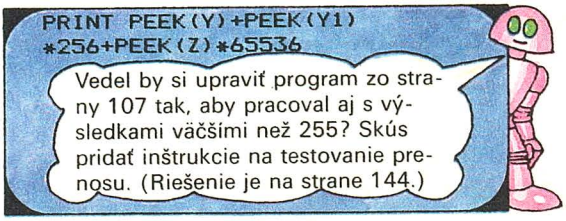
Výsledok sa uloží na adresu Y (4. riadok). Potom sa sčítajú 2 vyššie bajty a prenos (ak nejaký bol) z predchádzajúceho súčtu. Výsledok sa uloží na adresu Y1 (7. riadok).



Riadky 8 – 10 testujú podmienku prenosu rovnako ako na Z80.

Ako si pozrieť výsledok

Výsledok je uložený v troch bajtoch. Nižší bajt určuje počet jednotiek, vyšší počet 256-iek a prenos tentokrát určuje počet 65536-iek (256 × 256). Na zobrazenie výsledku použi príkaz z obrazovky vpravo. Nahraď v ňom Y, Y1 a Z adresami pre svoj počítač.

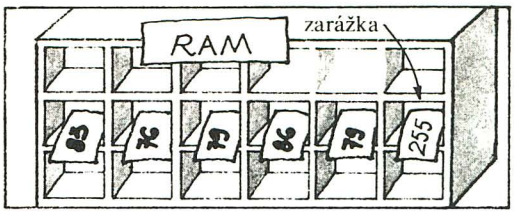


Píšeme na obrazovku

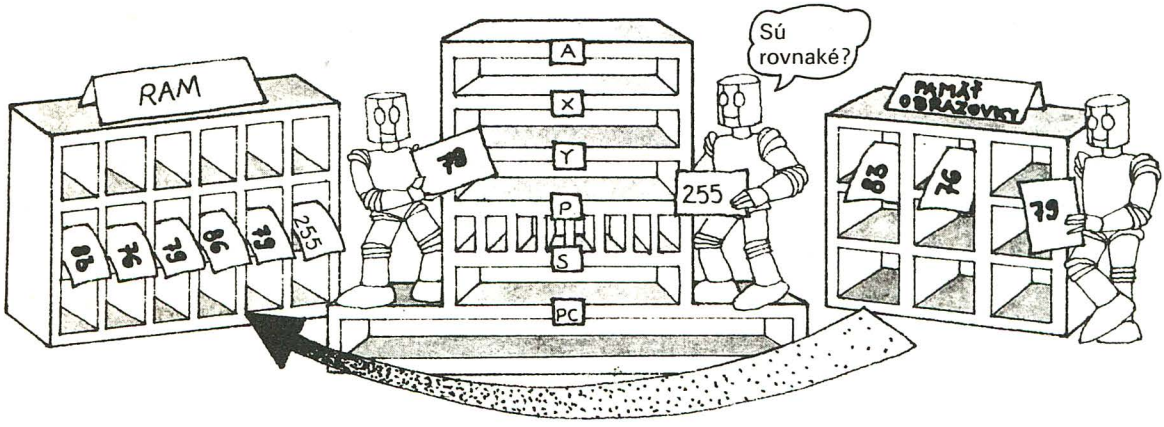
Náš ďalší program ti ukáže, ako môžeš zobraziť nejakú správu na obrazovke. Program pre Z80 je na stranách 113 a 114 a program pre 6502 je na strane 115. Pracujú podobne, aj keď pre každý z mikroprocesorov je použitá trocha iná metóda.

Programy na týchto stranách sú len pre počítače, ktoré majú rôzny textový a grafický režim (napr. Atari 130XE), alebo ktoré nevedia kresliť. Na väčšine počítačov sa objavia na obrazovke len vzorky z rôzne rozsvietených bodov. Na nich treba písmená zobrazovať pomocou volania podprogramu v ROM.

Ako pracuje program

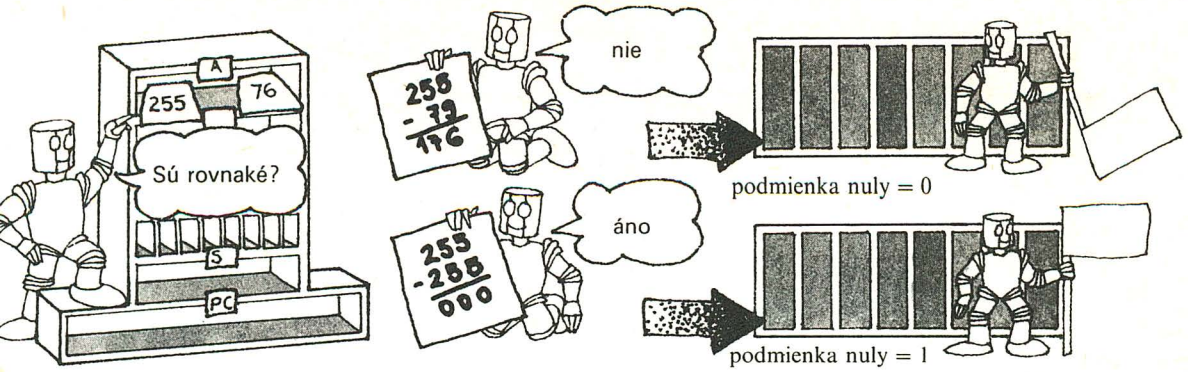


Najprv uložíš kódy všetkých písmen správy do pamäťových miest na začiatku oblasti pre strojový kód. Každé písmeno zaberie jeden bajt. Na konci správy ulož ešte bajt obsahujúci číslo 255. Ten bude pre počítač zarážkou — označuje koniec správy.



Program prikazuje počítaču postupne prenášať jednotlivé bajty do akumulátora a porovnávať ich s číslom 255. Ak sa bajt nerovná 255, počítač ho uloží do pamäte obrazovky a písmeno automaticky zobrazí. Potom počítač skočí na začiatok programu a vyberie z pamäte ďalší bajt správy.

Porovnávanie



Na porovnanie bajtu s obsahom akumulátora použi inštrukciu CP (pre Z80) alebo CMP (pre 6502). Počítač od čísla v akumulátore odčíta číslo v porovnávanom bajte (spraví to však len „v mysli", obsah akumulátora nezmení). Ak sú čísla rovnaké, výsledkom je 0 a počítač nastaví podmienku nuly (bit Z). Ak nie sú rovnaké, podmienku nuly vynuluje. Po porovnaní môžeš počítaču prikázať, aby prešiel na vykonávanie inej časti programu alebo pokračoval vo vykonávaní ďalšej inštrukcie podľa toho, či je podmienka nuly nastavená alebo nie.

Program pre Z80

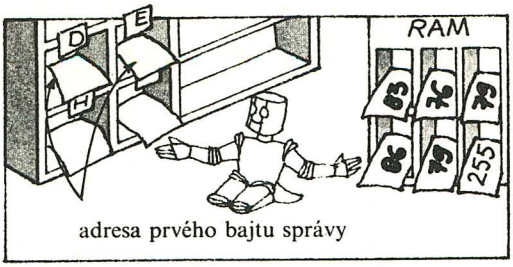
Skôr ako spustíš program, ulož do RAM svoju správu. Potom doplň adresy v 1. a 2. riadku. Posledná inštrukcia programu hovorí počítaču, aby sa vrátil na 3. inštrukciu. Do poslednej inštrukcie musíš teda vložiť adresu, na ktorej sa v tvojom počítači nachádza 3. inštrukcia.

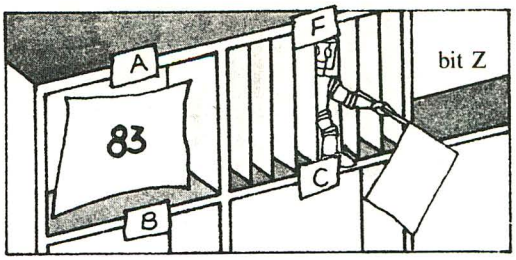
skratky	hex kódy
LD HL,adresa obrazovky	21 adresa obrazovky
LD DE,adresa správy	11 adresa správy
LD A,(DE)	1A
CP &FF	FE FF
RET Z	C8
LD (HL),A	77
INC DE	13
INC HL	23
JP adresa tretej inštrukcie	C3 adresa tretej inštrukcie

V tomto programe používame registrové páry HL a DE na uloženie adries, z ktorých počítač vyberá údaje alebo na ktoré ich ukladá. Takéto použitie registrov nazývame nepriame adresovanie. Inštrukcie v 3. a 6. riadku používajú nepriame adresovanie.

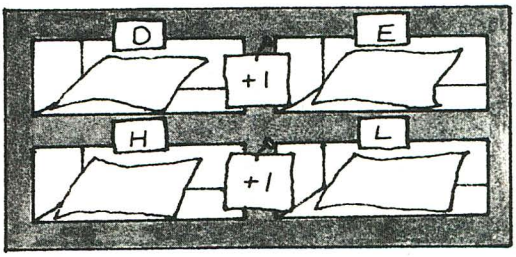
PROGRAM POKRAČUJE NA STRANE 114.

V prvých dvoch inštrukciách si počítač uloží adresu obrazovky (adresu, kam treba ukladať údaje) do registrového páru HL a adresu správy (adresu, z ktorej sa vyberajú údaje) do registrového páru DE.





LD A,(DE) hovorí počítaču, aby sa pozrel do pamäte na tú adresu, ktorá je uložená v DE, a vybral z nej bajt do akumulátora. To je nepriame adresovanie. Potom počítač porovná bajt v akumulátore s &FF (šestnástkový zápis čísla 255). RET Z prikáže počítaču vrátiť sa do basicu, ak je nastavená podmienka nuly (bit Z), teda ak sa bajt rovná 255. Ak je bit Z nulový, počítač pokračuje vo vykonávaní ďalších inštrukcií.



Aj inštrukcia LD (HL),A používa nepriame adresovanie. Prikazuje počítaču pozrieť si adresu v HL a uložiť do pamäte na túto adresu bajt z registra A, čiže bajt správy. INC je inštrukcia pre pripočítanie jednotky.

V 7. a 8. riadku teda počítač pripočíta jednotku k adresám v registroch DE a HL. Po návrate na 3. inštrukciu bude pracovať s ďalším bajtom správy a bude ho ukladať na ďalšie miesto v pamäti obrazovky.

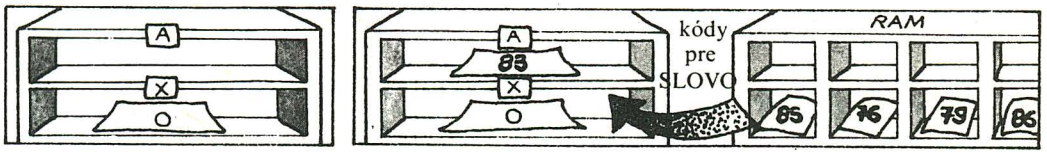
Program pre 6502

Tu je program, ktorý napíše správu pre 6502. Skôr ako ho spustíš, musíš do RAM uložiť kódy písmen správy a za ne kód 255. Adresu začiatku textu doplň do 2. riadka programu. V 5. riadku doplň adresu, na ktorej sa v tvojom počítači začína pamäť obrazovky.

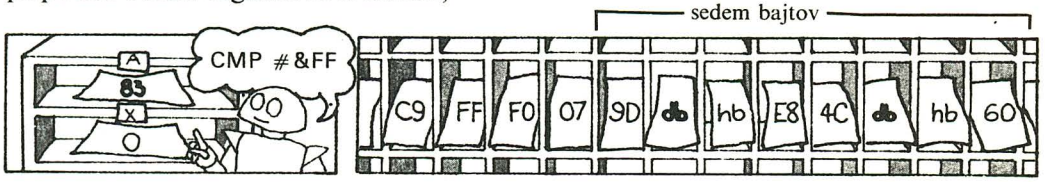
Do 7. riadka ešte musíš doplniť adresu, na ktorej bude v tvojom počítači uložená 2. inštrukcia programu. Tak zabezpečíš opakovanie programu pre všetky písmená správy.

skratky	hex kódy
LDX #&00	A2 00
LDA adresa správy,X	BD adresa správy
CMP #&FF	C9 FF
BEQ na inštrukciu RTS	F0 07
STA adresa obrazovky,X	9D adresa obrazovky
INX	E8
JMP adresa druhej inštrukcie	4C adresa druhej inštrukcie
RTS	60

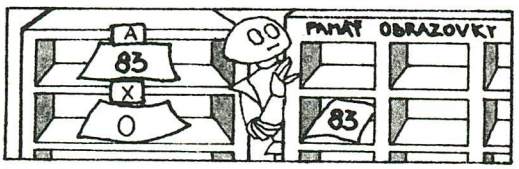
Tento program používa iný spôsob adresovania — nazýva sa indexované adresovanie. Adresa údaja sa vypočíta tak, že k adrese, ktorá je operandom inštrukcie, sa pripočíta obsah registra X alebo Y. Indexované adresovanie je v 2. a v 5. riadku.

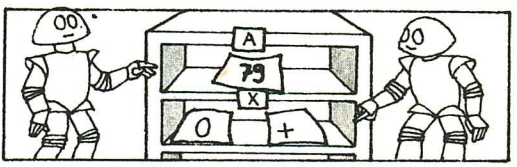


V 1. riadku uloží počítač do registra X nulu. V 2. riadku sa používa indexované adresovanie, čiže počítač pripočíta obsah registra X k adrese, ktorá je v inštrukcii. Výsledok je adresa údaja (bajtu tvojej správy), ktorý treba preniesť do akumulátora.



CMP v 3. riadku počítaču prikazuje porovnať bajt v akumulátore s hodnotou &FF (šestnástkový zápis čísla 255). Tento kód používame ako koncovú značku. Ak sú rovnaké, počítač nastaví bit Z. Ďalšia inštrukcia, BEQ (branch if equal, čítaj brenč if íkvl, skoč, ak sa rovnajú), hovorí počítaču, aby v prípade, že je nastavený bit Z, preskočil niekoľko inštrukcií. Koľko bajtov treba pri tom preskočiť, sa počítač dozvie z bajtu, ktorý nasleduje v pamäti za kódom inštrukcie BEQ. Keďže chceme, aby v prípade rovnosti počítač skočil na inštrukciu RTS, je v programe na tomto mieste číslo 7 — medzi BEQ a RTS je totiž sedem bajtov.





5. riadok programu prikazuje počítaču uložiť bajt z akumulátora na adresu, ktorú vypočíta sčítaním adresy v inštrukcii a obsahu registra X. Zase teda používame indexované adresovanie.

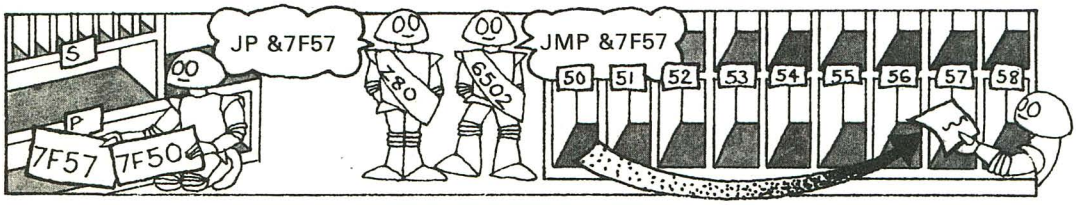
INX (skratka od increment X, čítaj inkrement X, zvýš X) prikáže počítaču pripočítať jednotku k obsahu registra X. Potom počítač skočí späť na 2. inštrukciu už s novou hodnotou X, čiže bude spracúvať ďalší bajt správy.

Skoky a vetvenie

O vetvení hovoríme vtedy, ak prikážeš počítaču prejsť na inštrukciu v inej časti programu — teda nie na tú, ktorá nasleduje. Poznáme tri spôsoby vetvenia: nepodmienený skok, vyvolanie podprogramu a podmienený skok. Nepodmienený skok prikazuje jednoducho skočiť na zadanú adresu. Pri podmienenom skoku počítač podľa výsledku predchádzajúcich operácií (presnejšie podľa nastavenia podmienok) buď skočí na inú inštrukciu, alebo pokračuje vo vykonávaní nasledujúcej inštrukcie. O podmienených skokoch píšeme na nasledujúcej strane.

Počítadlo adries

Počítadlo adries (register PC) je zvláštny šestnásťbitový register — je v ňom uložená adresa inštrukcie, ktorú má počítač vykonať. Keď počítač hľadá inštrukciu, pozrie sa do pamäte na adresu v registri PC. Potom sa obsah PC zväčší o jednotku, aby ukazoval na ďalšiu inštrukciu.



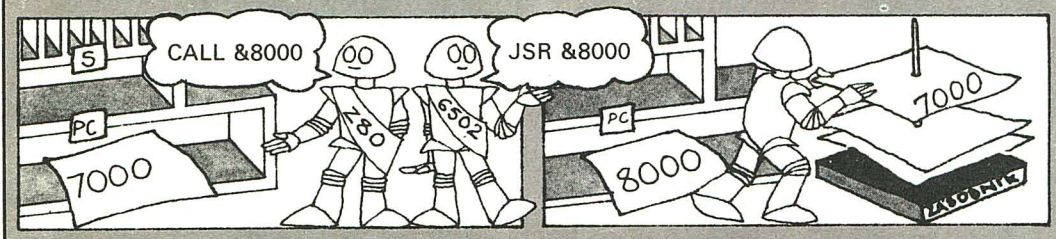
skok

Ak prikážeš počítaču skočiť na nejakú adresu, uloží si túto adresu do PC. A tak sa nasledujúca inštrukcia bude vyberať z pamäte z tejto adresy.

Skratky pre nepodmienený skok pre Z80 a 6502 vidíš na obrázku.

Podprogramy

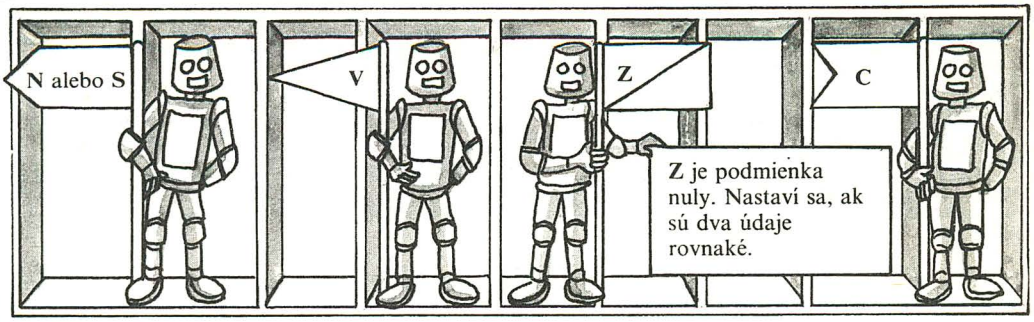
Skočiť do podprogramu prikážeš inštrukciou CALL adresa (pre Z80) alebo JSR adresa (pre 6502). CALL (čítaj kól) znamená volaj. JSR je skratka od jump to subroutine (čítaj džamp tu sabrutyn) a znamená skoč do podprogramu. Podprogramy v strojovom kóde sú podobné ako v basicu, musia mať na konci inštrukciu návratu (RET pre Z80 a RTS pre 6502).



Ak prikážeš počítaču skočiť na začiatok podprogramu, čiže vyvolať podprogram, počítač uloží adresu podprogramu do registra PC. Starý obsah PC, teda adresu inštrukcie, kam sa má po skončení podprogramu vrátiť, si však predtým uloží na vrchol strojového zásobníka. (Zásobník je zvláštna časť RAM — viac je o ňom na strane 90).

Podmienené skoky

V podmienenom skoku počítač najprv zistí stav niektorej podmienky v registri podmienok a potom podľa jej stavu buď vykonáva nasledujúcu inštrukciu, alebo skočí na iné miesto v programe. V podmienených skokoch môžeš testovať tieto podmienky:

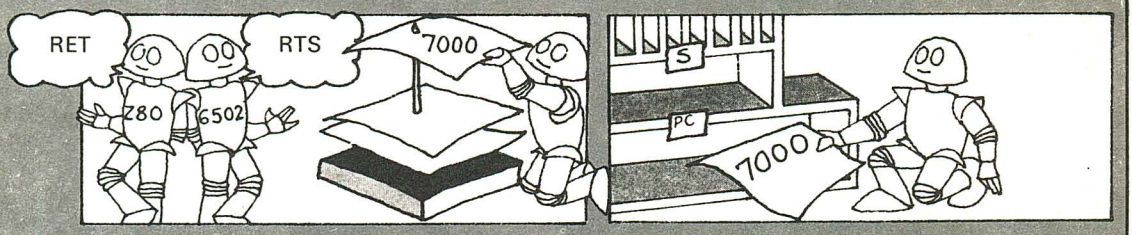


N alebo S je podmienka znamienka (pre Z80 používame označenie S, pre 6502 N). Podmienka sa nastaví, ak je výsledok operácie záporný. Ak je kladný, podmienka S bude nulová.

V alebo P/V. Táto podmienka sa na 6502 volá podmienka pretečenia. V procesore Z80 má dvojakú funkciu a nazýva sa podmienka parity (párnosti) a pretečenia. Ako podmienka pretečenia je nastavená vtedy, ak pri výpočte v dvojkovom doplnku (pozri stranu 119) nastal prenos do znamienkového bitu, a ako podmienka parity je nastavená vtedy, ak je vo výsledku nepárny počet jednotiek.

C je podmienka prenosu a nastavená je vtedy, ak sa výsledok nezmestí do jedného bajtu.

Na nastavenie podmienok pred podmieneným skokom sa najčastejšie používa inštrukcia CMP. Aj iné inštrukcie však nastavujú alebo nulujú podmienky, napríklad inštrukcia DEC na 6502 mení podmienku znamienka a podmienku nuly.*



Ak počítač nájde na konci podprogramu RET alebo RTS, vyberie adresu, ktorá je na vrchole zásobníka a uloží ju do registra PC. Počítač teda ďalej vykonáva inštrukcie, ktoré nasledujú za inštrukciou vyvolania podprogramu. Inštrukcie RET alebo RTS slúžia aj na návrat zo strojového programu do basicu — interpreter basicu tvoj strojový program totiž vyvolá ako podprogram.

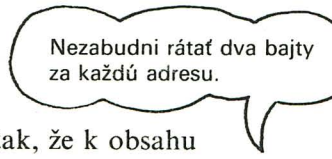
* V zozname inštrukcií svojho počítača nájdeš aj informáciu o tom, ktorá inštrukcia mení obsah ktorých podmienok.

Inštrukcie pre podmienené skoky

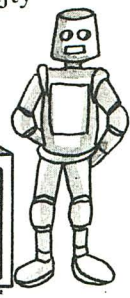
Tu nájdeš skratky pre podmienené skoky podľa ľubovoľnej podmienky.

Z80	Skoč, ak...	6502	Skoč, ak...
JP C	je prenos (C = 1)	BCS	je prenos (C = 1)
JP NC	nie je prenos (C = 0)	BCC	nie je prenos (C = 0)
JP Z	je rovnosť (Z = 1)	BEQ	je rovnosť (Z = 1)
JP NZ	je nerovnosť (Z = 0)	BNE	je nerovnosť (Z = 0)
JP M	je záporný výsledok (S = 1)	BMI	je záporný výsledok (S = 1)
JP P	nie je záporný výsledok (S = 0)	BPL	nie je záporný výsledok (S = 0)
JP PO	je nepárna parita (P/V = 1)	BVS	je pretečenie (V = 1)
JP PE	je párna parita (P/V = 0)	BVC	nie je pretečenie (V = 0)

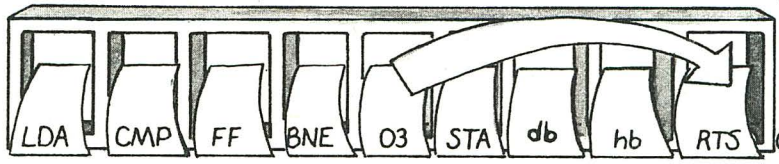
Ako vypočítať posunutie



Podmienený skok s relatívnym adresovaním vykonáva počítač tak, že k obsahu počítadla adries PC pripočíta posunutie. Posunutie vypočítaš tak, že spočítaš bajty od skoku po inštrukciu, na ktorú chceš skočiť (vrátane nej). Rátať začni od inštrukcie, ktorá nasleduje za skokom, a počítaj ju ako nultú (PC totiž bude pri vykonávaní skoku ukazovať práve sem). Tento postup ti objasnia dva krátke programy pre 6502. (Pre Z80 je postup rovnaký.)



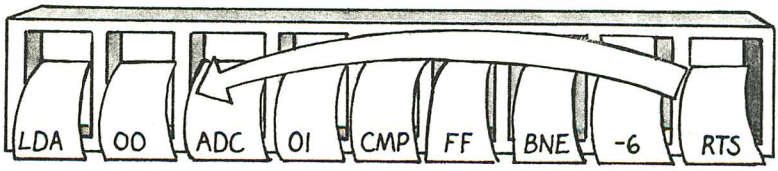
```
LDA adresa
CMP #&FF
BNE na RTS
STA adresa
RTS
```



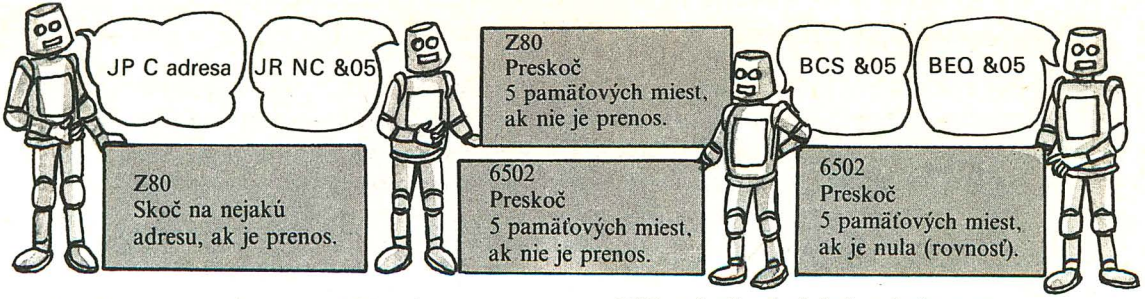
Aby v programe na hornom obrázku počítač skočil na inštrukciu RTS, musíš napísať posunutie 3.

V programe dolu sa posunutie, ktoré pošle počítač späť na inštrukciu ADC, rovná −6.

```
LDA #&00
ADC #&01
CMP #&FF
BNE na ADC
RTS
```



Táto inštrukcia je nultá.



Po JP test napíšeš pre Z80 adresu, na ktorú má počítač skočiť. Pre 6502 musíš zadať počet bajtov, ktoré má počítač preskočiť smerom dopredu alebo dozadu, aby sa dostal na žiadanú inštrukciu. Toto nazývame relatívne adresovanie a číslo nazývame posunutie.

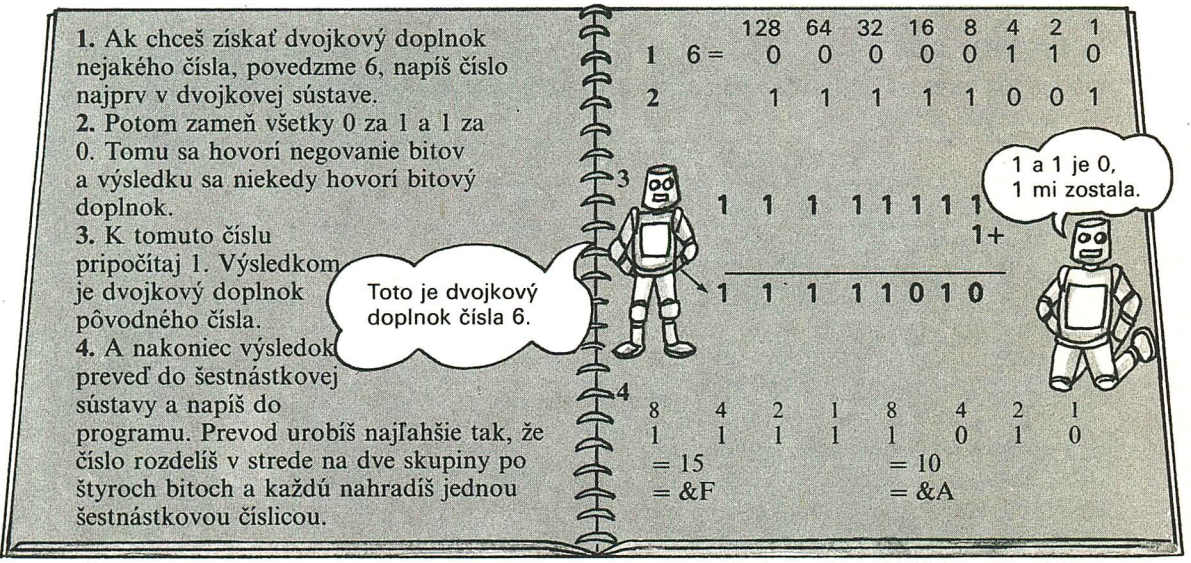
Z80 má ešte inú inštrukciu pre podmienený skok — JR test (skratka od jump relative, čítaj džamp riletyv, skoč relatívne). V tomto skoku nezadávaš adresu, ale posunutie. JR môžeš použiť len na testovanie bitov Z a C.

Skoky dopredu a dozadu

Pri skokoch dopredu stačí posunutie previesť do šestnástkovej sústavy a napísať ho do programu. Pri skokoch dozadu, ktoré majú posunutie záporné, je to o niečo zložitejšie. Doteraz sme nehovorili o tom, ako v dvojkovej sústave znázorniť záporné čísla. Používa sa na to metóda nazývaná dvojkový doplnok. Napríklad číslo −6 v ňom znázorníme ako dvojkový doplnok čísla 6. V dvojkovom doplnku považujeme čísla s nulovým najvyšším bitom za kladné, s jednotkovým najvyšším bitom za záporné.

Dvojkový doplnok

1. Ak chceš získať dvojkový doplnok nejakého čísla, povedzme 6, napíš číslo najprv v dvojkovej sústave.
2. Potom zameň všetky 0 za 1 a 1 za 0. Tomu sa hovorí negovanie bitov a výsledku sa niekedy hovorí bitový doplnok.
3. K tomuto číslu pripočítaj 1. Výsledkom je dvojkový doplnok pôvodného čísla.
4. A nakoniec výsledok preveď do šestnástkovej sústavy a napíš do programu. Prevod urobíš najľahšie tak, že číslo rozdelíš v strede na dve skupiny po štyroch bitoch a každú nahradíš jednou šestnástkovou číslicou.



Číslo −6 teda zobrazíme v počítači pomocou jeho dvojkového doplnku ako FA. Takto ho musíš napísať do programu v hornej časti tejto strany. V dvojkovom doplnku môžeš v jednom bajte zobraziť čísla od −128 do 127, čiže posunutie v relatívnych skokoch môže byť najviac 128 dozadu a najviac 127 dopredu.

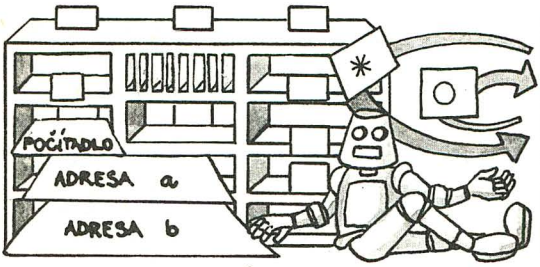
Vieš vypočítať dvojkový doplnok čísel 12, 18 a 9? (Riešenie je na strane 144.)

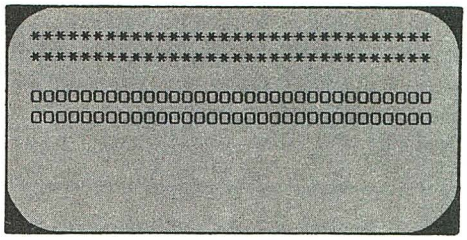
Program na blikanie obrazovky

Na tejto dvojstrane je program, ktorý bliká obrazovkou tak, že na nej vymieňa obsahy dvoch obdĺžnikov. Podobne pracujú jednoduché animačné programy. Program pre Z80 nájdeš na tejto strane, program pre 6502 na nasledujúcej. Na strane 122 je niekoľko rád — prečítaj si ich skôr, ako program spustíš.

Blikanie pre Z80

Program vymieňa obsahy dvoch obdĺžnikov na obrazovke tak, že prečíta do registrov po jednom bajte z každého obdĺžnika, a potom bajt z obdĺžnika **a** uloží na adresu bajtu z obdĺžnika **b** a naopak.





Program používa nepriame adresovanie. Adresy bajtov v obdĺžnikoch sú uložené v registroch HL a DE. Počítač tieto adresy využíva pri čítaní a zapisovaní bajtov. Po výmene jednej dvojice bajtov sa obsah HL a DE zvýši o jednotku inštrukciou INC a program sa opakuje pre ďalšiu dvojicu bajtov.

Počet bajtov, ktoré sa majú vymeniť, je na začiatku programu v registri B. Po každom zopakovaní programu sa obsah registra B zníži o jednotku. Register B je teda počítadlom. Ak B = 0, znamená to, že už sú všetky bajty vymenené.

Program pre Z80

n = počet bajtov v jednom obdĺžniku; **a** = prvá adresa obdĺžnika **a**; **b** = prvá adresa obdĺžnika **b**.

skratky	hex kódy	význam
LD B,**n**	06 **n**	Nastavenie počítadla.
LD HL,adresa **a**	21 adresa **a**	Do HL ulož adresu obdĺžnika **a**.
LD DE,adresa **b**	11 adresa **b**	Do DE ulož adresu obdĺžnika **b**.
LD C,(HL)	4E	Do C ulož bajt z adresy, ktorá je v HL (nepriame adresovanie).
LD A,(DE)	1A	do A ulož bajt z adresy, ktorá je v DE (nepriame adresovanie).
LD (HL),A	77	Obsah registra A ulož na adresu, ktorá je v HL (nepriame adresovanie).
LD A,C	79	Prenes bajt z C do A.
LD (DE),A	12	Obsah registra A ulož na adresu, ktorá je v DE (nepriame adresovanie).
INC HL INC DE	23 13	K HL a DE pripočítaj jednotku.
DEC B	05	Od počítadla v B odčítaj jednotku.
LD A,&00	3E 00	Do registra A ulož nulu.
CP B	B8	Porovnaj A s B.
JP NZ na 4. r.	C2 adresa 4. r.	Ak B nie je nula, skoč na 4. riadok programu.
RET	C9	Návrat.

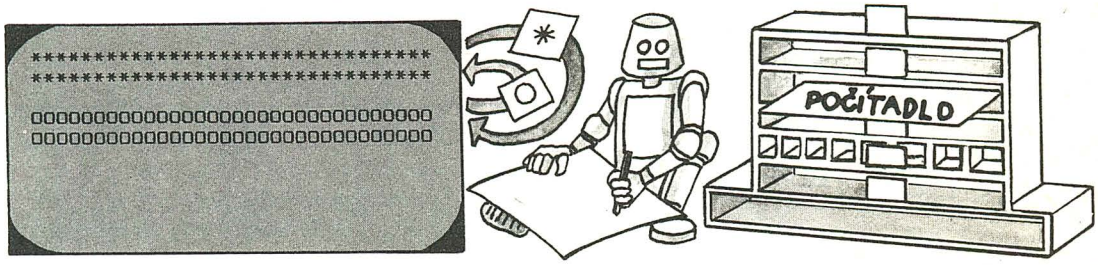
Ako nastaviť počiatočné údaje a adresy

n (počet znakov v jednom obdĺžniku) vypočítaš tak, že počet znakov v riadku vynásobíš počtom riadkov v obdĺžniku. Číslo preveď do šestnástkovej sústavy.

Adresy a a b Ak chceš vymeniť napríklad obsah prvých dvoch riadkov obrazovky s ďalšími dvoma riadkami, nastav adresu **a** na adresu začiatku obrazovky a adresu **b** na adresu obdĺžnika **a** plus počet vymieňaných znakov. Obe adresy preveď do šestnástkovej sústavy.

Blikanie pre 6502

Tento program vymieňa dva obdĺžniky na obrazovke bajt po bajte, pričom začína od posledného bajtu a končí prvým. Z oboch obdĺžnikov najprv vyberie po jednom bajte do registrov a potom uloží bajt z obdĺžnika **a** do obdĺžnika **b** a naopak. Potom sa program opakuje pre ďalšiu dvojicu bajtov.



Adresy oboch bajtov určíme indexovaným adresovaním. Do registra X sa na začiatku uloží počet vymieňaných bajtov. Pri vyberaní a ukladaní bajtov teda stačí pripočítať obsah registra X k počiatočnej adrese obdĺžnika. Inštrukcia DEX (decrement X, čítaj dekriment X, zmenši X) prikáže počítaču odčítať jednotku od obsahu registra X, čiže pri ďalšom opakovaní počítač vymení bajty na adrese o jednotku menšej.

Blikajúci program pre 6502

V programe musíš doplniť **n,** adresu **a** a adresu **b.** Na rozdiel od programu pre Z80 tu musíš napísať adresy **a** a **b** o jednotku menšie, teda také, aby po pripočítaní **n** vyšla adresa posledného bajtu v obdĺžnikoch (**n** aj adresy **a** a **b** musia byť v šestnástkovej sústave).

skratky	hex kódy	význam
LDX #**n**	A2 **n**	Naplň X počtom bajtov v obdĺžnikoch.
LDA adresa **a**,X	BD adresa **a**	Prenes bajt z adresy X+adresa **a** do akumulátora.
TAY	A8	Prenes obsah akumulátora do registra Y.
LDA adresa **b**,X	BD adresa **b**	Prenes bajt z adresy X+adresa **b** do akumulátora.
STA adresa **a**,X	9D adresa **a**	Ulož obsah akumulátora na adresu X+adresa **a**.
TYA	98	Prenes obsah registra Y späť do akumulátora.
STA adresa **b**,X	9D adresa **b**	Ulož obsah akumulátora na adresu X+adresa **b.**
DEX	CA	Zníž obsah registra X o jednotku. Ak bude X=0, nastaví sa bit Z.
BNE na 2. riadok	DO EF	Skoč o &EF bajtov nazad, ak X nie je rovné nule. EF je −17 vyjadrené v dvojkovom doplnku (pozri stranu 117).
RTS	60	Návrat.

Ako uložiť a spustiť program na Z80 a na 6502

Tento program najlepšie vyskúšaš ako podprogram v šestnástkovom ukladači:

1. Napíš do počítača šestnástkový ukladač a doplň doň v riadku 160 správny program pre mikroprocesor svojho počítača.

2. V riadku 180 potrebuješ dva cykly, ktorými uložíš písmená do pamäte obrazovky. Takto napríklad napíšeš dva riadky hviezdičiek (kód 42) a dva riadky núl (kód 48) na počítači, ktorý má v riadku 40 písmen:

```
180 FOR J=0 TO 79
190   POKE adresa obrazovky + J,42
200 NEXT J
210 FOR J=80 TO 159
220   POKE adresa obrazovky + J,48
230 NEXT J
```

3. Potom pridaj na koniec programu tieto riadky:

```
240 A=USR(adresa programu v strojovom kóde)
250 FOR K=1 TO 500
260 NEXT K
270 GOTO 240
```

Číslo 500 zmeň pre svoj počítač tak, aby bolo blikanie dobre viditeľné.

4. Teraz napíš RUN. Šestnástkový ukladač uloží tvoj program do pamäte a potom vyplní dva obdĺžniky písmenami. Riadok 240 prikáže počítaču vykonať program v strojovom kóde (musíš ho upraviť pre svoj počítač) — ten vymení obsahy obdĺžnikov len raz. Riadok 270 zabezpečí, aby sa strojový podprogram vyvolával opakovane, čo spôsobí blikanie obdĺžnikov. Program v strojovom kóde je veľmi rýchly, preto je medzi dvoma vyvolaniami zdržiavací cyklus.

Ako ďalej

O programovaní v strojovom kóde sa najviac dozvieš vtedy, keď budeš nielen skúšať svoje vlastné krátke programy, ale budeš aj čítať programy, ktoré napísali iní. Strojový kód spočiatku najlepšie využiješ v krátkych podprogramoch, ktoré vyvoláš z hlavného programu v basicu — podprogram môže totiž napríklad oveľa ľahšie vytvárať zvukové alebo obrazové efekty, lebo je oveľa kratší a rýchlejší než rovnaký program v basicu. Takéto podprogramy sa občas objavujú v časopisoch. Ak sú napísané pre tvoj typ počítača, môžeš ich použiť bezo zmien, ak sú napísané pre iný druh počítača, no s rovnakým mikroprocesorom, musíš zmeniť adresy uloženia programu, adresy použitých oblastí pamäte, prípadne adresy podprogramov v ROM.

Podprogramy v strojovom kóde

Ak chceš vložiť strojový podprogram do programu v basicu, postupuj takto:

1. Nájdi (alebo si urob) miesto v RAM (pozri strany 100–102).

2. Svoj strojový program ulož do riadka 160 šestnástkového ukladača zo strany 104. (Nezabudni na inštrukciu RET či RTS na jeho konci.) Ak treba, pridaj do šestnástkového ukladača riadky, ktoré uložia do pamäte nejaké údajové bajty. Upravený ukladač napíš do svojho počítača.

3. Riadky svojho programu začni číslovať takým číslom, ktoré nasleduje za posledným číslom riadka ukladača. Na tých miestach programu, kde sa má vykonať strojový podprogram, vlož príkaz na jeho vyvolanie (CALL či USR — podľa typu počítača).

Lepšie prostriedky na prácu v strojovom kóde

Niektoré počítače uľahčujú programovanie krátkych programov v strojovom kóde. Môžeš si na nich zvoliť, či chceš programovať v basicu, alebo komunikovať priamo s jednoduchým programom v ROM — s monitorom. Monitor zvyčajne umožňuje zapisovať do pamäte bajty v šestnástkovej sústave, zobrazovať časti pamäte na obrazovke, spúšťať strojové programy, zaznamenávať ich na magnetofón a znovu ich nahrávať do počítača.

Počítač PMD-85 uľahčuje programovanie krátkych strojových podprogramov aj inak. V basicu tohto počítača existuje príkaz CODE, ktorým možno vykonať krátky podprogram v strojovom kóde. Za slovom CODE treba napísať meno reťazcovej premennej, v ktorej je uložený strojový program v tvare šestnástkových číslic. Počítač uloží program do pamäte od adresy 7F00 (šestnástkovo) a spustí ho. Príkaz CODE teda nahrádza šestnástkový ukladač. Možno ním spúšťať strojové programy, ktoré nie sú dlhšie než 80 bajtov. Basic v PMD-85-2 a PMD-85-3 poskytuje aj ďalšie výhody — napríklad v programoch nemusíš zapisovať čísla v desiatkovej sústave, ale môžeš to robiť aj v šestnástkovej pomocou znaku ' (apostrof) pred číslom (namiesto znaku &, ktorý používame v tejto knihe), pomocou APOKE a APEEK možno zapísať do pamäte alebo prečítať z nej naraz dva bajty.

Ako ti pri programovaní v strojovom kóde môže pomôcť monitor a basic tvojho počítača, dozvieš sa z príručky.

K väčšine počítačov existuje assembler — program na kazete, diskete alebo v module, ktorý prekladá inštrukcie, zapísané pomocou skratiek, do strojového kódu. Lepšie assemblery umožňujú písať aj symbolické názvy pamäťových miest. Na pamäťové miesta sa teda netreba odvolávať ich adresami, ale menami (podobne ako v basicu). Súčasťou assemblera býva aj editor — pomocou neho píšeš a opravuješ svoj program — a ladiaci program (debugger, čítaj dybagr, program na odstránenie chýb), pomocou ktorého sleduješ krok za krokom činnosť svojho programu, a tak v ňom môžeš nájsť a odstrániť chyby. Pomocou assemblera možno napísať v strojovom kóde aj dlhé programy.

4. Napíš program v basicu do svojho počítača (už je v ňom napísaný šestnástkový ukladač) a spusti ho. Počítač bude vykonávať príkazy basicu. Keď narazí na príkaz, ktorým ho posielaš do strojového podprogramu, začne vykonávať strojové inštrukcie na zadanej adrese v pamäti. Keď nájde inštrukciu návratu, vráti sa späť na nasledujúci príkaz v basicu.

Tabuľky prevodov medzi desiatkovou a šestnástkovou sústavou

Pomocou tejto tabuľky môžeš prevádzať čísla 0 až FF zo šestnástkovej sústavy do desiatkovej a naopak.

Zo šestnástkovej sústavy do desiatkovej
Pri prevode zo šestnástkovej sústavy do desiatkovej nájdi prvú číslicu (0 až F) v prvom stĺpci tabuľky a druhú číslicu v prvom riadku tabuľky. Číslo v desiatkovej sústave prečítaš v tabuľke tam, kde sa stretne riadok a stĺpec, v ktorom si našiel jeho šestnástkové číslice. Napríklad &A1 je 161.

Z desiatkovej sústavy do šestnástkovej
Nájdi číslo v tabuľke. Prvou číslicou výsledku je označený riadok a druhou stĺpec toho políčka, v ktorom si našiel desiatkový zápis čísla. Napríklad 154 je &9A.

prvá hex číslica \ druhá hex číslica	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Ako prevádzať adresy
Ak chceš previesť adresu zo šestnástkovej sústavy do desiatkovej, preveď najprv prvé dve číslice — to je číslo stránky. Potom druhé dve číslice — umiestnenie v stránke. Výslednú adresu vypočítaš tak, že číslo stránky vynásobíš číslom 256 a pripočítaš k nemu umiestnenie v stránke.

Tabuľka dvojkového doplnku

Nájdeš v nej dvojkové doplnky čísel 1 až 128 zapísané v šestnástkovej sústave. Sú to vlastne zápisy čísel −1 až −128 v počítači. Ak chceš zistiť dvojkový doplnok čísla, nájdi ho najprv v tabuľke. Prvá šestnástková číslica výsledku je označenie riadka, druhá označenie stĺpca.

prvá hex číslica \ druhá hex číslica	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
F	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
E	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
D	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
C	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
B	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
9	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
8	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128

Z jednej sústavy do druhej pomocou kalkulačky

Ak prevádzaš číslo z jednej sústavy do druhej pomocou kalkulačky, nezabudni, že tá ti ukáže výsledok podielu ako desatinné číslo. Napríklad ak chceš previesť 134 do šestnástkovej sústavy, musíš ho vydeliť číslom 16 a výsledok aj zvyšok previesť do šestnástkovej sústavy. Kalkulačka ti zobrazí výsledok delenia 134:8 = 8.375.

Zvyšok zistíš tak, že od výsledku odčítaš číslo, ktoré je v ňom pred desatinnou bodkou, a výsledok odčítania vynásobíš tým, čím si delil, čiže 16.

$8.375 - 8 = 0.375 \times 16 = 6$

Teda 134:16 = 8 zvyšok 6, a preto 134 je &86.

Skratky a kódy inštrukcií pre Z80

Na nasledujúcich stranách nájdeš skratky a kódy inštrukcií pre Z80, ktoré sme používali v tejto knihe. O implicitnom adresovaní hovoríme vtedy, ak je operand jasný už z kódu inštrukcie, čiže nemusí byť v inštrukcii zvlášť uvedený. Z80 má aj niektoré ďalšie inštrukcie. Ak budeš časom chcieť hlbšie preniknúť do programovania v strojovom kóde, budeš potrebovať zoznam všetkých inštrukcií pre Z80. V zoznamoch používame tieto skratky:

n = číslo
nn = dvojbajtové číslo
r = register
rr = registrový pár
x = adresa
c = podmienka (test)
p = posunutie

ADC A,n Pripočítaj s prenosom číslo n k akumulátoru (adresovanie s priamym operandom).	
ADC A,n	CE n

ADC A,r Pripočítaj s prenosom obsah registra k akumulátoru (implicitné adresovanie).	
ADC A,A	8F
ADC A,B	88
ADC A,C	89
ADC A,D	8A
ADC A,E	8B
ADC A,H	8C
ADC A,L	8D

ADC HL,rr Pripočítaj s prenosom obsah páru rr k páru HL (implicitné adresovanie).	
ADC HL,BC	ED4A
ADC HL,DE	ED5A
ADC HL,HL	ED6A

ADD A,n Pripočítaj číslo n k akumulátoru (adresovanie s priamym operandom).	
ADD A,n	C6 n

ADD A,r Pripočítaj obsah registra r k akumulátoru (implicitné adresovanie).	
ADD A,A	87
ADD A,B	80
ADD A,C	81
ADD A,D	82
ADD A,E	83
ADD A,H	84
ADD A,L	85

ADD HL,rr Pripočítaj obsah páru rr k páru HL (implicitné adresovanie).	
ADD HL,BC	09
ADD HL,DE	19
ADD HL,HL	29

CALL x Vykonaj podprogram s počiatočnou adresou x (adresovanie s priamym operandom).

CALL x	CD x

CALL c,x Vykonaj podprogram s počiatočnou adresou x, ak je splnená podmienka. Podmienka c môže byť Z (rovná sa), NZ (nerovná sa), C (prenos), NC (nie je prenos), PE (párna parita), PO (nepárna parita), M (mínus), P (plus); (adresovanie s priamym operandom).

CALL Z,x	CC x
CALL NZ,x	C4 x
CALL C,x	DC x
CALL NC,x	D4 x
CALL PE,x	EC x
CALL PO,x	E4 x
CALL M,x	FC x
CALL P,x	F4 x

CCF Zmeň obsah bitu C na opačný (implicitné adresovanie).

CCF	3F

CP n Porovnaj obsah akumulátora s číslom n (adresovanie s priamym operandom).

CP n	FE n

CP r Porovnaj obsah akumulátora s registrom r (implicitné adresovanie).

CP A	BF
CP B	B8
CP C	B9
CP D	BA
CP E	BB
CP H	BC
CP L	BD

CP (HL) Porovnaj obsah akumulátora s obsahom adresy, ktorá je v HL (nepriame adresovanie).

CP (HL)	BE

DEC r Odčítaj jednotku od registra r (implicitné adresovanie).

DEC A	3D
DEC B	05
DEC C	0D
DEC D	15
DEC E	1D
DEC H	25
DEC L	2D

DEC rr Odčítaj jednotku od registrového páru rr (implicitné adresovanie).

DEC BC	0B
DEC DE	1B
DEC HL	2B
DEC IX	DD2B
DEC IY	FD2B

DEC (HL) Odčítaj jednotku od pamäťového miesta, ktorého adresa je v HL (nepriame adresovanie).

DEC (HL)	35

INC r Pripočítaj jednotku k obsahu registra r (implicitné adresovanie).

INC A	3C
INC B	04
INC C	0C
INC D	14
INC E	1C
INC H	24
INC L	2C

INC rr Pripočítaj jednotku k obsahu registrového páru rr (implicitné adresovanie).

INC BC	03
INC DE	13
INC HL	23

INC (HL) Pripočítaj jednotku k pamäťovému miestu, ktorého adresa je v HL (nepriame adresovanie).

INC (HL)	34

JP x Skoč na adresu x (adresovanie s priamym operandom).

JP x	C3 x

JP (rr) Skoč na adresu, ktorá je uložená v registrovom páre (implicitné adresovanie).

JP (HL)	E9
JP (IX)	DDE9
JP (IY)	FDE9

JP c,x Skoč na adresu x, ak je splnená podmienka c. Podmienka c môže byť Z (rovná sa), NZ (nerovná sa), C (prenos), NC (nie je prenos), PE (párna parita), PO (nepárna parita), M (mínus), P (plus); (adresovanie s priamym operandom).

JP Z,x	CA x
JP NZ,x	C2 x
JP C,x	DA x
JP NC,x	D2 x
JP PE,x	EA x
JP PO,x	E2 x
JP M,x	FA x
JP P,x	F2 x

JR p Skoč o p bajtov (p je posunutie); (relatívne adresovanie).

JR p	18 p

JR c,p Skoč o p bajtov (p je posunutie), ak je splnená podmienka c. Podmienka c môže byť NZ (nerovná sa), Z (rovná sa), NC (nie je prenos) alebo C (je prenos); (relatívne adresovanie).

JR NZ,p	20 p
JR Z,p	28 p
JR NC,p	30 p
JR C,p	38 p

LD r,n Naplň register r hodnotou n (adresovanie s priamym operandom).

LD A,n	3E n
LD B,n	06 n
LD C,n	0E n
LD D,n	16 n
LD E,n	1E n
LD H,n	26 n
LD L,n	2E n

LD rr,nn Naplň registrový pár rr hodnotou nn (adresovanie s priamym operandom).

LD BC,nn	01 nn
LD DE,nn	11 nn
LD HL,nn	21 nn

LD A,(x) Naplň akumulátor obsahom pamäťového miesta s adresou x (priame adresovanie).

LD A,(x)	3A x

LD rr,(x) Naplň registrový pár rr obsahom pamäťových miest x a x + 1 (priame adresovanie).

LD BC,(x)	ED4B x
LD DE,(x)	ED5B x
LD HL,(x)	2A x

LD A,r Presuň obsah registra r do akumulátora (implicitné adresovanie).

LD A,A	7F
LD A,B	78
LD A,C	79
LD A,D	7A
LD A,E	7B
LD A,H	7C
LD A,L	7D

LD B,r Presuň obsah registra r do registra B (implicitné adresovanie).

LD B,A	47
LD B,B	40
LD B,C	41
LD B,D	42
LD B,E	43
LD B,H	44
LD B,L	45

LD C,r Presuň obsah registra r do registra C (implicitné adresovanie).

LD C,A	4F
LD C,B	48
LD C,C	49
LD C,D	4A
LD C,E	4B
LD C,H	4C
LD C,L	4D

LD D,r Presuň obsah registra r do registra D (implicitné adresovanie).

LD D,A	57
LD D,B	50
LD D,C	51
LD D,D	52
LD D,E	53
LD D,H	54
LD D,L	55

LD E,r Presuň obsah registra r do registra E (implicitné adresovanie).

LD E,A	5F
LD E,B	58
LD E,C	59
LD E,D	5A
LD E,E	5B
LD E,H	5C
LD E,L	5D

LD H,r Presuň obsah registra r do registra H (implicitné adresovanie).

LD H,A	67
LD H,B	60
LD H,C	61
LD H,D	62
LD H,E	63
LD H,H	64
LD H,L	65

LD L,r Presuň obsah registra r do registra L (implicitné adresovanie).

LD L,A	6F
LD L,B	68
LD L,C	69
LD L,D	6A
LD L,E	6B
LD L,H	6C
LD L,L	6D

LD r,(rr) Naplň register r obsahom pamäťového miesta, ktorého adresa je v páre rr (nepriame adresovanie).

LD A,(BC)	0A
LD A,(DE)	1A
LD A,(HL)	7E
LD B,(HL)	46
LD C,(HL)	4E
LD D,(HL)	56
LD E,(HL)	5E
LD H,(HL)	66
LD L,(HL)	6E

LD (x),A Ulož obsah akumulátora na pamäťové miesto s adresou x (priame adresovanie).

LD (x),A	32 x

LD (x),rr Ulož obsah registrového páru rr do pamäťových miest s adresami x a x + 1 (priame adresovanie).

LD (x),BC	ED43 x
LD (x),DE	ED53 x
LD (x),HL	22 x

LD (rr),r Ulož obsah registra r do pamäťového miesta, ktorého adresa je v páre rr (nepriame adresovanie).

LD (BC),A	02
LD (DE),A	12
LD (HL),A	77
LD (HL),B	70
LD (HL),C	71
LD (HL),D	72
LD (HL),E	73
LD (HL),H	74
LD (HL),L	75

LD (rr),n Ulož číslo n do pamäťového miesta, ktorého adresa je v páre rr (adresovanie s priamym operandom a nepriame adresovanie).

LD (HL),n	36

RET Vráť sa z podprogramu (implicitné adresovanie).

RET	C9

RET c Vráť sa z podprogramu, ak je splnená podmienka c. Podmienka c môže byť Z (rovná sa), NZ (nerovná sa), C (prenos), NC (nie je prenos), PE (párna parita), PO (nepárna parita), M (mínus), P (plus); (implicitné adresovanie).

RET Z	C8
RET NZ	C0
RET C	D8
RET NC	D0
RET PE	E8
RET PO	E0
RET M	F8
RET P	F0

SBC A,n Odčítaj s prenosom číslo n od obsahu akumulátora (adresovanie s priamym operandom).

SBC A,n	DE n

SBC A,r Odčítaj s prenosom obsah registra r od akumulátora (implicitné adresovanie).

SBC A,A	9F
SBC A,B	98
SBC A,C	99
SBC A,D	9A
SBC A,E	9B
SBC A,H	9C
SBC A,L	9D

SBC HL,rr Odčítaj s prenosom obsah registrového páru rr od páru HL (implicitné adresovanie).

SBC HL,BC	ED42
SBC HL,DE	ED52
SBC HL,HL	ED62

SBC A,(HL) Odčítaj s prenosom obsah pamäťového miesta, ktorého adresa je v HL, od akumulátora (nepriame adresovanie).	
SBC A,(HL)	9E

SCF Nastav bit C (podmienku prenosu) (implicitné adresovanie).	
SCF	37

SUB n Odčítaj číslo n od akumulátora (adresovanie s priamym operandom).	
SUB n	D6 n

SUB r Odčítaj obsah registra r od akumulátora (implicitné adresovanie).	
SUB A	97
SUB B	90
SUB C	91
SUB D	92
SUB E	93
SUB H	94
SUB L	95

SUB (HL) Odčítaj obsah pamäťového miesta, ktorého adresa je v HL, od akumulátora (nepriame adresovanie).	
SUB (HL)	96

Riešenia hlavolamov

Strana 91

&A7 je v desiatkovej sústave 167. 513 je &201.

Strana 106

1. 25 + 73 (25 je &19, 73 je &49)

Tip: Dvojkový doplnok čísla získaš jednoducho tak, že číslo odčítaš od 256 a výsledok prevedieš do šestnástkovej sústavy. Napríklad 256 − 6 = 250 = &FA.

Z80		**6502**		**význam**
skratky	**hex kódy**	**skratky**	**hex kódy**	
LD A,&19	3E 19	LDA #&19	A9 19	Ulož &19 do akumulátora.
ADD A,&49	C6 49	ADC #&49	69 49	Pripočítaj &49 k akumulátoru.
LD (adresa),A	32 adresa	STA adresa	8D adresa	Ulož obsah akumulátora do pamäte.
RET	C9	RTS	60	Vráť sa.

2. 64 + 12 + 14 (64 je &40, 12 je &0C, 14 je &0E)

Z80		**6502**		**význam**
skratky	**hex kódy**	**skratky**	**hex kódy**	
LD A,&40	3E 40	LDA #&40	A9 40	Ulož &40 do akumulátora.
ADD A,&0C	C6 0C	ADC #&0C	69 0C	Pripočítaj &0C k akumulátoru.
ADD A,&0E	C6 0E	ADC #&0E	69 0E	Pripočítaj &0E k akumulátoru.
LD (adresa),A	32 adresa	STA adresa	8D adresa	Ulož obsah akumulátora do pamäte.
RET	C9	RET	60	Vráť sa.

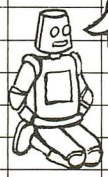
Riešenia pokračujú na strane 144.

Skratky a kódy inštrukcií pre 6502

Tabuľka zhŕňa inštrukcie procesora 6502, o ktorých sme hovorili v tejto knihe (zopár inštrukcií je pridaných). Skratka je v ľavom stĺpci a v tabuľke sú šestnástkové kódy pre rôzne spôsoby adresovania. Adresovanie v nultej stránke je vlastne rovnaké ako priame adresovanie, len adresa musí byť v nultej stránke pamäte, teda od 0 do 255 (pozri stranu 90). Implikované adresovanie označuje

situáciu, keď netreba určovať operand (napríklad pri CLC). 6502 má množstvo ďalších inštrukcií. Ak sa budeš viac venovať programovaniu v strojovom kóde, budeš potrebovať úplný zoznam inštrukcií.

spôsob adresovania	s priamym operandom	priame	v nultej stránke	indexované X	indexované Y	implikované	relatívne
operand je	údaj	ľubovoľná adresa	adresa v nultej stránke	adresa +register X	adresa +register Y	žiadny	posunutie
ADC Pripočítaj operand a podmienku C k akumulátoru.	69	6D	65	7D	79		
BCC Skoč, ak nie je prenos.							90
BCS Skoč, ak je prenos.							B0
BEQ Skoč, ak sú rovnaké.							F0
BMI Skoč, ak je záporné.							30
BNE Skoč, ak sú rôzne.							D0
BPL Skoč, ak nie je záporné.							10
BVC Skoč, ak nie je pretečenie.							50
BVS Skoč, ak je pretečenie.							70
CLC Nuluj podmienku prenosu.						18	
CMP Porovnaj s akumulátorom.	C9	CD	C5	DD	D9		
CPX Porovnaj s registrom X.	E0	EC	E4				
CPY Porovnaj s registrom Y.	C0	CC	C4				
DEC Odčítaj jednotku od pamäťového miesta.		CE	C6	DE			
DEX Odčítaj jednotku od registra X.						CA	
DEY Odčítaj jednotku od registra Y.						88	
INC Pripočítaj jednotku k pamäťovému miestu.		EE	E6	FE			
INX Pripočítaj jednotku k registru X.						E8	
INY Pripočítaj jednotku k registru Y.						C8	
JMP Skoč na adresu v operande.		4C					
JSR Skoč do podprogramu, ktorý sa začína na adrese v operande.		20					
LDA Naplň akumulátor operandom.	A9	AD	A5	BD	B9		
LDX Naplň register X.	A2	AE	A6		BE		
LDY Naplň register Y.	A0	AC	A4	BC			
RTS Vráť sa z podprogramu.						60	
SBC Odčítaj s prenosom od akumulátora.	E9	ED	E5	FD	F9		
SEC Nastav podmienku prenosu.						38	
STA Ulož akumulátor na adresu.		8D	85	9D	99		
STX Ulož register X.		8E	86				
STY Ulož register Y.		8C	84				
TAX Prenes akumulátor do registra X.						AA	
TAY Prenes akumulátor do registra Y.						A8	
TXA Prenes X do akumulátora.						8A	
TYA Prenes Y do akumulátora.						98	



Slovník strojového kódu

 # znak krížik na niektorých počítačoch sa ním označujú čísla v šestnástkovej sústave. V programoch pre mikroprocesor 6502 sú ním označené údaje v inštrukciách (adresovanie s priamym operandom)

& znak ampersand na niektorých počítačoch sa čísla v šestnástkovej sústave označujú týmto znakom, na iných sa používa znak ' (apostrof), $ (dolár) alebo písmeno H za číslom

absolútna adresa skutočná adresa údajov

absolútne adresovanie pozri priame adresovanie

adresa číslo, ktoré označuje pamäťové miesto; je to vlastne jeho poradové číslo

adresovanie s priamym operandom spôsob adresovania, pri ktorom sa údaj, s ktorým sa má pracovať, nachádza priamo v inštrukcii

adresovanie v nultej stránke používa sa len pre 6502. Je to vlastne priame adresovanie, ale operand musí byť v nultej stránke, teda na adresách od 0 do 255

akumulátor najdôležitejší register. Sú v ňom uložené údaje, s ktorými sa vykonávajú aritmetické alebo logické operácie

aritmeticko-logická jednotka časť mikroprocesora, v ktorej sa vykonávajú aritmetické a logické operácie

assembler program, ktorý prekladá inštrukcie zapísané v assemblerovom jazyku (teda pomocou skratiek) do strojového kódu. Slovom assembler niekedy označujeme aj assemblerový jazyk

assemblerový jazyk jazyk, v ktorom sa zapisujú strojové inštrukcie pomocou skratiek a adresy údajov môžu byť nahradené menami. V odbornej literatúre sa používa názov jazyk symbolických adries

bajt (byte, čítaj bajt) osem bitov, teda osemciferné číslo v dvojkovej sústave. Bajt je zvyčajne najmenšia jednotka pamäte počítača

bit jedna číslica v dvojkovej sústave: 0 alebo 1. V počítači je zobrazená ako „je impulz" alebo „nie je impulz"

bitový doplnok čísla (complement, čítaj komplement) číslo, ktoré má v dvojkovej sústave vymenené hodnoty všetkých bitov z 1 na 0 a naopak. Napríklad bitový doplnok čísla 00000101 je 11111010

cieľový program program v strojovom kóde, ktorý bol preložený z assemblerového jazyka alebo z nejakého vyššieho programovacieho jazyka

disassembler program, ktorý dokáže zobraziť obsah časti pamäte v assemblerovom jazyku. Je užitočný pri ladení programov a pri zisťovaní obsahu ROM. K niektorým počítačom existuje disassembler na kazete alebo v module ROM

dvojková sústava číselná sústava, ktorá používa len dve číslice (0 a 1)

dvojkový doplnok spôsob zápisu záporných čísel. Dvojkový doplnok vypočítaš takto: nájdeš bitový doplnok čísla (zameníš nuly za jednotky a jednotky za nuly) a pripočítaš k nemu jednotku

implicitné adresovanie spôsob adresovania, pri ktorom je operand jasný už z kódu inštrukcie a nemusí byť v inštrukcii zvlášť určený

implikované adresovanie rovnaké ako implicitné adresovanie

indexované adresovanie spôsob adresovania, pri ktorom sa adresa údaja vypočíta tak, že sa sčíta adresa v inštrukcii s obsahom indexového registra

indexový register používa sa pri indexovanom adresovaní na vypočítanie adresy pamäťového miesta

inštrukcia príkaz pre základnú jednotku počítača

interpreter program, ktorý riadi vykonávanie programu v basicu

kód inštrukcie časť inštrukcie, ktorá hovorí počítaču, čo má urobiť

mikroprocesor integrovaný obvod, ktorý

obsahuje základnú jednotku mikropočítača. Vykonáva všetky inštrukcie, teda riadi prácu celého mikropočítača

nepriame adresovanie spôsob adresovania, pri ktorom je adresa údaja uložená v registri alebo v pamäti (pre Z80 len v registri)

nižší bajt dve pravé číslice v šestnástkovom zápise adresy, ktoré určujú umiestnenie v stránke pamäte

nultá stránka prvých 256 pamäťových miest

operačný systém skupina programov v strojovom kóde, ktoré počítaču predpisujú, ako má robiť základné úkony, inak povedané zariadia, že sa počítač „správa rozumne". Operačný systém mikropočítačov býva uložený v ROM

operand časť inštrukcie, ktorá hovorí počítaču, kde nájde údaje, s ktorými má niečo urobiť

pamäť obrazovky časť RAM, v ktorej je uložená informácia práve zobrazovaná na obrazovke

počítadlo adries (program counter, čítaj prougräm kauntr, skratka PC) register obsahujúci adresu inštrukcie, ktorá sa má vykonať ako ďalšia

podmienený skok pri podmienenom skoku počítač podľa nastavenia niektorej podmienky buď pokračuje vo vykonávaní inštrukcií na nasledujúcej adrese, alebo skočí na iné miesto v pamäti

podmienka bit v registri podmienok. Jeho nastavenie označuje, že nastala nejaká udalosť, napríklad prenos pri sčítaní, zhoda čísel pri porovnávaní

podmienka nuly bit v registri podmienok, ktorý sa nastaví vtedy, ak je výsledkom operácie nula alebo ak sú porovnávané čísla rovnaké

podmienka prenosu bit v registri podmienok, ktorý je nastavený vždy, ak sa výsledok sčítania nezmestí do ôsmich bitov

podmienka znamienka bit v registri podmienok, ktorý rozlišuje záporné (ak je nastavený) a nezáporné (ak je nulový) výsledky operácií

posunutie číslo, ktoré sa používa v skokových inštrukciách. Hovorí počítaču, koľko pamäťových miest má preskočiť

používateľova RAM časť pamäte, v ktorej je uložený program v basicu

priame adresovanie spôsob adresovania, pri ktorom je v inštrukcii uložená adresa údaja. Niekedy sa nazýva aj absolútne adresovanie

RAMTOP označenie poslednej adresy v oblasti používateľovej RAM na počítači ZX Spectrum

register miesto pre údaje v základnej jednotke. Sem sa uložia údaje, s ktorými procesor vykonáva operácie

relatívne adresovanie spôsob adresovania, pri ktorom sa adresa (väčšinou skoku) vypočíta tak, že sa sčíta obsah počítadla adries (PC) a posunutia, ktoré je zadané v inštrukcii

šestnástkový ukladač program v basicu, ktorý prekladá šestnástkové zápisy inštrukcií do strojového kódu a ukladá ich do pamäte počítača

skok inštrukcia, ktorá prikáže počítaču pokračovať vo vykonávaní inštrukcií na inom mieste pamäte

smerník (pointer, čítaj pointr) dva bajty v pamäti alebo register či registrový pár, v ktorom je uložená adresa nejakého údaju

stavový register procesora register podmienok v procesore 6502

stránka pamäte časť pamäte; v mikropočítačoch je to najčastejšie 256 bajtov

súbor inštrukcií všetky operácie, ktoré vie vykonať niektorý mikroprocesor

systémové premenné pamäťové miesta v RAM, ktoré obsahujú informáciu o tom, v akom stave sa práve nachádza počítač

šestnástková sústava (skrátene hex) číselná sústava, ktorá používa šestnásť číslic (označujeme ich 0 až 9 a A až F). Každá číslica v čísle má 16-krát väčšiu hodnotu než číslica vpravo od nej

ukazovateľ zásobníka (stack pointer, čítaj stek pointr, skratka SP) register v základnej jednotke, ktorý obsahuje adresu poslednej položky v zásobníku

vyrovnávacia pamäť (buffer, čítaj bafr)

časť pamäte, kam sa dočasne ukladajú údaje na ich ceste z miesta určenia (napríklad z magnetofónu), prípadne na ceste na miesto určenia (napríklad na tlačiareň)

vyšší bajt dve ľavé číslice v šestnástkovom zápise adresy, ktoré určujú číslo stránky s pamäťovým miestom. Hovorí vlastne, koľkokrát sa v čísle nachádza 256

zásobník (stack, čítaj stek) časť pamäte, ktorá sa využíva spôsobom „posledný uložený, prvý vybraný". To, čo sa do zásobníka uloží ako posledné, sa vyberá ako prvé. Takto sa ukladajú napríklad adresy návratov z podprogramov

zdrojový program program napísaný v assemblerovom jazyku alebo vo vyššom programovacom jazyku (napríklad v basicu)

Riešenia hlavolamov

Na nasledujúcich stranách nájdeš riešenia programátorských hlavolamov a cvičení z prvej časti knihy. Programy sú napísané v bežnom basicu. Niektoré príkazy, napr. RND a GCLEAR, budeš musieť zmeniť tak, aby im tvoj počítač rozumel. Ak niektorý z programov nefunguje, najprv skontroluj, či tvoj počítač pozná všetky použité slová basicu. Pri úprave programov ti pomôže tabuľka na strane 147. Riadky, ktoré treba pre počítače Sinclair podstatne zmeniť, sú označené & (vedľa programu vždy nájdeš aj návod, ako ich zmeniť). Pre počítač ZX81 sú označené @

Možno si niektoré z hlavolamov riešil ináč než my. Ak program na tvojom počítači pracuje, je to v poriadku. Pozri si však, či je tvoje riešenie aspoň také pekné a jednoduché ako naše.

Príkaz PRINT (strany 14–15)

Programátorské hlavolamy

1
```
40 PRINT "AHOJ ";
50 PRINT A$
```

Bodkočiarka spôsobí, že nasledujúcu správu počítač napíše do toho istého riadka.

2
```
40 PRINT "AHOJ",A$
50 Vymaž tento riadok
```

Vymaž tento riadok.
Čiarka spôsobí, že počítač vynechá niekoľko medzier pred A$.

3
```
40 PRINT TAB(6);"AHOJ ";A$
50 Vymaž tento riadok
```

Vymaž tento riadok.
Na niektorých počítačoch netreba za TAB písať bodkočiarku.

4
```
40 PRINT "          DOVIDENIA"
50 PRINT "          ";A$
```

Premenné (strany 16–17)

Výber mena premennej

Tieto názvy obsahujú slová basicu a nemôžeš ich použiť ako mená premenných: LETO$ (LET): RUN$ (RUN); TOPOĽ$ (TO). Počítač ZX81 pripúšťa iba jednopísmenkové mená premenných. takže na ňom nemôžeš použiť ani premennú DESAŤ.

Hlavolam s príkazom PRINT

```
10 LET A=66
20 LET B=77
30 LET HR$="HRDZAVÝCH ROBOTOV ZJE"
40 PRINT A;" ";HR$;
50 PRINT B;" OPEKANÝCH SAFALÁDOK"
```

Medzera vnútri úvodzoviek spôsobí, že počítač napíše medzeru.

Pohľad do programu

```
45 PRINT "C JE ";C
55 PRINT "D JE ";D
```

Hlavolam stupne Celzia a Fahrenheita

```
40 LET C=9/B
50 LET D=C*A
60 LET R=D+F
```

Zameň riadky 40 – 60.

Ďalšie programátorské hlavolamy

1

```
10 LET U$="UG"
20 PRINT "SLOVENSKÉ SLOVO"
30 INPUT S$
40 PRINT "UGLIJSKÉ SLOVO";
50 PRINT U$;S$
```

V riadku 50 počítač najprv napíše UG a potom slovo uložené v S$.

2

```
10 PRINT "KOĽKO METROV SI UBEHOL? ";
20 INPUT S
30 PRINT "ZA KOĽKO SEKÚND? ";
40 INPUT T
50 LET V=S/T
60 PRINT "BEŽAL SI RÝCHLOSŤOU ";V;
70 PRINT T;"HODINY"
```

Rýchlosť sa vypočíta v riadku 50 a uloží sa do premennej V.

3

```
10  LET A=30
20  LET B=20
30  LET S=35
40  LET T=S/B
50  LET C=A*T
60  LET D=C+S
70  PRINT "MUSIA SI KÚPIŤ ";
80  PRINT D;" SAFALÁDOK."
90  PRINT "ZJEDIA ICH ZA ";
100 PRINT T;" HODINY."
```

A a B sú počty safaládok, ktoré roboti zjedia za hodinu. S je počet safaládok, ktoré chce zjesť Modráčik, T je čas, ktorý potrebuje na jedenie. V riadku 50 sa vypočíta, koľko safaládok zje za čas T Červenáčik, D je potrebný počet safaládok.

Opakovanie činností (strany 18–19)

Hlavolamy s cyklami

1

```
10 FOR J=1 TO 100
20 PRINT "AHOJ ";
30 NEXT J
```

Bodkočiarka v riadku 20 znamená, že nasledujúcu správu má počítač napísať do toho istého riadka.

2

```
10 FOR J=1 TO 25
20 PRINT TAB(15);"AHOJ"
30 NEXT J
```

Počítač napíše 25-krát pod seba AHOJ, na začiatku každého riadka vynechá 15 medzier.

3

Nesprávny je riadok 20, pretože mení hodnotu premennej cyklu I. Pri každom opakovaní cyklu sa zmení hodnota I z 1 na 0.

Kroky

1

```
10 FOR I=25 TO 1 STEP -1
20 PRINT TAB(I);"AHOJ"
30 NEXT I
```

Premennú cyklu použi v TAB. Presvedč sa, či I nie je väčšie ako šírka tvojej obrazovky. Pri každom opakovaní cyklu sa hodnota I zmenší o 1.

2

```
10 FOR L=5 TO 0 STEP -1
20 PRINT TAB(5);L;TAB(10);L*L
30 NEXT L
```

Krok –1 spôsobí, že hodnota premennej v cykle L bude klesať od 5 do 0. Pri každom opakovaní cyklu počítač vypíše L a jeho druhú mocninu.

3

```
10 PRINT "ŠTART ";
20 INPUT A
30 PRINT "STOP ";
40 INPUT B
50 PRINT "KROK ";
60 INPUT C
70 FOR J=A TO B STEP C
80 PRINT J;"  ";
90 NEXT J
```

V cykle v riadkoch 70 – 90 sa použije A ako počiatočná hodnota, B ako konečná hodnota premennej v cykle a C ako krok.

Tajné posolstvo

```
10 PRINT "TAJNÉ POSOLSTVO"
20 PRINT "OBJAVÍ SA NA 5 SEKÚND"
30 PRINT "A POTOM ZMIZNE"
40 PRINT
50 PRINT "ČAKAJ POSLA X 2.00 LETISKO"
60 FOR I=1 TO 1000
70 NEXT
80 GCLEAR
```

60 FOR I=1 TO 1000 … 70 NEXT — zdržiavací cyklus

80 GCLEAR — vymazanie obrazovky

Obrázkové hlavolamy

2

```
50 PRINT TAB(A-K);"*";
60 PRINT TAB(A+K);"*"
70 NEXT K
```

Najvrchnejšia hviezdička je v A-tom stĺpci. Ľavá strana sa nakreslí, ak od A odčítaš K, pravá, ak k A pripočítaš K.

3

```
30
40 FOR K=9 TO 1 STEP -1
80 PRINT TAB(A);"*"
```

Vymaž tento riadok.
Zmeň tieto riadky, aby sa obrázok kreslil zhora nadol. Podobne vytvoríš programy aj pre ostatné obrázky, zmení sa len číslo za TAB.

Cykly (strany 20–21)

Cyklus, ktorý spomalí vykonávanie programu

Tento vnorený cyklus môžeš napísať do riadku 45 alebo 55.

```
45 FOR K=1 TO 1000
48 NEXT K
```

Chyby v cykloch

Chyby sú v riadkoch 50 a 70. Správne by mali vyzerať takto:

```
50 NEXT K
70 NEXT I
```

Dvojkové počítadlo

Program, ktorý bude písať osemciferné dvojkové čísla, vytvoríš pridaním štyroch vnorených cyklov (E, F, G, H). Prečísluj riadky programu a zmeň riadok s príkazom PRINT:

```
PRINT H+G*2+F*4+E*8+D*16+C*32+B*64+
A*128;" = ";
PRINT A;B;C;D;E;F;G;H
```

Blikajúca správa

```
10 FOR J=1 TO 10
20 GCLEAR
30 FOR K=1 TO 1000:NEXT K
40 PRINT TAB(5):"NEBEZPEČENSTVO"
50 PRINT TAB(6);" VZDUŠNÝ ÚTOK"
60 FOR K=1 TO 1000:NEXT K
70 NEXT J
```

Tento program napíše blikajúcu správu. Možno budeš musieť zmeniť veľkosť zdržiavacieho cyklu a číslo v TAB. Ak na svojom počítači nemôžeš napísať do jedného riadka viac príkazov, napíš NEXT K do nového riadka.

Počítačové hodiny

```
10 FOR J=0 TO 59
20 FOR K=0 TO 59
30 PRINT J;":";K
40 FOR L=1 TO 500:NEXT L
50 GCLEAR
60 NEXT K
70 NEXT J
```

minútový cyklus — 10 FOR J=0 TO 59 … 70 NEXT J

sekundový cyklus — 20 FOR K=0 TO 59 … 60 NEXT K

Zvoľ veľkosť cyklu K tak, aby trval 1 sekundu.

Program Štart rakety

```
 10  GCLEAR
@20  FOR I=1 TO 20
 30  PRINT
 40  NEXT I
 50  PRINT "    *    "
 60  PRINT "   ***   "
 70  PRINT "   ***   "
 80  PRINT "   ***   "
 90  PRINT " ** ** "
 100 FOR J=1 TO 25
@110 PRINT
 120 FOR K=1 TO 1000
 130 NEXT K
 140 NEXT J
```

120 FOR K=1 TO 1000 … 130 NEXT K — zdržiavací cyklus

V cykle v riadkoch 20–40 sa napíšu prázdne riadky. V riadkoch 50–90 sa na spodný okraj obrazovky nakreslí raketa. Cyklus v riadkoch 100–140 sa môže zopakovať toľkokrát, koľko riadkov má tvoja obrazovka. Pri každom opakovaní cyklu sa napíše prázdny riadok a raketa sa posunie hore.

Zmeny pre počítače Sinclair

Na počítači Spectrum musíš po tom, čo sa objaví na obrazovke správa, stlačiť ENTER.

(a) Pre ZX81 zmeň tieto riadky:

```
20  FOR I=1 TO 17
110 SCROLL
```

Program Skákajúci panáčik

```
10  FOR J=1 TO 25 STEP 2
20  GCLEAR
30  PRINT
40  PRINT TAB(J);" O "
50  PRINT TAB(J);"<0>"
60  PRINT TAB(J);"/ \"
70  FOR K=1 TO 1000
80  NEXT K
90  GCLEAR
100 PRINT TAB(J+1);"<O>"
110 PRINT TAB(J+1);" 0 "
120 PRINT TAB(J+1);"/ \"
130 FOR K=1 TO 1000
140 NEXT K
150 NEXT J
```

Veľkosť cyklu v riadkoch 10–150 zvoľ podľa šírky svojej obrazovky. V riadkoch 40–60 sa nakreslí panáčik v prvej pozícii. Cykly v riadkoch 70–80 a 130–140 sú zdržiavacie — panáčik sa zakaždým na chvíľu zastaví. V riadkoch 100–120 sa nakreslí panáčik v druhej pozícii.

Príkaz IF/THEN (strany 22–23)

Násobilka

```
55 IF B <> J*A THEN PRINT "ZLE. ";
     J;" X ";A;" = ";J*A
```

Ak pridáš do programu takýto riadok, počítač ti odpovie aj vtedy, keď je tvoja odpoveď nesprávna.

Heslo

```
50 IF P$ <> S$ THEN PRINT "ZLE.
     ODÍĎ!";
```

Doplň do programu tento riadok (správu môžeš zmeniť).

```
10 LET S$="SAFALÁDKY"
20 LET N=007
30 PRINT "NAPÍŠ HESLO"
40 INPUT P$
50 PRINT "TAJNÉ ČÍSLO ";
60 INPUT SN
70 IF P$ <> S$ OR SN <> N THEN PRINT
   "ZLE. ODÍĎ!"
80 IF P$ = S$ AND SN = N THEN "DOBRE.
   POKRAČUJ!"
```

Takto zmenený program Heslo sa ťa spýta na tajné slovo a číslo.

Počítač — kalkulačka

```
10  PRINT "ZADAJ ČÍSLO ";
20  INPUT X
30  PRINT "ĎALŠIE ";
40  INPUT Y
50  PRINT "CHCEŠ ICH: "
60  PRINT "SČÍTAŤ, ODČÍTAŤ,"
70  PRINT "VYDELIŤ ALEBO VYNÁSOBIŤ?"
80  INPUT A$
90  IF A$ = "SČÍTAŤ" THEN LET A=X+Y
100 IF A$ = "ODČÍTAŤ" THEN LET A=X-Y
110 IF A$ = "VYDELIŤ" THEN LET A=X/Y
120 IF A$ = "VYNÁSOBIŤ" THEN LET A=X*Y
130 PRINT "VÝSLEDOK JE ";A
```

Počítač vykoná požadovanú aritmetickú operáciu podľa odpovede v A$.

Hádanie čísel a slov

```
45 IF Y < X THEN PRINT "MÁLO"
46 IF Y > X THEN PRINT "VĚĽA"
```

Pridaj k programu tieto riadky, aby ti počítač oznamoval, či je tvoj odhad vysoký alebo nízky.

```
5 LET N=0
55 LET N=N+1
56 IF N > 5 THEN STOP
```

V programe sa budú počítať pokusy. Hráč má povolených 5 pokusov.

Hádanie slova

```
10  PRINT "ZADAJ SLOVO ";
20  INPUT S$
30  PRINT "A POMÔCKU ";
40  INPUT P$
50  GCLEAR
60  PRINT "POMÔCKA : ";
70  PRINT P$
80  PRINT "HÁDAJ SLOVO "
90  INPUT H$
100 IF H$ = S$ THEN GOTO 130
110 PRINT "NIE"
120 GOTO 90
130 PRINT "ÁNO"
```

V riadkoch 10–40 sa načíta slovo a pomôcka.
V riadkoch 50–60 sa vyčistí obrazovka a vypíše pomôcka.
Ak hráč slovo neuhádne, riadok 120 pošle počítač späť na riadok 90.

Dostihy

Označené riadky programu Dostihy vyzerajú po doplnení takto:

```
80  IF G1 = H1 AND G2 = H2 THEN GOTO 160
90  IF G1 = H2 OR G2 = H1 THEN GOTO 150
100 IF (G1 = H1 AND G2 <> H2) OR (G2 = H2
    AND G1 <> H1) THEN GOTO 140
120 IF N = 4 THEN GOTO 170
```

Riadok 80: Ak hráč určil správne poradie koní, počítač pokračuje riadkom 160.
Riadok 90: Ak správne vybral koňa, ale zle určil jeho umiestnenie, počítač pokračuje riadkom 150.
Riadok 100: Ak určil správne umiestnenie len jedného koňa, počítač pokračuje riadkom 140.
Riadok 120: Ak vyčerpal všetky 4 pokusy, počítač pokračuje riadkom 170.

Ako hru zlepšiť

```
15  LET B=0
140 PRINT "JEDEN ODHAD SPRÁVNY":LET
    B=2: GOTO 120
160 PRINT "SPRÁVNE":LET B=4
180 PRINT "POČET ZÍSKANÝCH BODOV: ";B
```

Navrhujeme bodovanie, v ktorom sa pridelia hráčovi 4 body, ak uhádne víťaza aj druhého v poradí, a 2 body, ak uhádne len jedného. Počet bodov je uložený v premennej B.

```
190 PRINT "CHCEŠ POKRAČOVAŤ V HRE? (A/N)"
200 INPUT Z$
210 IF Z$ = "A" THEN GOTO 10
220 IF Z$ = "N" THEN STOP
```

Pridaním týchto riadkov k programu umožníš hráčovi, aby sa rozhodol, či chce v hre pokračovať.

Náhodné čísla (strany 24–25)

Náhodné čísla od 10 do 20

```
INT(RND(1)*11+10)
```

Vytvoríš ich tak, že náhodné číslo vynásobíš počtom čísel v intervale (11) a k súčinu pripočítaš dolnú hranicu intervalu (10). RND použi v tvare zrozumiteľnom tvojmu počítaču.

Programátorský hlavolam

```
10 LET X=INT(RND(1)*20+1)
```

Zmeň riadok 10. Počítač náhodne vyberie číslo od 1 do 20.

Číselná postupnosť

K programu pridaj nasledujúce riadky:

```
5   PRINT "UHÁDNI NASLEDUJÚCE"
6   PRINT "ČÍSLO V TEJTO"
7   PRINT "POSTUPNOSTI"
10  LET X=INT(RND(1)*10+1)
20  LET Y=INT(RND(1)*10+1)
60  LET A=X+4*Y
70  INPUT N
80  IF N = A THEN PRINT "SPRÁVNE"
90  IF N <> A THEN PRINT "ZLE. NASLEDUJE
    ČÍSLO ";A
100 GOTO 5
```

V riadku 60 sa vypočítava nasledujúci člen postupnosti.

Odlet z planéty Zorgos

```
10  LET C=10
```
V C je uložený počet hráčových čipov.

```
20  PRINT "POČET TVOJICH ČIPOV:";C
30  PRINT "TVOJ VKLAD: ";
40  INPUT V

50  IF V > C THEN PRINT "NEMÁŠ TOĽKO
    ČIPOV":GOTO 20
```
Počítač sa presvedčí, či hráč nevložil do hry viac čipov, než má.

```
60  LET C=C-V
```
Vypočíta, koľko čipov hráčovi zostalo.

```
70  LET X=INT(RND(1)*6+1)
80  LET Y=INT(RND(1)*6+1)
```
Náhodne vyberie čísla od 1 do 6.

```
90  PRINT "AK SI PRIPRAVENÝ, STLAČ P:";
100 INPUT P$
110 IF P$ <> "P" THEN GOTO 90
```
Skontroluje hráčovu odpoveď.

```
120 PRINT TAB(5);X;TAB(10);Y
```
Napíše náhodne vybrané čísla na obrazovku.

```
130 IF X = Y THEN GOTO 250
140 IF X+Y = 10 OR X+Y = 11 THEN GOTO 210
150 IF X+Y = 6 OR X+Y = 7 THEN GOTO 190
```
Podľa výsledku pokračuje v rôznych riadkoch.

```
160 PRINT "ĽUTUJEM, VKLAD SI PREHRAL"
170 IF C=0 THEN GOTO 290
```
Ak hráčovi nezostal ani jeden čip, hra sa končí.

```
180 GOTO 20
```
Návrat, ďalšie kolo hry.

```
190 PRINT "VKLAD SA TI VRÁTIL"
200 LET C=C+V:GOTO 20
```
Počítač zmení hodnotu C.

```
210 PRINT "VYHRAL SI TROJNÁSOBOK VKLADU"
220 LET C=C+(V*3)
230 IF C >= 50 THEN GOTO 320
```
Ak C je väčšie než 50 alebo rovné 50, hráč vyhráva.

```
240 GOTO 20
250 PRINT "VYHRAL SI DVOJNÁSOBOK VKLADU"
260 LET C=C+(V*2)
270 IF C >= 50 THEN GOTO 320
280 GOTO 20
290 PRINT "PREHRAL SI VŠETKY ČIPY!"
300 PRINT "TVOJ ODLET JE NEUSKUTOČNITEĽNÝ"
310 STOP
320 PRINT "BLAHOŽELÁM. MÔŽEŠ ODLETIEŤ"
330 PRINT "MÁŠ ";C;" ČIPOV"
```

Náhodná postupnosť

```
24 LET R=INT(RND(1)*3+1)
25 IF R = 1 THEN GOTO 30
26 IF R = 2 THEN GOTO 40
27 IF R = 3 THEN GOTO 50
```

Počítač náhodne vyberie číslo od 1 do 3 a zaznamená si ho v R. V závislosti od hodnoty R vyberie typ postupnosti.

```
30 FOR I=1 TO 3
33 PRINT X+I*I
35 NEXT I
37 LET A=X+4*4
39 GOTO 70
40 FOR I=1 TO 3
43 PRINT I*I-Y
45 NEXT I
47 LET A=4*4-Y
49 GOTO 70
50 FOR I=1 TO 3
53 PRINT X+Y-I*I
55 NEXT I
57 LET A=X+Y-4*4
```

Z programu vymaž riadky 30 až 50 a pripoj k nemu tieto tri cykly. V každom sa vytvára iná postupnosť. Po skončení prvého a druhého cyklu pokračuje program riadkom 70.

Papier, kameň alebo nožničky: odhaľ chyby

V programe treba opraviť nasledujúce riadky:

```
50  LET R=INT(RND(1)*3+1)
80  IF R = 3 THEN LET C$="NOŽNIČKY"
150 IF C$ = "NOŽNIČKY" AND A$="KAMEŇ"
    THEN LET F=1
210 IF F = 1 THEN PRINT "VYHRAL SI"
230 IF F = 0 THEN LET C=C+1
240 IF F = 1 THEN LET A=1+A
280 IF C < 10 AND A < 10 THEN GOTO 40
```

Znakové reťazce (strany 26–27)

Hlavolamy s reťazcami

1

```
&40 PRINT TAB(K);MID$(N$,K,1)
```

Pri každom vykonaní príkazu sa napíše K-té písmeno na K-tú pozíciu v riadku.

2

```
 10 LET K$="KROKODÍL"
 20 LET L=LEN(K$)
 30 FOR J=L TO 1 STEP -1
&40 PRINT MID$(K$,J,1)
 50 NEXT J
```

Pri vypisovaní slova odzadu použi cyklus s krokom −1.

3

```
&50 PRINT RIGHT$(S$,L-J)+LEFT$(S$,J)
```

Pri J-tom opakovaní cyklu sa napíše L − J posledných písmen a J prvých písmen slova ŠTVOREC (L je dĺžka slova). Program spusti s rôznymi slovami.

Najdlhšie slovo

```
50 IF LEN(S$) > LEN(A$)
   THEN LET A$=S$
```

Pri každom opakovaní cyklu porovná počítač dĺžky slov A$ a S$. Ak S$ je dlhšie ako A$, nahradí slovo A$ slovom S$ a ďalej porovnáva. Po ukončení cyklu je v A$ uložené najdlhšie slovo.

Najkratšie slovo

```
10 LET A$="<>@@@$@%%^*()RTEW!!!;
   XXXXXXXXXXXXXXXX"
50 IF LEN(S$) < LEN(A$)
   THEN LET A$=S$
70 PRINT "NAJKRATŠIE SLOVO : "
```

Tento program sa od predchádzajúceho líši len riadkami 10 a 50. V riadku 10 sa do A$ uloží dlhý reťazec, s ktorým sa budú porovnávať všetky slová. Nie je podstatné, aké znaky tento reťazec obsahuje.

Zámena slov vo vete

Neúplné riadky v programe treba doplniť takto:

```
40 LET V$=" "+V$+" "
70 LET S$=" "+S$+" "
```

Na začiatok a na koniec reťazcov V$ a S$ musíš v riadku 140 pridať medzeru preto, aby počítač vyhľadával iba celé slová. Ak by si to nespravil, mohol by nahradiť aj časť niektorého slova. Skús program spustiť bez medzier a pozoruj, ako bude počítač pracovať.

```
100 LET LV=LEN(V$)
110 LET LS=LEN(S$)
```

Do LV sa uloží dĺžka reťazca V$ a do LS dĺžka S$. Príkaz z riadku 100 sa vykoná ešte raz (riadok 160), aby sa do LV uložila dĺžka novovytvorenej vety.

```
&140 IF MID$(V$,K,LS) = S$
     THEN LET A$=V$
```

Pri K-tom opakovaní cyklu sa s S$ porovná LS znakov reťazca V$ počnúc K-tým (LS je dĺžka S$). Ak niektorá postupnosť znakov je rovná S$, uloží sa celá veta do novej premennej A$.

```
&150 IF A$ = V$ THEN LET V$=LEFT$(A$,K)
     +N$+RIGHT$(A$,LV-(K+LS-2))
```

LEFT$(A$,K) prikáže počítaču vziať znaky nachádzajúce sa vo vete vľavo od slova, ktoré nahrádzame. Príkazu najlepšie porozumieš, ak si na kus papiera napíšeš vetu a skúsiš príkaz vykonať.

```
180 IF K <= LV-LS+1 THEN GOTO 140
```

Program pokračuje riadkom 140 a hľadá sa ďalšie slovo, ktoré treba nahradiť.

Zmeny pre počítače Sinclair

Hlavolamy s reťazcami

Do programov 1, 2 a 3 vsuň nasledujúce riadky:

1
```
40 PRINT TAB(K);N$(K TO K)
```

2
```
40 PRINT K$(J TO J)
```

3
```
50 PRINT S$(J+1 TO )+S$( TO J)
```

Zámena slov vo vete

Zmeň tieto riadky:

```
140 IF V$(K TO K+LS-1) = S$ THEN
    LET A$=V$
150 IF A$ = V$ THEN LET V$=A$( TO K)
    +N$+A$(K+LS-1 TO )
```

Ešte raz o znakoch (strana 28)

Písmenkové hlavolamy

1
```
@10 FOR K=65 TO 90
```

Čísla 65 až 90 sú kódmi písmen abecedy v kóde ASCII. Pre počítače Sinclair musíš použiť čísla 38 až 63.

2
```
10 FOR K=97 TO 122
20 PRINT CHR$(K);
30 NEXT K
```

Týmto spôsobom napíšeš malú abecedu (na počítači, ktorý ju pozná).

3
```
@10 LET R=INT(RND(1)*26+65)
  20 PRINT CHR$(R);
  30 GOTO 10
```

Toto je najjednoduchší spôsob, ako napísať náhodnú postupnosť písmen. Pre počítač ZX81 nahraď číslo 65 (kód prvého písmena abecedy) číslom 38.

Porovnávanie písmen

```
10 INPUT X$,Y$
20 IF X$ < Y$ THEN PRINT X$;" JE
   PRED ";Y$
30 IF X$ > Y$ THEN PRINT Y$;" JE
   PRED ";X$
40 GOTO 10
```

Ak na svojom počítači môžeš načítať naraz len jeden údaj, vlož do programu ešte jeden riadok s príkazom INPUT.

Malé — veľké

```
50 IF X$ >= "a" AND X$ <= "z" THEN PRINT
   CHR$(ASC(X$)-32);
60 IF X$ >= "A" AND X$ <= "Z" THEN PRINT
   CHR$(ASC(X$)+32);
```

V riadkoch 50 a 60 chýba číslo 32, čo je rozdiel medzi kódom malého a veľkého písmena. Over si to v tabuľke kódov vo svojej príručke.

Šifrujeme správu (strana 29)

Šifra

Nájdeš tu úplný program s doplnenými číslami a znakmi. Ak ho chceš spustiť na ZX Spectrum, nahraď príkaz ASC príkazom CODE. Verzia programu pre počítač ZX81 je vpravo.

```
 10  PRINT "NAPÍŠ TAJNÚ SPRÁVU"
 20  INPUT S$
 30  FOR J=1 TO LEN(S$)
&40  LET X=ASC(MID$(S$,J,1))
 50  IF X < 65 OR X > 90
     THEN LET N=X: GOTO 100
 60  IF INT(J/2) = J/2 THEN LET N=X+1
 70  IF INT(J/2) <> J/2 THEN LET N=X-1
 80  IF N < 65 THEN LET N=N+26
 90  IF N > 90 THEN LET N=N-26
 100 PRINT CHR$(N);
 110 NEXT J
```

Šifrujeme správu na počítači ZX81

```
10  PRINT "NAPÍŠ TAJNÚ SPRÁVU"
20  INPUT S$
30  FOR J=1 TO LEN(S$)
40  LET X=CODE(S$(J TO J))
50  IF X < 38 OR X > 63 THEN LET N=X
55  IF X < 38 OR X > 63 THEN GOTO 100
60  IF INT(J/2) = J/2 THEN LET N=X+1
70  IF INT(J/2) <> J/2 THEN LET N=X-1
80  IF N < 38 THEN LET N=N+26
90  IF N > 63 THEN LET N=N-26
100 PRINT CHR$(N);
110 NEXT J
```

V riadkoch 60 a 70 sa zisťuje, či číslo J je párne alebo nepárne tak, že podiel J/2 sa porovná s celou časťou podielu J/2. Ak sú rovnaké, číslo J je párne.

Číselná šifra

Stači, ak do programu Šifra vsunieš riadok, ktorý prečíta šifrovacie číslo. V riadku 60 pripočítaj ku kódu ASCII písmena (X) šifrovacie číslo (K). Ak výsledný súčet je väčší než 90, odpočítaj od neho číslo 26.

```
25 PRINT "NAPÍŠ ŠIFROVACIE ČÍSLO"
27 INPUT K
60 LET N=X+K
70 Vymaž tento riadok.
```

Šifrovanie v cykle

Opäť stačí upraviť program Šifra. V riadku 60 pripočítaj ku kódu ASCII písmena hodnotu premennej cyklu J.

```
60 LET N=X+J
70 Vymaž tento riadok.
```

Dešifrovanie

Zmeň riadok 60:

```
60 LET N=X-J
```

Obrátená šifra

```
 10 INPUT "NAPÍŠ TAJNÚ SPRÁVU ";S$
 20 FOR J=1 TO LEN(S$) STEP 2
&30 PRINT MID$(S$,J+1,1);
&40 PRINT MID$(S$,J,1);
 50 NEXT J
```

V programe je cyklus s krokom 2. Pri každom opakovaní cyklu sa vezmú nasledujúce dva znaky správy. Najprv sa napíše druhý (J1) a potom prvý z nich (J).

Obrátená šifra na počítačoch Sinclair

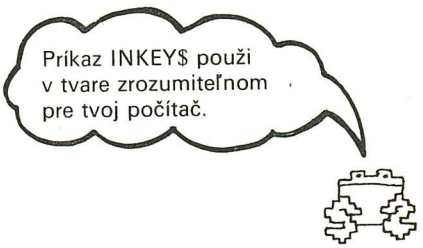
```
30 PRINT S$(J+1 TO J+1);
40 PRINT S$(J TO J);
```

Príkaz INKEY$ (strana 30)

Výpis AHOJ

```
10 LET A$=INKEY$
20 IF A$ = "" THEN PRINT " ";
30 IF A$ <> "" THEN PRINT "AHOJ";
40 GOTO 10
```

Ak nebol stlačený ani jeden kláves, reťazec A$ je prázdny a napíše sa medzera (riadok 20). Po stlačení klávesu sa príslušný znak uloží do A$ a napíše sa slovo AHOJ (riadok 30).



Ako prikázať počítaču, aby čakal

```
10 LET A$ = INKEY$
20 IF A$ = "" THEN GOTO 10
30 Zvyšok programu...
```

Riadok 20 sa bude opakovať dovtedy, kým nestlačíš ľubovoľný kláves.

Rýchla matematika

Ak chceš, aby program pracoval správne, oprav nasledujúce riadky.

`20 PRINT "HNEĎ AKO UVIDÍŠ SPRÁVNY SÚČET,"`	Chýbajú úvodzovky.
`50 LET X=INT(RND(1)*25+1)`	Náhodné číslo treba vynásobiť 25 a pripočítať 1.
`80 PRINT X;" + ";Y;" = "`	Odstráň úvodzovky pred X.
`110 LET A$=INKEY$`	Premenná musí byť na ľavej strane.
`140 IF N < 50 THEN GOTO 90`	Maximálny súčet čísel X a Y je 50, preto N musí rásť až po 50.
`160 FOR K=1 TO 1000:NEXT K`	Nesprávna premenná za NEXT.
`170 GOTO 40`	Zmeň číslo riadka.
`190 PRINT "SPRÁVNE: ODPOVEĎ JE ";X+Y`	Premenné nemôžu byť v úvodzovkách.

Program Autoškola (strana 31)

Úplný program hry Autoškola:

```
10  GCLEAR
20  LET A=5
30  LET L=1
40  LET S=10
50  LET P=L+S

60  IF L <= 1 THEN LET N=1
70  IF L >= 25 THEN LET N=0

80  IF N = 1 THEN LET L=L+1
90  IF N = 0 THEN LET L=L-1

100 LET A$=INKEY$

110 IF A$ = ">" THEN LET A=A+1
120 IF A$ = "<" THEN LET A=A-1
130 PRINT TAB(L);"!";TAB(A);"*";TAB(P);"!"
140 IF A <= L OR A >= P THEN PRINT "***HAVÁRIA***
150 GOTO 50
```

Ak L dosiahne niektorú zo strán obrazovky, zmení sa hodnota N.

AK N = 1, hodnota L rastie, ak N = 0, hodnota L klesá.

Príkazy umožnia hráčovi riadiť auto.

Nakreslí sa cesta.

Zisťuje sa, či nedošlo k havárii.

Návrat na riadok 50: nastavia sa nové hodnoty premenných P a L.

Ak program beží príliš rýchlo, vlož doň zdržiavací cyklus.

```
142 FOR K=1 TO 400:NEXT K
```

Zmeny pre počítač ZX81

Pridaj riadok: `125 SCROLL`

Pre tento počítač je výhodnejšie, ak spúšťaš program aj s bodovaním. Ak chceš, aby program po havárii ďalej pokračoval, napíš CONT.

Bodovací systém

```
140 IF A <= L OR A >= P THEN GOTO 160
145 IF A$ <> "" THEN LET B=B+1
160 PRINT "***HAVÁRIA***"
170 LET HH=HH+1
180 IF HH <= 5 THEN GOTO 20
190 PRINT "KONIEC HRY"
200 PRINT "ZÍSKAL SI ";B;" BODOV
210 STOP
```

Za každý manéver s autom, ktorý nespôsobí haváriu, získa hráč 1 bod (riadok 145). Po piatich haváriách sa program v riadku 180 zastaví. Premenným MH (počet havárií) a B (počet bodov) musíš na začiatku programu priradiť počiatočné hodnoty.

Náhodné kľukatenie cesty

```
75 IF L = 5 OR L = 10 THEN
   LET N=INT(RND(1)*2)
76 IF L = 15 OR L = 20 THEN
   LET N=INT(RND(1)*2)
```

Jeden zo spôsobov, ako kresliť rôzne cesty, je meniť hodnotu premennej N v závislosti od náhodne vybraného čísla. V programe sa mení hodnota N, ak L je rovné 5, 10, 15 a 20.

Používame DATA (strany 32–33)

Kontrola mena

```
50 IF X$ = "DANO" THEN PRINT "MENO NIE
   JE V ZOZNAME":STOP
80 DATA ERIKA,IVANA,ZUZKA,DANKA
90 DATA JURAJ,ANDREJ,DANO
```

Do riadkov DATA napíš ľubovoľný zoznam mien. Presvedč sa, či si mená navzájom oddelil čiarkami. Posledné meno zoznamu napíš do riadka 50. Ak na svojom počítači nemôžeš písať do jedného riadka viac príkazov, napíš príkaz STOP spolu s podmienkou IF/THEN do nového riadka.

Používame RESTORE

```
10 FOR J=65 TO 90
```
— kód ASCII

```
20 FOR I=1 TO 19
```
— počet údajov

```
&40 IF LEFT$(N$,1) = CHR$(J) THEN PRINT N$
```

Zmeny pre počítače Sinclair

```
40 IF N$(1 TO 1) = CHR$(J) THEN PRINT N$
```

Oprav chyby

1. V riadkoch DATA sú symboly – a /, preto musíš v riadku 20 použiť reťazcovú premennú (napr. N$).
2. Príkaz GOTO spôsobí, že počítač sa po prečítaní všetkých údajov z riadkov DATA bude usilovať čítať ešte ďalšie údaje. Vznikne chyba a počítač vypíše „OUT OF DATA AT LINE 50" alebo podobný text. Chybu odstrániš tak, že údaje budeš načítavať v cykle. Cyklus sa zopakuje toľkokrát, koľko je údajov. Iná možnosť je pomocou príkazu IF/THEN načítavať údaje dovtedy, kým sa nenačíta posledný z nich (pozri program Kontrola mena).

Reštaurácia U Jána

```
10  PRINT "VÍTAME VÁS V REŠTAURÁCII U JÁNA"
20  PRINT "KOĽKO MÁTE PEŇAZÍ?"
30  INPUT X
40  PRINT "MÔŽETE SI VYBRAŤ"
50  PRINT
60  READ Y,Y$
70  IF Y <= X THEN PRINT Y$
80  IF Y$ = "KARAMELOVÝ MLIEČNY KOKTAIL"
    THEN STOP
90  GOTO 60
100 DATA 12.20,FRANCÚZSKE ZEMIAKY
```

Ostatné jedlá z jedálneho lístka napíš do riadkov DATA. Ako posledný uveď karamelový mliečny koktail.

Telefónny zoznam

```
10  PRINT "AKO SA VOLÁ OSOBA,"
20  PRINT "KTOREJ CHCEŠ TELEFONOVAŤ?"
30  PRINT
40  INPUT N$
50  READ X$,Y$
60  IF N$ = X$ THEN GOTO 90
70  IF X$ = " " THEN PRINT "MENO NIE JE V ZOZNAME":GOTO 100
80  GOTO 50
90  PRINT X$;": ";Y$
100 PRINT "CHCEŠ ĎALŠIE TELEFÓNNE"
110 INPUT "ČÍSLO";A$
120 IF A$ = "ÁNO" THEN GOTO 150
130 IF A$ = "NIE" THEN STOP
140 PRINT "NEROZUMIEM":GOTO 110
150 RESTORE:GOTO 10
160 DATA MAXIPES FÍK,564 71
170 DATA PETER ROBOT,601 444
```

Meno sa uloží do premennej X$ a číslo do Y$.

Ak počítač nájde meno, pokračuje riadkom 90.

Do úvodzoviek napíš posledné meno zo svojho zoznamu.

Napíše, ak odpoveď nebola ani ÁNO, ani NIE.

Počítač znova začne prehľadávať zoznam.

Ako údaje napíš svoj telefónny zoznam.

Používame polia (strany 34–35)

Číselné polia

```
10 DIM N(6)
50 DATA 1066,1216,1485,1603,1665,1959
```

Výpis údajov

Údaje vypisuj v cykle. Nezabudni prečíslovať riadky DATA.

```
50 FOR K=1 TO 6
60 PRINT "N(";K;") JE ";N(K)
70 NEXT K
```

V tomto cykle sa prvky poľa vypisujú v náhodnom poradí.

```
50 FOR K=1 TO 10
60 LET R=INT(RND(1)*6+1)
70 PRINT N(R)
80 NEXT K
```

Reťazcové polia

Zmeny pre počítače Sinclair

```
 10 DIM M$(5,7)
@30 INPUT M$(I)
```

```
&10 DIM M$(5)
 20 FOR I=1 TO 5
@30 READ M$(I)
 40 NEXT I
 50 FOR I=1 TO 5
 60 PRINT "M$(";I;") JE ";M$(I)
 70 NEXT I
 80 DATA TÓNO,KAROL,JÚLIA,EVA,GABA
```

Počítač — kalendár

Neúplné riadky doplň takto:

```
&10 DIM M$(12),D(12)
@30 READ M$(K),D(K)
 70 PRINT M$(N);" MÁ ";
 80 PRINT D(N);" DNÍ"
```

Riadky DATA doplň takto:

```
@90 DATA JANUÁR,31
```

Ako zmeniť program:

```
 50 PRINT "KTORÝ MESIAC?"
 60 INPUT A$
 62 FOR I=1 TO 12
&63 IF M$(I) = A$ THEN LET F=I
 64 NEXT I
 70 PRINT A$;" MÁ ";
 80 PRINT D(F);" DNÍ"
```

Zmeny pre počítače Sinclair

```
 10 DIM M$(12,9)
 15 DIM D(12)
@30 INPUT M$(K)
@35 INPUT D(K)
 63 IF M$(I)(TO LEN(A$)) = A$ THEN
    LET F=I
```

Čo robiť?

```
&10  DIM I$(10)
 20  FOR K=1 TO 10
@30  READ I$(K)
 40  NEXT K
 50  PRINT "VYBER SI ČÍSLO OD 1 DO 10"
 60  INPUT N
 70  PRINT "A ČO TAKTO ";
 80  PRINT I$(N)
 90  PRINT "SÚHLASÍŠ ?";
 100 INPUT A$
 110 IF A$ = "ÁNO" THEN STOP
 120 PRINT: GOTO 50
 130 DATA "NAKRESLIŤ OBRAZ","PREČÍTAŤ SI
     KNIHU"
```

Doplň ďalšie údaje do riadka 130.

Zmeny pre počítače Sinclair

```
 10 DIM I$(10,?)
@30 INPUT I$(K)
```

Spočítaj počet znakov v najdlhšom z údajov a výsledok napíš do riadka 10.

Graf náhodných čísel

```
100 FOR K=1 TO 10
110 PRINT K;TAB(4);
120 FOR L=1 TO A(K)
130 PRINT "*";
140 NEXT L
150 PRINT
160 NEXT K
```

Podprogramy (strana 36)

Zmrzlinová anketa

```
&10  DIM I$(5),N(5)
 20  FOR K=1 TO 5
@30  READ I$(K),N(K)
 40  GOSUB 70
 50  NEXT K
 60  STOP
 70  PRINT I$(K);TAB(11);" ";
 80  FOR L=1 TO N(K)
 90  PRINT "*";
 100 NEXT L
 110 RETURN
 120 DATA ČUČORIEDKOVÁ,16,BANÁNOVÁ,14
 130 DATA PUNČOVÁ,8,PISTÁCIOVÁ,10
 140 DATA JAHODOVÁ,18
```

TAB v riadku 70 sme použili preto, aby sa hviezdičky písali vždy od toho istého stĺpca.

Zmeny pre počítače Sinclair

```
 10 DIM I$(5,12)
 15 DIM N(5)
@30 INPUT I$(K)
@35 INPUT N(K)
```

Ponorka

```
200 PRINT "NEUHÁDOL SI"
210 IF B = Y THEN GOTO 250
220 IF B < Y THEN PRINT "VIAC NA SEVER"
230 IF B > Y THEN PRINT "VIAC NA JUH"
240 IF A = X THEN GOTO 270
250 IF A < X THEN PRINT "VIAC NA VÝCHOD"
260 IF A > X THEN PRINT "VIAC NA ZÁPAD"
270 RETURN
```

V podprograme sa porovná poloha ponorky s tvojím odhadom a na obrazovku sa napíše správa.

Dvadsať hádaniek

```
&10  DIM H$(20),O$(20)
 20  FOR K=1 TO 20
@30  READ H$(K),O$(K)
 40  NEXT K
 50  FOR L=1 TO 20
 60  PRINT H$(L)
 70  INPUT X$
 80  IF X$ = O$(L) THEN PRINT "SPRÁVNE"
 90  IF X$ <> O$(L) THEN PRINT "HLÚPOSŤ.
     ";O$(L)
 100 PRINT
 110 NEXT L
 120 DATA "ČO RASTIE KOREŇOM
     HORE? ",ZUB
 130 DATA "DO AKÝCH SUDOV NEMOŽNO NALIAŤ
     VÍNO? ",DO PLNÝCH
```

Hádanky a správne odpovede napíš do riadkov DATA.

Zmeny pre počítače Sinclair

```
 10 DIM H$(20, ? )
 15 DIM O$(20)
@30 INPUT H$(K)
@35 INPUT O$(K)
```

Spočítaj počet znakov v najdlhšej hádanke a toto číslo napíš do riadka 10.

Ovocná lotéria (strana 37)

```
&10  DIM F$(6)
 20  FOR K=1 TO 6
@30  READ F$(K)
 40  NEXT K
 50  GCLEAR
 60  LET T=0
 70  PRINT "MÁŠ ";T;" ŽETÓNOV"
 80  LET I$ = INKEY$
 90  IF I$ = "" THEN GOTO 80
100  LET T=T-1
110  GCLEAR

120  LET R=INT(RND(1)*6+1)
130  LET A$=F$(R)
140  LET R=INT(RND(1)*6+1)
150  LET B$=F$(R)
160  LET R=INT(RND(1)*6+1)
170  LET C$=F$(R)

180  PRINT

190  PRINT A$;"  ";B$;"  ";C$

200  PRINT

210  IF A$ = B$ AND B$ = C$ AND
     C$ = "VIŠNE" THEN GOSUB 270
220  IF A$ = B$ AND B$ = C$ AND
     C$ <> "VIŠNE" THEN GOSUB 310
230  IF (A$ = B$ AND B$ <> C$) OR
     (A$ = C$ AND B$ <> C) OR (B$ = C$
     AND C$ <> A$) THEN GOSUB 350

240  IF T > 0 THEN GOTO 70
250  PRINT "NEMÁŠ ŽIADEN ŽETÓN"
260  STOP

270  PRINT "3 VIŠNE"
280  PRINT "HLAVNÁ VÝHRA"
290  LET T=T+20
300  RETURN

310  PRINT "3 ROVNAKÉ"
320  PRINT "VÝHRA 5 ŽETÓNOV"
330  LET T=T+5
340  RETURN

350  PRINT "2 ROVNAKÉ"
360  PRINT "VÝHRA 2 ŽETÓNY"
370  LET T=T+2
380  RETURN

390  DATA CITRÓN,VIŠNE,MELÓN
400  DATA ZVONČEK,HROZNO,SLIVKA
```

Počítač vyberie 3 druhy ovocia.

Napíše ich.

Zistí typ výhry.

1. podprogram

2. podprogram

3. podprogram

Zmeny pre počítače Sinclair

```
 10 DIM F$(6,7)
@20 INPUT F$(K)
```

Hlavolam Robot-bežec (strana 52)

```
 10 INPUT "AKÁ JE TEPLOTA ";TEP
 20 INPUT "KOĽKO SEKÚND POBEŽÍ ";S
@30 IF TEP <> 25 THEN LET M=S*(500+10*(TEP-25))
    ELSE LET M=500*S
 40 IF M < 1 THEN PRINT "PRE ZAKA JE PRÍLIŠ
    CHLADNO"
 50 PRINT "PRI TEPLOTE ";S;" ZABEHNE ZAK ZA ";
    S;" SEKÚND ";M;" METROV"
```

Pre počítače Sinclair zmeň riadok 30 takto:

```
30 IF TEP <> 25 THEN LET
   M=S*(500+10*(TEP-25))
35 IF TEP = 25 THEN LET M=500*S
```

Počítač najprv vypočíta súčin 10*(TEP-25), ktorý hovorí, o koľko sa zmení Zakova rýchlosť v závislosti od teploty. Ak k výsledku pripočíta 500, dostane vzdialenosť, ktorú Zak ubehne za sekundu. Vynásobením počtom sekúnd zistí dĺžku ubehnutej dráhy.

Shellovo triedenie (strana 62)

```
90  LET X=0
95 LET VYM=0
231 LET X=X+1
271 LET VYM=VYM+1
365 PRINT "UROBIL SOM ";X;" POROVNANÍ
    A ";VYM;" VÝMEN"
```

Hrubšie stĺpce (strana 65)

Čísla uprav tak, aby boli vhodné pre tvoj počítač.

```
285 FOR J=1 TO 8 STEP 2
290 PLOT INT(I*X+J),1
300 DRAW INT(I*X+J),INT(B(I)*Y)
310 PLOT INT(I*X-4-J),1
320 DRAW INT(I*X-4-J),INT(S(I)*Y)
325 NEXT J
```

Pokračovanie odpovedí na otázky zo strojového kódu

Strana 108

00011010 je desiatkovo 26.
11111011 je desiatkovo 251.
10101010 je desiatkovo 170.

	desiatkovo		hex	
	vyšší bajt	nižší bajt	vyšší bajt	nižší bajt
307	1	51	&01	&33
21214	82	222	&52	&DE
759	2	247	&02	&F7
1023	3	255	&03	&FF

Strana 111

Program zo strany 107 upravíš pre veľké čísla tak, že v ňom zrušíš príkaz návratu a pridáš tieto riadky. Výsledok si môžeš pozrieť príkazom PRINT PEEK (adresa 3) +PEEK (adresa 4) *256

Z80		6502		význam
skratky	hex kódy	skratky	hex kódy	
LDA A,&00	3E 00	LDA #&00	A9 00	Daj 0 do akumulátora.
ADC A,&00	CE 00	ADC #&00	69 00	Pripočítaj 0 s prenosom.
LD (adresa 4),A	32 adresa 4	STA adresa 4	8D adresa 4	Ulož akumulátor do pamäte.
RET	C9	RTS	60	Návrat.

Strana 119

Šestnástkový zápis dvojkového doplnku čísla 12 je &F4, čísla 18 je &EE a čísla 9 je &F7.

Zmeny v šestnástkovom ukladači

Pre ZX81 zmeň tieto riadky:

```
 40 INPUT H$
 70 LET X=(CODE(H$)-28)*16
 80 zrušiť
 90 LET Y=CODE(H$(2 TO))-28
100 LET X=X+Y
110 zrušiť
155 zrušiť
160 zrušiť
```

Pre počítače Atari zmeň riadok:

```
90 LET Y=ASC(A$(2))
```

Zhrnutie basicu

V tejto časti zhrnieme príkazy basicu vysvetlené v knihe. Príkazom, ktoré sú označené hviezdičkou, nerozumejú všetky počítače. Musíš ich zmeniť podľa tabuľky na strane 147 a podľa príručky k svojmu počítaču.

ABS odtrhne znamienko pred číslom a nechá jeho absolútnu hodnotu. Napríklad ABS(−5) je 5.

***ASC** zmení znak na jeho číselný kód podľa tabuľky ASCII, v ktorej každému znaku zodpovedá číslo. Napríklad ASC("A") je 65, čo je číslo prislúchajúce znaku A. Na počítačoch Sinclair sa namiesto ASC používa CODE.

**CHR$** mení podľa tabuľky čísla na písmená. Na počítačoch, ktoré používajú tabuľku ASCII, je CHR$(65) znak A. Na počítačoch ZX81 sa používa iná tabuľka, znak A je CHR$(38).

CLS zmaže všetko na obrazovke.

***CODE** sa používa na počítačoch Sinclair namiesto ASC na prevod písmen na ich číselný kód. Počítače Spectrum používajú na prevod tabuľku ASCII, počítače ZX81 majú vlastnú tabuľku.

***DATA** pozri READ/DATA.

***DIM** oznamuje počítaču, koľko pamäte má vyhradiť pre pole. Za DIM nasleduje meno poľa a počet jeho prvkov. Napríklad DIM A(5) vytvorí miesto pre päť čísel a označí ho A. Na počítačoch Sinclair musíš pri deklarácii reťazcového poľa uviesť aj počet znakov najdlhšieho reťazca.

FOR/TO...NEXT znamená, že počítač má určený počet krát opakovať príkazy medzi FOR/TO a NEXT. Tento príkaz sa nazýva cyklus.

GOSUB opustí hlavný program a vykoná podprogram. Za GOSUB musíš napísať číslo prvého riadka podprogramu. Na konci podprogramu musí byť príkaz RETURN, ktorý pošle počítač späť do hlavného programu — na príkaz za GOSUB.

GOTO skočí na riadok, ktorého číslo je za GOTO.

IF/THEN za IF nasleduje podmienka a počítač testuje, či bola splnená. Ak áno, vykoná príkaz za THEN, ak nie, príkaz nevykoná.

***INKEY$** zisťuje, či je stlačený kláves, a ak áno, ktorý. Na rozdiel od príkazu INPUT nečaká, kým kláves stlačíš. Pretože počítač je veľmi rýchly, INKEY$ sa zvyčajne používa v cykle. Na niektorých počítačoch môžeš za INKEY$ napísať číslo: oznámi počítaču, ako dlho má čakať na stlačenie klávesu.

INPUT označí miesto v pamäti, napíše otáznik (PMD-85-2 len zapípa) a čaká, kým napíšeš niečo, čo môže uložiť na označené miesto. Za INPUT musíš napísať meno premennej.

INT z čísla odsekne všetko, čo sa nachádza vpravo od desatinnej bodky. Napríklad INT(6.732) je 6. Pri zápornom čísle sa k výsledku ešte pripočíta jednotka, teda INT(−3.2) je −4.

***LEFT$** znamená, že počítač má vziať určený počet znakov zo začiatku reťazca. Napríklad LEFT$(A$,4)

vezme prvé štyri znaky reťazca A$. Počítače Sinclair tento príkaz nepoznajú.

LEN zistí dĺžku reťazca, teda počet znakov, z ktorých sa reťazec skladá (vrátane medzier a interpunkčných znamienok).

LET vyhradí miesto v pamäti a uloží doň hodnotu.

LIST zobrazí na obrazovku program.

***MID$** znamená, že počítač má vziať určený počet znakov z reťazca. Napríklad MID$(A$,5,2) prikáže vziať z reťazca A$ dva znaky počnúc piatym. Počítače Sinclair nepoznajú tento príkaz.

NEXT označuje koniec cyklu. Pozri FOR/TO... NEXT.

NEW vymaže program.

PRINT napíše niečo na obrazovku. Samotné PRINT urobí prázdny riadok.

***READ/DATA** príkaz READ vezme údaj napísaný za slovom DATA a uloží ho do premennej napísanej za READ.

REM uvádza riadok s komentárom. V komentári sa zvyčajne vysvetľuje úloha jednotlivých častí programu.

RESTORE znamená, že pri čítaní údajov za DATA sa má začať od prvého údaja.

RETURN sa používa na ukončenie podprogramu, pošle počítač späť do hlavného programu — na príkaz za GOSUB.

***RIGHT$** znamená, že počítač má vziať určený počet znakov z konca reťazca. Napríklad RIGHT$(A$,4) prikáže vziať posledné štyri znaky reťazca A$. Počítače Sinclair nepoznajú tento príkaz.

***RND** náhodne vyberie číslo.

RUN vykoná príkazy programu.

***SPC** napíše niekoľko medzier pozdĺž obrazovky. Niektoré počítače tento príkaz nepoznajú.

STEP sa používa spolu s FOR/TO... NEXT. Určuje, o koľko treba meniť hodnotu premennej cyklu.

***TAB** posunie kurzor na obrazovke o určený počet medzier doprava. Obyčajne sa používa spolu s príkazom PRINT, ak treba napísať niečo do stredu obrazovky. Na niektorých počítačoch sa dá pomocou príkazu TAB posúvať kurzor aj smerom dolu. Na väčšine počítačov musíš za TAB napísať bodkočiarku.

THEN sa používa s IF. Určuje, čo sa má vykonať, ak je splnená podmienka za IF. Pozri IF/THEN.

Symboly basicu

*	násobenie
/	delenie
SQR	druhá odmocnina
^	umocňovanie
>	väčší než
<	menší než
>=	väčší alebo rovný
<=	menší alebo rovný
<>	nerovnosť

Tabuľka rozdielov basicu

Príkaz použitý v tejto knihe	GCLEAR vymaže obrazovku	INT(RND(1)*N+1) vyberie náhodné číslo od 1 do N	X$=INKEY$ prečíta stlačený kláves
PMD-85	GCLEAR	INT(RND(1)*N+1)	X=INKEY (len kĺúče)
PP01	CLEAR	TRUNC(RND*N+1)	X=KEY
Commodore 64	PRINT CHR$(147)	INT(RND(1)*N+1)	GET X$
Atari 130XE	PRINT CHR$(125)	INT(RND(1)*N+1)	X=PEEK(764)
ZX Spectrum	CLS	INT(RND*N+1)	X$=INKEY$
ZX81	CLS	INT(RND*N+1)	X$=INKEY$

Ďalšie rozdiely na počítačoch Sinclair

LEFT$ RIGHT$ MID$	Počítače Sinclair (ZX81 a ZX Spectrum) narábajú s reťazcami inak než ostatné počítače. Pri prepisovaní príkazov vľavo musíš zistiť číslo prvého a posledného znaku, ktoré chceš vybrať, napísať čísla do zátvoriek za meno reťazcovej premennej a oddeliť ich slovom TO. Ozrejmia ti to ukážky vpravo.	LET X$="LAMALIMA" PRINT X$(1 TO 4) LAMA PRINT X$(5 TO) LIMA PRINT X$(3 TO 6) MALI
ASC	Namiesto ASC musíš použiť CODE. ZX81 má vlastnú tabuľku kódov. ZX Spectrum používa tabuľku ASCII.	PRINT CODE("A")
DIM	Pri dimenzovaní reťazcového poľa musíš v príkaze DIM napísať za menom poľa dve čísla v zátvorkách. Prvé je počet položiek v poli, druhé je počet znakov najdlhšieho reťazca, ktorý chceš do poľa uložiť. V našom príklade pole A$ má päť reťazcov s počtom znakov najviac 12.	DIM A$(5,12)
READ/DATA	ZX Spectrum má príkazy READ a DATA také isté ako ostatné počítače. ZX81 podobné príkazy nemá, preto ich musíš nahradiť príkazom INPUT.	10 FOR K=1 TO 5 20 INPUT A$(K) 30 NEXT K

Zmeny v programoch na stranách 42–71

Na tejto strane sa dozvieš, ako zmeniť programy zo strán 42 až 71 tak, aby bežali na počítačoch Sinclair (teda ZX81, ZX Spectrum, ZX Spectrum+, Didaktik gama a pod.). Rady pre úpravu kresliacich programov platia pre všetky počítače, na ktorých sa pri kreslení určujú len zmeny súradníc oproti bodu, v ktorom sa nachádzalo pero. Okrem zmien, ktoré nájdeš na tejto strane, musíš použiť v programe správny tvar funkcie RND a grafických príkazov.

Hra so slovami pre počítače Sinclair

Zmeň meno premennej OVER$ na O$ (alebo radšej V$, lebo písmeno O si ľahko zmýliš s číslicou 0) a zmeň riadky 170 a 200:

```
170 LET P$=S$(R TO R)
200 LET V$=V$(2 TO)+P$
```

Databáza pre ZX81

```
10  DIM T$(10,9): DIM Y(12)
15  DIM M(10,12)
120
417
425
```

Rovnaké ako pre ZX81

Nezabudni všetky reťazce v riadkoch DATA dať do úvodzoviek.

Databáza pre ZX Spectrum

```
10   DIM T$(10,9)
12   DIM Y(12)
14   DIM M$(10,12)
```

Druhý index je dĺžka najdlhšieho reťazca.

```
120  GOSUB 100+100*C
```

Premennú C používa počítač na vypočítanie riadka.

```
417  LET L=LEN(Z$)
425  IF Z$ = T$(I) (1 TO L) THEN GOTO 440
```

Počítač porovná reťazce.

```
455  IF VAL(M$(I,J)) = 1 THEN PRINT
     "MAJSTER SVETA ROKU ";Y(J)
460  IF VAL(M$(I,J)) = 2 THEN PRINT
     "FINALISTA ROKU ";Y(J)
555  IF VAL(M$(I,J)) = 1 THEN PRINT T$(J)
```

Údaje pre maticu sú uložené v reťazcovom poli. Pomocou príkazu VAL ich počítač prevedie na čísla.

```
2170 IF C > 0 AND C<6 THEN GOTO 2180
2175 PRINT "NAPÍŠ ČÍSLO OD 1 DO 5"
2176 GOTO 2150
```

V riadkoch 1000 až 1050 vypočíta, v ktorých rokoch sa konali majstrovstvá sveta.

```
1000 FOR I=1 TO 3
1010 LET Y(I)=1926+I*4
1020 NEXT I
1030 FOR I=0 TO 8
1040 LET Y(I+4)=1950+4*I
1050 NEXT I
```

Riadky 1100 až 1120 nahraď desiatimi riadkami s príkazmi LET.

```
1100 LET T$(1)="URUGUAJ"
1110 LET T$(2)="ARGENTÍNA"
```

Riadky 1200 až 1310 nahraď ďalšími desiatimi riadkami s príkazmi LET.

```
1200 LET M$(1)="100100000000"
1210 LET M$(2)="200000000010"
```

Vsuň nový riadok.

```
1310 RETURN
```

Jednoduchá grafika a kreslenie grafov

Niektoré počítače (napríklad ZX spectrum) pri kreslení merajú vzdialenosť X a Y nie od rohu obrazovky, ale od posledne nakresleného bodu. V takom prípade musíš zmeniť program

Jednoduchá grafika

```
175 DRAW CX(1)-CX(4),CY(1)-CY(4)
180 FOR I=2 TO 4
185 DRAW CX(I)-CX(I-1),CY(I)-CY(I-1)
530 DRAW LX(RI+2,I)-LX(RI,I),
        LY(RI+2,I)-LY(RI,I)
```

Súradnice bodu vypočíta počítač tak, že odčíta súradnice posledneho nakresleného bodu.

Jednoduchá grafika (na strane 55) a programy Stĺpcový diagram a Graf (na stranách 64 a 65). Slová DRAW a PLOT musíš možno nahradiť vhodnými slovami pre tvoj počítač.

Program Stĺpcový diagram

```
220 PLOT 1,H:DRAW 0,-V+1:DRAW S,0
280 DRAW 0,INT(B(I)*Y)
300 DRAW 0,INT(S(I)*Y)
```

Program Graf

```
290 DRAW X,INT((B(N)-B(N-1))*Y)
330 DRAW X,INT((S(N)-S(N-1))*Y)
```

Konverzačný program pre ZX Spectrum

Pre Spectrum spravte tieto zmeny v programe:

```
110 DIM V$(10,11):DIM N$(10,16):DIM A$(10,11)
120 DIM T$(10,14):DIM S$(10,14)
130 DIM M$(30,29):DIM Q$(30,7):DIM C(30)
245 PRINT I$
500 FOR Q=1 TO 30
510 LET P$=""
512 FOR I=1 TO LEN (Q$(Q))
514 IF Q$(Q)(I TO I)<>" " THEN
    LET P$=P$+Q$(Q)(I TO I)
516 NEXT I
530 FOR T=1 TO L2-LEN(P$)+1
540 IF I$(T TO T+LEN(P$)-1)=P$ THEN
    GOTO 560
550 NEXT T:NEXT Q:GOTO 600
560
570 - Zmen premennu SLOVO na Q
580
670 GOTO 680+E*20
```

Namiesto SLOVO musíme použiť Q

Do P$ dá frázu z Q$

Zisťuje, či sa I$ rovná P$

Konverzačný program pre ZX81

Pre ZX81 potrebuješ zmeniť spôsob zadávania údajov do poľa Q$ (ZX81 nepozná READ a DATA). Jedna možnosť je použiť príkazy INPUT. Program zmeň takto:

1. Urob rovnaké zmeny ako pre ZX Spectrum, len príkazy DIM napíš do riadkov 970 až 990.
2. Zaměň príkazy READ a DATA príkazmi INPUT napríklad takto:

```
1000 REM UDAJE PRE PODPROGRAM
     ROZUMNA ODPOVED
1010 FOR I=1 TO 30
1020 INPUT Q$(I)
1030 NEXT I
```

3. Zmeň riadok 1720:

```
1720 STOP
```

4. Napíš program do počítača, potom napíš RUN 970, a potom zapíš všetky údaje, ktoré počítač postupne vyžiada.

5. Program potom spustiš GOTO 100. Nestláčaj RUN, lebo by sa ti stratili všetky údaje.
6. Teraz môžeš program nahrať na kazetu. Keď ho znova nahráš, spustiš ho GOTO 100.

Register

V registri uvádzame slová basicu veľkými písmenami.

adresovanie s priamym operandom, 98, 107, 113, 130
adresovanie v nultej stránke, 130, 127
adresovanie, 88–89, 99, 126
akumulátor, 94–95, 97, 30, 112, 130
ALJ (aritmeticko-logická jednotka), 94, 95, 130
AND, 22, 52
ASCII kód, 28, 93, 104, 112
ASC, 28, 145, 147
assembler, 85, 96, 120, 130
assemblerový jazyk, 85, 97, 99
bajt, 84, 93, 99, 100, 126
bit, 84, 126
bod obrazu, 8
bodkočiarky, 5, 6
BREAK, 7
bublinové triedenie, 58–59, 60, 63
CHR$, 28, 145
chyby, 4, 6, 7, 41, 51
cieľový program, 98, 130
CLS, 140
CODE, 28, 145, 147
CONTINUE, 41
CPU (riadiaca jednotka), 94–95
cyklus, 8, 40, 52
čiarka, 5, 7, 11, 51, 60, 71
číselné premenné, 6, 10, 50
čísla riadkov, 4, 6, 40
DATA, 7, 11, 32, 145
deklarovanie polí, 90, 67
delenie, 5
DELETE, 14
desiatkové čísla, 91, 121
DIM, 9, 11, 34, 53, 145, 147
disassembler, 126
doplnok, 126
dvojbodka, 11
dvojkové čísla, 99, 108
dvojkový doplnok, 119, 121, 130
dvojkový kód, 99, 108
dvojrozmerné pole, 9, 51
ELSE, 51, 52
END, 51
ENTER, 4, 14
EOL, 4, 14
GCLEAR, 4, 140
GET, 30, 14ý
GOSUB, 7, 36, 48, 145
GOTO, 7, 22, 145
grafické príkazy, 8, 11, 54
grafy, 64–65
IF/THEN, 7, 22–23, 52, 145
implicitné adresovanie, 130
indexované adresovanie, 114, 119, 130
indexový register, 94–95, 130
indexy, 9, 34, 47
inštrukcia návratu (v strojovom kóde), 103, 115
kód operácie, 96, 98, 99
kódy znakov, 93, 112
kurzor, 4, 14
ladenie, 6, 41
LEFT$, 9, 26, 138, 145
LEN, 9, 26, 146
LET, 6, 10, 16, 146
LIST, 4, 15, 146
mapa pamäte, 88
matematické symboly, 14, 139
matice, 46
MEMEND, 100, 101
meno premennej, 6, 10
menu, 46, 48, 52
MID$, 9, 26, 42, 146
mikroprocesor, 87, 96, 127
náhodné čísla, 8, 35, 63
násobenie, 5
nepriame adresovanie (Z80), 113, 118, 131
NEW, 14, 146
nižší bajt, 99, 108, 110–111, 130
nultá stránka, 90, 131
odčitovanie, 5
oddeľovač, 60
odmocnina, 5
ON 48, 49, 69
operačný systém, 88, 90, 93, 100, 131
operand, 96, 98, 107, 127
opravovanie chýb, 14–15
OR, 22, 52
pamäť obrazovky, 88, 89, 93, 131
pamäťové miesta, 88–89, 90, 92–93
pamäť, 88–89, 90, 92–93
PEEK, 92–93, 101, 106, 111
PLOT, 54, 55, 64
počiatočné hodnoty premenných, 10
počítač Apple II, 63
počítač Atari 130XE, 3, 147
počítač Atari 800XL, 3
počítač Commodore, 3, 63, 87, 147
počítač PMD-85, 64, 147
počítač PP01, 147
počítač ZX Spectrum, 3, 54, 83, 93, 102, 147
počítač ZX81, 3, 16, 28, 34, 83, 89, 93, 102, 147
počítače Sinclair, 9, 10, 11, 26, 28, 34, 147
počítadlo adries, 94–95, 115, 131
podmienené skoky, 115, 116–117, 131
podmienka nuly, 112, 113, 114, 116, 131
podmienka parity (pretečenia), 116
podmienka prenosu, 94, 95, 97, 109, 110, 111, 116, 131
podmienka pretečenia, 116
podmienka znamienka, 94, 116, 131
podmienková premenná, 77
podprogram Opakovanie odpovede, 72
podprogramy, 7, 40, 48
prodprogramy v strojovom kóde, 115
POKE, 92–93, 103
polia, 9, 34–35, 47, 53, 54
porovnávanie, 112
posunutie, 116–117, 131
používateľova RAM, 88, 100, 131
premenná, 5, 6, 8, 9, 10, 40, 43
prevod adries medzi desiatkovou a šestnástkovou sústavou, 91
prevod zo šestnástkovej sústavy do desiatkovej, 91, 124
priame adresovanie, 107, 131
priame príkazy, 4, 14–15

PRINT, 5, 14–15, 146
program Autoškola, 31, 140
program Banka údajov, 46—53
program Blikajúca správa, 20, 134
program Číselná postupnosť, 24, 136
program Číselná šifra, 29, 138
program Čo robiť?, 35, 142
program Dostihy, 23, 135
program Dvadsať hádaniek, 35, 142
program Dvojkové počítadlo, 20, 134
program Graf náhodných čísel, 35, 142
program Hádanie čísel a slov, 23, 135
program Hádanie slova, 23, 135
program Heslo 22, 135
program Hovoríte po uglijsky?, 17, 133
program Hra so slovami, 42
program Chyby v cykloch, 20, 134
program Kontrola mena, 32, 140
program Malé — veľké, 28, 138
program Najdlhšie slovo, 27, 137
program Najkratšie slovo, 27, 137
program Násobilka, 22, 135
program Obrátená šifra, 29, 139
program Obrázkové hlavolamy, 19, 134
program Odlet z planéty Zorgos, 25, 136
program Ovocná lotéria, 37, 143
program Papier, kameň alebo nožničky, 25, 137
program Počítač — kalendár, 35, 142
program Počítač — kalkulačka, 22, 135
program Počítačové hodiny, 20, 134
program Pohľad do programu, 17, 133
program Poklad, 74–79, 80–81
program Ponorka, 36, 142
program Porovnávanie písmen, 28, 138
program Reštaurácia u Jána, 33, 149
program Rýchla matematika, 30, 139
program Safaládky, 17, 133
program Shellovo triedenie, 58, 61–62, 63, 143
program Skákajúci panáčik, 21, 134
program Stĺpcový diagram, 64–65
program Stupne Celzia a Fahrenheita, 17, 133
program Šifra, 29, 138
program Šifrovanie v cykle, 29, 139
program Štart rakety, 21, 134
program Tajné posolstvo, 19, 134
program Telefónny zoznam, 33, 141
program Výpočet rýchlosti, 17, 133
program Zmrzlinová anketa, 36, 142
prvky poľa, 34
RAM (pamäť s ľubovoľným prístupom), 86, 92, 93, 100–101
RAMTOP, 88, 100, 101, 131
RAMTOP — znižovanie, 100–102
READ/DATA, 7, 9, 32, 139, 140
register podmienok, 94–95
registre, 94–95, 107, 110, 111, 131
relatívne adresovanie, 116, 131
REM 7, 40
REM — ukladanie bajtov strojového kódu, 102
RESTORE, 32, 146
RETURN, 4, 7, 14–15, 146
reťazec — narábanie s reťazcami, 42, 66
reťazec — reťazcové premenné, 10, 16–17, 50, 60
riadky obsahujúce viac príkazov, 11, 43, 44
RIGHT$, 9, 26, 42, 146
RND, 8, 24, 146
ROM, 86, 92, 93
RUN, 4, 15, 146
sčitovanie, 5
sčitovanie v strojovom kóde, 97, 107, 110
skoky (v strojovom kóde), 113, 115, 131
skratky (v strojovom kóde), 85, 96–97, 125–129
SPC, 14, 139
spôsoby adresovania, 107, 125–129
spustenie strojového programu, 105
stavový register procesora, 95, 109, 131
STEP, 8
STOP, 41
stránka (pamäte), 90, 91, 99, 101, 131
strojový kód — ako písať programy, 106
strojový kód — dĺžka programu, 100
strojový kód — kam ho uložiť do pamäte, 100–102
strojový kód — podprogramy, 119, 120
súradnice, 8, 54, 64
systémové premenné, 90, 100, 101, 127
šestnástková sústava (hex), 85, 88, 91, 126
šestnástkové kódy, 96, 98, 99
šestnástkový ukladač, 85, 104, 126
šestnástkový výpis (dump), 99
štandardný basic, 10, 41
TAB, 14, 139
triedenie údajov, 58–63
údajové bajty, 103, 108
ukazovateľ zásobníka, 94–95, 127
umiestnenie v stránke (pamäťového miesta), 91, 99, 101
úvodzovky, 5, 11
VAL, 50
vetvenie, 22
vetvenie v strojovom kóde, 114, 116
vnorené cykly, 9, 20, 44, 45, 51
vyhradené pre operačný systém, 89
vyrovnávacia pamäť, 126
vyšší bajt, 99, 108, 110, 111, 132

vývojový diagram, 75
zásobník (LIFO), 90, 94, 95, 100, 116, 132
zátvorky, 5
zdrojový program, 98, 132
zdržiavací cyklus, 19, 43, 44
znak krížik #, 92, 96, 98, 130
znaky, 4, 60

SKÚSIŠ PROGRAMOVAŤ?
(basic a strojový kód)

Lisa Wattsová a Gaby Waters
Ilustrovali Graham Round a Naomi Reed.
Odborní poradcovia pre basic B. Reffin Smith, N. Cutler a M. Wharton.
Odborní poradcovia pre strojový kód A. P. Stephenson a C. Oxlade.

Z anglického originálu (The Beginner's Programming Handbook — In Basic & Machine Code, Usborne Publishing Ltd., London 1984) preložili Ivana Štefáneková a Peter Tomcsányi.

Vydali Mladé letá, Bratislava, ako svoju 7211. publikáciu, v roku 1991. Pre čitateľov od 13 rokov. Vydanie prvé. Zodpovedná redaktorka Jana Steinerová. Výtvarný redaktor Martin Kellenberger. Technická redaktorka Irena Závodná. Jazyková redaktorka Albína Gardošová. AH 21,90 (text 10,00, ilustr. 11,90). VH 22,18. Sadzbu vyhotovili Západoslovenské tlačiarne, z. p., závod Svornosť, Bratislava. Vytlačili Východoslovenské tlačiarne, š. p., Košice.

ISBN 80-06-00178-2

SKÚSIŠ PROGRAMOVAŤ?

BASIC A STROJOVÝ KÓD

Kniha Skúsiš programovať? voľne nadväzuje na knihu Skúsiš to s mikropočítačom?. Dozvieš sa v nej o niečo viac ako v predchádzajúcej o programovaní v basicu, najbežnejšom programovacom jazyku pre mikropočítače, a základné informácie o programovaní v strojovom kóde, ktorý je síce zložitý, ale niekedy vhodnejší ako basic — hlavne vtedy, keď chceš na obrazovke vytvoriť nejaké efekty alebo zrýchliť hru.

Kniha plná farebných ilustrácií je rozdelená do troch samostatných častí. V prvej nájdeš cvičenia, hlavolamy i krátke programy, ktoré ti pomôžu pri pochopení príkazov a základných pravidiel programovania v basicu. Druhá časť obsahuje niekoľko hotových zložitejších programov s podrobným vysvetlením. Vytvárajú napríklad banky údajov, kreslia grafy a obrázky, triedia údaje. Pri ich čítaní a spúšťaní pochopíš, ako sa robia takéto náročnejšie programy. Tretia časť ťa pomaly, doslova krok za krokom naučí rozumieť programovaniu v strojovom kóde. Ukáže ti, ako písať jednoduché programy, napríklad na rozblikanie správy na obrazovke.

Programy, ktoré sú v knihe, môžeš spustiť na počítačoch PMD-85, PP01, ZX81, ZX-Spectrum. Ak máš prístup k inému počítaču, nájdeš v knihe rady, ako si programy preň upraviť.

ISBN 80—06—00178—2