FLOP

ATARI BASIC

Uživatelská příručka

FLOP ROŽNOV 1991

MANUÁLY PROGRAMŮ ATARI XL/XE

OBSAH

Č Á S T I

ZÁKLADY PROGRAMOVÁNÍ V ATARI BASICU

ÚVOD

Váš počítač ATARI má zabudovaný ATARI BASIC, verzi jednoho z nejpopulárnějších programovacích jazyků. BASIC (Beginner's Allpurpose Symbolic Instruction Code - mnohaúčelový kód symbolických instrukcí pro začátečníky) byl vyvinut v USA na vysoké škole v Dartmouth v roce 1960 za účelem výuky programování začínajících vysokoškoláků. Od té doby se stal nejobvyklejším programovacím jazykem pro domácí počítače.

Ačkoliv je BASIC jednoduchý jazyk, je každá jeho verze mírně odlišná. ATARI BASIC má některé důležité specifické vlastnosti. Například některá slova ATARI BASICu vytvářejí snadno zvuky a barevnou grafiku. ATARI BASIC je speciálně určen pro začínající programátory. Na rozdíl od mnoha verzí BASICů kontroluje ATARI BASIC každý psaný programový řádek a hlásí, když jste udělali v instrukci chybu. A ovšem, když se učíte programovat v ATARI BASICu, snadněji se pak naučíte i jiné verzi BASICu.

ATARI BASIC je dostupný hned po zapnutí počítače. Nechcete-li používat BASIC držte při zapínání počítače stlačenou klávesu (OPTION). Jiný způsob opuštění BASICu je napsání příkazu BYE, který aktivuje samočinný test, nebo DOS, který aktivuje Diskový Operační Systém (je-li ovšem počítač připojen k disketové jednotce).

Druhou část této příručky, "Programování v jazyce ATARI BASIC", tvoří jednoduchá výuka pro ty, kteří s BASICem pracují poprvé. Na rozdíl od většiny výukových systémů, kde nejprve studujete skladbu jazyka, umožňuje Vám tento přístup začít ihned psát programy a slovní hry, řešit matematické problémy a využívat zvukových a grafických schopností počítače ATARI. Po prostudování této lekce budete mnohem lépe rozumět funkci kláves a činnosti počítače. Výukový systém Vám pomůže dostat z Vašeho počítače ATARI maximum.

Příprava pro práci s klávesnicí

Před tím, než začnete programovat v jazyce ATARI BASIC, musíte se seznámit se zvláštnostmi klávesnice počítače ATARI 130XE (800XL).

Funkce automatického opakování

Začněte napsáním písmene A:

A

Pokračujte držením stlačené klávesy /A/ a sledujte, jak narůstá řada písmen A. Když je řádek naplněn, kurzor automaticky skočí na další řádek. Není potřeba používat /RETURN/.

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Právě jste použili jednu z vnitřních funkcí klávesnice ATARI - funkci automatického opakování. Tuto funkci má většina kláves, včetně mezerníku. Slyšeli jste zvukový signál při dokončování třetího řádku? Toto varující bzučení - rovněž jedna z vnitřních funkcí ATARI BASICu - Vás upozorňuje, že instrukce se stává příliš dlouhou. Řádek s programovou instrukcí nemůže být delší než tři řádky na obrazovce.

Chybová hlášení

Najděte na klávesnici klávesu (RETURN) a stlačte ji. Na obrazovce se objeví slovo ERROR (chyba), následované třemi řadami A, jak jste je napsali. Nyní s Vámi počítač konverzuje. Oznamuje Vám, že nerozumí tomu, co jste mu napsali, protože řady A nejsou součástí programovacího jazyka ATARI BASIC. V dalších postupech nestlačujte klávesu (RETURN) dříve, než k tomu budete vybídnuti.

Velká a malá písmena

Stlačíte-li klávesu (CAPS) a potom podržíte klávesu (A), vytiskne se na obrazovce řada malých písmen A:

aaaaaaaaaaaa

Návrat k velkým písmenům provedete opětným stlačením klávesy (CAPS). Stlačíte-li pak opět klavesu (A) a chvíli ji podržíte, vytiskne se na obrazovce řada velkých A:

AAAAAAAAAAAA

Zkuste napsat slovo - slovo začínající písmenem A, jako například ATARI. Napište následující slova a přitom přepínejte klávesou (CAPS) mezi velkými a malými písmeny:

ATARI atari

Číslice se zobrazují stejné, ať píšete velká, nebo malá písmena. Narozdíl od psacího stroje má počítač dvě různé klávesy pro ovládání přechodu na velká písmena a pro přeřaďovač (Shift). Píšete-li velká nebo malá písmena, vždy se na obrazovce objeví symbol, uvedený ve spodní části označení klávesy. Zobrazení symbolu, který se nalézá v horní části označení klávesy, dosáhnete pomocí přeřaďovače - klávesy (SHIFT). Na klávesnici jsou dvě klávesy (SHIFT). Můžete použít kteroukoliv z nich.

Zkuste napsat s použitím kláves (CAPS), (SHIFT) a (1):

!!!ATARI 130XE!!! !!!atari 130xe!!!

Zkoušejte různá slova, písmena a interpunkční znaménka.

Grafické symboly

Mnoho kláves je označeno dvěma nebo třemi symboly. Každá klávesa s písmenem má na horní straně písmeno a na přední straně grafický

symbol. Některé z dalších kláves mají tři symboly nebo slova, všechny na horní straně. Jedna z funkcí klávesy je aktivována stlačením samotné klávesy, druhá stlačením (SHIFT) a klávesy a třetí stlačením (CONTROL) a klávesy. Grafické symboly jsou získávány stlačením (CONTROL) a příslušné klávesy.

Grafické symboly (symboly uvedené na přední straně abecedních kláves) vytisknete pomocí klávesy (CONTROL) na levé straně klávesnice. Nejdříve stlačte klávesu (CONTROL) a držte ji stlačenou. Přitom stlačte klávesu s grafickým symbolem. Pak obé klávesy uvolněte.

Držte stlačenou klávesu (CONTROL) a zkuste napsat ATARI. Na obrazovce uvidíte:

Na obrazovce se objevilo pouze pět znaků namísto písmen. Použijete-li klávesy (CONTROL) současně s klávesami s číslicemi, neobjeví se žádné grafické symboly.

Grafické symboly jsou velmi užitečné při tvorbě grafické úpravy zobrazení, při vytváření rámečků a jednoduchých uměleckých prací. V grafickém módu můžete klávesnici "uzamknout" držením tlačítka (CONTROL) a současným stlačením (CAPS). Zpět z grafického módu se vrátíte stlačením tlačítka (CAPS).

Ovládání kurzoru

Klávesa (CONTROL) je nejčastěji používána k řízení pohybu kurzoru. Kurzor je malý bílý čtvereček, který označuje místo na obrazovce, ve kterém se nacházíte. Napravo od písmene P naleznete klávesu s šipkou nahoru. Šipka, stejně jako označení klávesy (CONTROL), je nakreslena bíle, což znamená, že funkce této klávesy je aktivována pouze při použití klávesy (CONTROL). Stlačte klávesu (CONTROL) a pak současně klávesu se šipkou nahoru a pozorujte pohyb kurzoru k hořejšku obrazovky. Dosáhne-li kurzor vrcholu vrátí se skokem na spodek obrazovky a začíná se znovu po obrazovce stěhovat vzhůru. Nyní vyzkoušejte ostatní klávesy se směrovými šipkami. Nezapomeňte použít klávesu (CONTROL).

Mazání obrazovky

Klávesa (CONTROL) je často používána ve spojení s klávesou (CLEAR) k vymazání obrazovky. Držte stlačenou klávesu (CONTROL) a stlačte klávesu (CLEAR). Tento postup by měl způsobit vymazání obrazovky a vrácení kurzoru do levého horního rohu obrazovky.

Zapište nyní obrazovku různými písmeny, čísly, slovy a grafickými symboly. Použijte tentokrát k vymazání obrazovky klávesu (SHIFT) spolu s klávesou (CLEAR). Obě funkce (SHIFT CLEAR) a (CONTROL CLEAR) jsou totožné - mažou obrazovku a vrací kurzor do levého horního rohu obrazovky.

Vkládání

Klávesa (CONTROL) spolu s klávesou (INSERT) vkládají v textovém nebo programovém řádku mezeru pro znak. Funkci si procvičte na příkladu: Napište

```
! ! ! ATARI 130XE ! ! !
```

Umístěte kurzor na první písmeno A ve slově ATARI. Držte stlačenou klávesu (CONTROL) a pak 11 krát stlačte klávesu (INSERT). Nápis nyní bude vypadat:

```
! ! !           ATARI 130XE ! ! !
```

Doprostřed řádku bylo vloženo 11 mezer. Tato funkce je velmi užitečná při vkládání písmen nebo slov. Použitím kláves pro pohyb kurzoru (šipky a CONTROL) vraťte kurzor na první prázdné místo za třetí vykřičník. Stlačte jednou mezerník a napište TOTO JE ME; Text pak bude vypadat:

```
! ! ! TOTO JE ME ATARI 130XE ! ! !
```

Místo z řady jednotlivých mezer vytvoříte místo pro nový prázdný řádek držením klávesy (SHIFT) a současným stlačením klávesy (INSERT). Na obrazovce se objeví celý nový prázdný řádek. Vložte několik dalších

prázdných řádků, ale ne zase tolik, abyste měli prázdnou obrazovku. V následující úloze použijete právě napsané věty.

Vypouštění

Vypouštění znaků pomocí kláves (CONTROL) a (DELETE BK SP) je stejně snadné jako jejich vkládání. Posuňte kurzor na první písmeno T ve slově TOTO. Při stlačeném (CONTROL) stlačte jedenáckrát (DELETE BK SP). Vaše věta bude vypadat nyní takto:

! ! ! ATARI 130XE ! ! !

Nyní víte, jak spolu pracují klávesy (CONTROL) a (DELETE BK SP). Posuňte nyní kurzor na první písmeno A a pozorujte, co se bude dít, použijete-li samotnou klávesu (DELETE BK SP). Stlačte nyní třikrát (DELETE BK SP). Na obrazovce nyní bude:

ATARI 130XE ! ! !

Je-li klávesa (DELETE BK SP) použita samostatně, posouvá kurzor doleva a maže vše, co přijde kurzoru do cesty. Místo vymazaných znaků však nechává mezery, zatímco při použití společně s klávesou (CONTROL) posouvá na místo vypuštěného znaku celý zbytek řádku napravo.

Třetí funkce klávesy (DELETE BK SP) vyžaduje použití klávesy (SHIFT). Stlačíte-li při stlačené klávese (SHIFT) klávesu (DELETE BK SP), bude vypuštěn celý řádek a kurzor se vrátí na levý okraj. Nezáleží na tom, na kterém místě řádku bude při stlačení (SHIFT DELETE BK SP) kurzor umístěn; celý řádek bude z obrazovky vymazán.

Tabelování

Posuňte na prázdné obrazovce kurzor na levý kraj a natiskněte hvězdičku. Stlačte klávesu (TAB). Pokaždé, když se kurzor zastaví, natiskněte kvězdičku. Po pátém stlačení (TAB) budete mít na obrazovce rozmístěno šest hvězdiček:

* * * * * *

Stlačujte nyní pouze klávesu (TAB) a všimněte si, že se pokaždé zastaví na předem označeném místě. První značka je 5 mezer od levého okraje (běžné odsazení na začátku odstavce), ostatní značky jsou odděleny osmi mezerami. Přemístěte kurzor nahorů na první hvězdičku a posuňte jej o tři místa doprava. Při stlačené klávese (SHIFT) pak stlačte (TAB). Tím jste aktivovali funkci (SET TAB) (nastav tabulátor). Posuňte kurzor zpět na levý kraj a stlačte (TAB). Kurzor skočí do nově nastavené polohy.

Pokračujte ve stlačování (TAB). Kurzor prochází všemi nastavenými tabelačními značkami včetně nově přidané. Po odskoku na následující řádek ignoruje kurzor novou tabelační značku. (Ale na všech dalších řádcích prochází kurzor všemi značkami - novou i předešle nastavenými.) Vraťte kurzor na první hvězdičku a stlačte (TAB). Nová tabelační značka je dosud na místě.

Vraťte kurzor na levý okraj. Stlačením (TAB) přesuňte kurzor na první tabelační značku (tři místa). Pomocí kláves (CONTROL) a (TAB) aktivujte funkci (CTR TAB) - mazání značky. Pomocí (TAB) se přesuňte na další značku a vymažte ji také. Vraťte kurzor zpět na levý okraj téhož řádku a stlačte samotnou klávesu (TAB). Kurzor by měl přeskočit dvě značky. Pokračujte ve stlačování (TAB) až kurzor přejde na následující řádek. (CLR TAB) na tomto řádku nevymazala druhou značku. (Avšak ze všech následujících řádků byly vymazány obě značky.)

I n v e r z n í z o b r a z e n í (Inverze video)

Napište slovo ATARI. Stlačte klávesu inverzního zobrazení a napište znovu ATARI. Pak opět stlačte klávesu Inverse Video a znovu napište ATARI. Na obrazovce bude nápis, který bude obsahovat tři slova ATARI, z nichž prostřední bude zobrazeno inverzně.

Klávesa Inverze Video vytváří inverzní zobrazení znaků na obrazovce - modrá písmena na bílém pozadí. Tato funkce je velmi užitečná při zvýrazňování písmen ve Vašich programech. Stačí jeden dotyk klávesy Inverze Video, aby se změnil způsob zobrazení písmene.

Některé další klávesy

Další důležitou klávesou je klávesa Escape (ESC). Stlačíte-li ji jednou, nic se nestane. Stlačíte-li ji dvakrát nebo vícekrát, objeví se na obrazovce grafický znak: . Stlačte (RETURN) a zkuste to znovu. V dalších částech budete klávesu (ESC) potřebovat.

Klávesa (BREAK) je v pravém horním rohu. Stlačíte-li tuto klávesu, skočí kurzor na další řádek a posune se na levý okraj. Jak používat klávesu (BREAK) se dovíte v odstavci o programových cyklech.

Když stlačíte klávesy (HELP), (START), (SELECT) a (OPTION), nic se nestane. Tyto klávesy jsou programovatelné a často se používají v tržním software.

Po stlačení klávesy (RESET) se na jednu až dvě sekundy obrazovka vyčistí a pak se objeví nápis READY. Klávesa (RESET) znovu spouští systém. Tuto klávesu byste měli používat velmi opatrně, protože v mnoha programech dojde ke ztrátě informací, které vkládáte, nebo které jste vložili.

Napsání jednoduchého BASIC programu:

NEW. LIST. PRINT. RUN.

Nyní, když jste zvládli klávesnici, je snadné napsat Váš první program. Vymažte obrazovku a vraťte kurzor do levého horního rohu.

NEW: Vymazání paměti počítače

Napište slovo NEW a stlačte (RETURN):

```
NEW
```

Příkaz NEW znamená pro počítač, aby vymazáním všech starých instrukcí, které by mohly být v paměti počítače, připravil počítač pro soubor nových instrukcí.

LIST: Ověření obsahu paměti

Abyste měli jistotu, že v paměti počítače nic nezůstalo, požádejte počítač o výpis programu. Napište příkaz LIST a stlačte (RETURN):

```
LIST
```

Jestliže jste před tím napsali příkaz NEW správně, nemělo by se na obrazovce objevit nic jiného než slovo READY. Nyní můžete začít s novým programem. Napište první řádek programu. Opište řádek přesně jak jej vidíte napsaný zde a po posledních uvozovkách stlačte (RETURN):

```
10 PRINT"SLYSEL JSEM O BASNIKU JMENEM SAM"
```

Veškeré řádky s programovými instrukcemi jsou v programech BASIC číslovány. Při psaní tohoto jednořádkového programu dejte pozor, aby 1 a 0 v čísle řádku 10 byly číslice a ne písmena. Použijeteli místo číslic písmena, obdržíte chybové hlášení.

Číslovaný programový řádek může být delší než je řádek na

obrazovce. Po naplnění jednoho řádku na obrazovce kurzor automaticky přeskakuje na začátek dalšího řádku. Klavesu (RETURN) stlačujte vždy až na konci programového řádku, nikoliv řádku na obrazovce.

Stlačením (RETURN) oznamujete počítači, že máte ukončeno psaní instrukce (programového řádku) a že počítač má tento řádek uložit do své paměti. Při stlačení (RETURN) se kurzor vrátí na levý okraj, takže můžete rovnou začít psát další řádek programu.

RUN: P r o v á d ě c í i n s t r u k c e

Chcete-li, aby počítač vykonal Vaše programové instrukce, musíte napsat příkaz RUN. Příkazem RUN říkáte počítači, aby provedl instrukce, které má uloženy v paměti. Napište RUN, stlačte (RETURN) a dívejte se, co se bude dit:

```
RUN

SLYSEL JSEM O BASNIKU JMENEM SAM
```

První a jedinou instrukci, řadek 10, bylo vytisknout text v uvozovkách. Vymažte obrazovku, napište opět RUN a stlačte (RETURN). Počítač opět provede instrukci a vytiskne SLYSEL JSEM O BASNIKU JMENEM SAM.

Přesto, že instrukce již není na obrazovce, počítač si pamatuje, co má dělat. Váš program je uložen v programovatelné části počítačové paměti RAM. Napíšete-li LIST, počítač Vám na obrazovku vytiskne všechny instrukce, uložené v jeho paměti RAM. Napište příkaz LIST a stlačte (RETURN). Vaše obrazovka by měla vypadat následovně:

```
LIST

10 PRINT"SLYSEL JSEM O BASNIKU JMENEM SAM"
```

Vypadá-li Vaše obrazovka jinak, zapomněli jste pravděpodobně po ukončení každého zápisu stlačit (RETURN) nebo jste napsali příkaz LIST do řádku, kde již bylo něco napsáno. Napište následující řádek a dejte příkaz RUN:

```
20 PRINT"JEDNOHO DNE JSEM SE S NIM SETKAL, A K MENU PREKVAPENI,"
RUN
```

Na obrazovce se objeví slova, uzavřená do uvozovek, z obou řádků programu. Napište příkaz LIST a přesvědčte se, jaké instrukce má počítač uloženy ve své paměti RAM. Objeví se řádky 10 a 20.

Číslování řádků: tvoření pořadí

Každý řádek, obsahující instrukce, musí v BASIC programu (rozumě) v programu, napsaném v jazyce BASIC) mít na začátku číslo. Tato čísla se nazývají "čísla řádků". Počítač vykonává instrukce počínaje nejnižším číslem řádku, potom pokračuje podle jejich pořadí, až jsou všechny instrukce vykonány. Obvykle se řádky číslují 10,20,30,40 atd., takže v případě pozdější potřeby zůstává dost prostoru pro vložení dalších řádků. Zkuste nyní vložit řádek. Přidejte řádek 15 a přikažte počítači, aby vykonal program.

Vaše obrazovka by měla vypadat následovně:

```
15 PRINT"JEHOZ BASNE HOVORILY O VLASTI."

RUN

SLYSEL JSEM  O BASNIKU JMENEM  SAM
JEHOZ BASNE  HOVORILY O VLASTI.
JEDNOHO DNE JSEM SE S NIM SETKAL, A K MEMU PREKVAPENI,
```

Počítač vložil automaticky řádek 15 mezi řádky 10 a 20. Napište další řádek :

```
30 PRINT"JEHO MYSLENKY BYLY KREMENNYM PISKEM."

RUN

LIST
```

Příkazy RUN i LIST způsobí vytištění všech čtyř řádků na obrazovku.

Chybové hlášení: Počítač říká "Nerozumím"

Příkaz PRINT říká počítači, aby vytisknul to, co je uzavřeno v uvozovkách. Počítač se nestará o to, jaká slova nebo symboly jsou v uvozovkách; slova nemusí být správně napsaná ani nemusí dávat smysl. Vyzkoušejte následující instrukci:

```
40 PRINT"NZDRA SEMA!"
RUN
```

Dokonce i když je v uvozovkách nesmyslná věta s pravopisnými chybami, počítač udělá, co má přikázáno. Zkuste však zkomolit intrukci PRINT a uvidíte, co se stane:

```
50 PRIMT"NAZDAR SAME!"
```

Počítač Vám pošle zprávu o chybě - chybové hlášení. Počítač kontroluje pouze tu část instrukce, která je vně uvozovek, protože tato část je určena pro počítač. Instrukce, uvedené v uvozovkách, jsou určeny Vám, takže počítač je přesně kopíruje. Posuňte se na volný řádek, ale nemažte obrazovku. Spusťte program.

Objeví se zpráva o chybě - Error 17 at line 50 - chyba 17 řádku 50, v řádku, ve kterém jste záměrně zkomolili PRINT. Toto chybové hlášení 17 se nazývá "Syntaktická chyba". Naznačuje, že instrukce byla pro počítač nedešifrovatelná. (Úplný seznam chybových hlášení najdete v Dodatku D.)

Chybu naznačenou chybovým hlášením lze opravit několika způsoby. Nejsnadnějším řešením je posunout kurzor na řádek, který obsahuje chybu. Umístěte kurzor na chybné písmeno M v PRINT a zaměňte jej (přepište) za N. Stlačte (RETURN). (V tomto případě můžete stlačit (RETURN) bez ohledu na polohu kurzoru v řádku, dokonce i když je kurzor uprostřed slova PRINT). Tentokrát se již žádné chybové hlášení neobjeví. Vymažte obrazovku a spusťte program. Na obrazovce by se nyní nemělo objevit žádné chybové hlášení.

Jiným způsobem opravy chyby je vymazání chybného řádku. Vyzkoušejte si tento způsob napsáním řádku se záměrnou chybou. Vynechejte tentokrát v příkazu PRINT uvozovky a pak spusťte a vypište program:

```
60 PRINT MEL JSEM JEDNOU PROGRAM NAZVANY BOZON

RUN

LIST
```

Stlačíte-li (RETURN) a pokoušíte-li se rozběhnout nebo vypsat program, objeví se chybové hlášení. Chybný řádek se jednoduše vymaže napsáním jeho čísla řádku a stlačením (RETURN).

Nyní se program vypisuje a běží bez chyb, i když řádek 60 neobsahuje žádné instrukce. Řádek MEL JSEM JEDNOU PROGRAM NAZVANY BOZON byl vymazán. Napsání čísla řádku a stlačení (RETURN) způsobí vymazání celého řádku z paměti počítače. Napište řádek správně:

```
60 PRINT"MEL JSEM JEDNOU PROGRAM NAZVANY BOZON"

RUN
```

PRINT: V y t v á ř e n í p r á z d n ý c h ř á d k ů

Vložení prázdného řádku po textu dělá text čitelnějším. Vytvořte následující instrukcí prázdný řádek mezi textem, obsaženým v instrukcích na řádcích 30 a 40 :

```
35 PRINT
RUN
LIST
```

Když za příkazem PRINT nic nenásleduje, vytvoří počítač prázdný řádek. Vložte ještě jeden prázdný řádek mezi řádky 50 a 60. Použijte číslo řádku 55 a po něm napište jen PRINT.

?: Z k r a t k a p r o P R I N T

Při psaní instrukce PRINT můžete ušetřit čas a námahu, když místo PRINT napíšete otazník. Vyzkoušejte následující řádek programu:

```
70 ? "KTERY BEZEL OD RANA DO VECERA."

RUN

LIST
```

Program běží stejně s otazníkem jako s PRINT. Otazník je jen pohodlná zkratka. Všechny následující instrukce PRINT, použité v tomto výkladu, jsou psány pomocí slova PRINT, avšak můžete místo něj bez obav použít otazník.

Délka logického řádku

Někdy obsahuje text příliš mnoho znaků, než aby se vešli do jednoho nebo dvou řádků. Před napsáním následující ukázky nastavte na televizoru nebo monitoru hlasitost tak, aby byly slyšitelné zvuky z počítače:

```
80 PRINT "ODMITAL REAGOVAT NA ESCAPE, BREAK, CONTROL NEBO LIST
A STALE BEZEL, I KDYZ JSEM POCITAC VYPNUL A VYTAHL SNURU ZE
ZASUVKY."
```

Když se kurzor blíží ke konci třetího řádku, ozve se bzučák. Bzučák Vám naznačuje, že se blížíte k maximální délce programového řádku. Programový řádek nemůže být delší než tři řádky na obrazovce. Tento limit se nazývá "logický řádek". (Nyní můžete již stáhnout hlasitost.)

Zobrazení na obrazovce

Při delším textu, který přesahuje na obrazovce jeden řádek, bývají často slova na konci řádku nevhodně rozdělené. Umístění textu na řádku je také jiné, když píšete instrukce, nebo když text vytiskne počítač během zpracování programu. Chcete-li se vyhnout těmto problémům, rozmyslete si, co chcete mít na tom kterém řádku a napište pro každý řádek samostatnou instrukci. Přepište nyní větu z řádku 80 do následujícího formátu:

```
80 PRINT "POCITAC REAGOVAT NA ESCAPE, "
90 PRINT "BREAK, CONTROL NEBO LIST "
100 PRINT "A STALE BEZEL, I KDYZ JSEM POCITAC VYPNUL"
110 PRINT " A VYTAHL SNURU ZE ZASUVKY."
RUN
LIST
```

T i s k g r a f i c k ý c h s y m b o l ů

K vytváření jednoduchých předloh můžete použít v instrukci PRINT grafických symbolů. Oddělte od sebe texty, které jste vytvořili. Pro vytvoření grafických symbolů použijte /CONTROL J/ a /CONTROL H/:

```
58 PRINT "▙"
115 PRINT "▟"
```

PRINT "↑": M a z á n í o b r a z o v k y

Chcete-li na začátku programu, nebo na některém jiném místě vymazat obrazovku, napište číslo řádku, PRINT a uvozovky. Pak stlačte /ESC/ a pak /SHIFT CLEAR/ nebo /CONTROL CLEAR/ klávesy. Na obrazovce se objeví zahnutá šipka. Napište uvozovky a stlačte /RETURN/. Vypište a spusťte program:

```
5 PRINT "↑"
RUN
LIST
```

/CONTROL 1/: Z a s t a v e n í ř á d k o v á n í

Nyní program vypadá lépe, ale je příliš dlouhý a všechny jeho řádky se nevejdou najednou na obrazovku. Při vypisování programu můžete řádky, běžící zdola nahoru po obrazovce, zastavit stlačením /CONTROL 1/. Napište příkaz LIST. Ke stlačení kláves /CONTROL/ a /1/ použijte dvou prstů levé ruky a jedním prstem pravé ruky stlačte /RETURN/. /CONTROL 1/ zastavuje i znovu rozbíhá pohyb řádků po obrazovce.

Vytvoření konverzační smyčky:

GOTO, DIM, INPUT

Smyčky nebo také cykly přikazují počítači vrátit se v programu zpět a opakovat automaticky instrukce, ve smyčce zahrnuté. Jedním z takovýchto příkazů je instrukce GOTO (jdi na řádek).
Příkazy DIM a INPUT Vám umožňují konverzovat s počítačem systémem otázek a odpovědí.

GOTO

Nejjednodušší programovou smyčkou je smyčka vytvořená pomocí instrukce GOTO. GOTO je vždy následována číslem řádku, na kterém má program pokračovat. K vytvoření smyčky potřebujete dva příkazy. Pomocí následujícího programu vytvořte nekonečný cyklus:

```
NEW
110 PRINT"GRATULUJEME!"
120 GOTO 110
RUN
```

Běh nekonečné smyčky přerušíte buď vypnutím počítače, nebo klávesou /RESET/ nebo použitím /BREAK/. Zastavíte-li běh programu pomocí klávesy /BREAK/, objeví se na obrazovce jedna ze zpráv:

```
STOPPED AT LINE 110 (zastaveno na řádku 110)
   nebo
STOPPED AT LINE 120 (zastaveno na řádku 120)
```

Počítač Vám oznamuje, kterou instrukci právě vykonává, když obdržel příkaz k zastavení.

Č á r k a: Tabelátor

Váš program si můžete zpestřit přidáním čárky za instrukci PRINT. Vypište program, přesuňte kurzor do mezery za posledními uvozovkami

a napište čárku. Pak stlačte /RETURN/. Spusťte program a pozorujte účinek:

```
LIST
110 PRINT"GRATULUJEME!",
120 GOTO 110
RUN
```

Čárka plní funkci tabelátoru. Při každém průchodu programu instrukcí na řádku 110 je vytisknuto vše v uvozovkách počínaje na následující tabelační značce. Tabelační značky jsou od sebe vzdáleny 10 znaků. Nezapomeňte na přerušení cyklu klávesou /BREAK/.

S t ř e d n í k: počítačové "lepidlo"

Další účinek má středník. Vypište program, nahraďte čárku v řádku 110 středníkem, stlačte /RETURN/ a spusťte program:

```
LIST
110 PRINT"GRATULUJEME!";
120 GOTO 110
RUN
```

Středník na konci instrukce PRINT způsobí, že další slovo GRATULUJEME bude napsané těsně za předchozí, bez jakékoliv mezery, stejně tak každé další bude "nalepeno" na předchozím. Chcete-li mít slova oddělena mezerou, musíte ji vložit do uvozovek. Vypište program a upravte následovně řádek 110:

```
110 PRINT"GRATULUJEME! ";
RUN
```

D v o j t e č k a: oddělovač příkazů

Dvojtečka odděluje instrukce. Umožňuje použití více instrukcí v jednom programovém řádku. Nahraďte v řádku 110 středník dvojtečkou a přidejte následující instrukci PRINT:

```
110 PRINT"GRATULUJEME!":PRINT"PRAVE JSTE VYHRAL V LOTERII."
RUN
```

Se stoupajícími programovacími schopnostmi bude pro Vás nabývat na důležitosti otázka stupné zaplnění počítačové paměti Vašimi programy. Jednou cestou, vedoucí k šetření paměti, je spojování instrukcí do jednoho řádku pomocí dvojteček. Na následující příkaz odpoví počítač číslem, udávajícím dosud nezaplněnou kapacitu paměti. Údaj je v bytech:

```
PRINT FRE (0)
```

Přeprogramujte řádek 110, aby každá instrukce PRINT byla na jednom řádku:

```
110 PRINT "GRATULUJEME!"
115 PRINT "PRAVE JSTE VYHRAL V LOTERII."
PRINT FRE (0)
```

Srovnejte dvě různá obdržená čísla, udávající volnou kapacitu paměti. Druhý údaj po přeprogramování je o dvě až tři jednotky menší než první. Protože u začínajících programátorů je důležitější jednoduchost v zápisu programu než šetření kapacitou paměti, budou obvykle programové řádky v této části příručky obsahovat pouze jednu instrukci v řádku. Výjimku bude tvořit instrukce PRINT, která vkládá do programu prázdný řádek. Přepište ještě jednou řádek 110 a sledujte účinek:

```
110 PRINT: PRINT"GRATULUJEME!"
RUN
```

DIM a INPUT : Dimenzování a zavádění řetězcových proměnných

Počítač je zkonstruován tak, že umí konverzovat pomocí otázek a odpovědí. K položení otázky můžete použít příkaz PRINT a k získání odpovědi příkaz INPUT. Vložíte-li však do počítače odpověď, musí počítač vědět, kam ji má uložit. Počítač ukládá odpověď do paměťové buňky v paměti RAM, která se nazývá "proměnná". Je-li odpovědí číslo, je to

"číselná proměnná". Je-li odpověď sestavena z písmen, číslic, nebo dalších znaků a jejich kombinací, nazývá se stringová nebo "řetězcová proměnná". Váš ATARI potřebuje vědět, jak velký prostor bude potřebovat na uložení odpovědi, aby si mohl potřebné místo v paměti rezervovat. Velikost potřebného prostoru sdělíme počítači pomocí dimenzování příslušné proměnné instrukcí DIM.

Příkaz DIM vždy doprovází příkaz INPUT, jedná-li se o řetězcové proměnné, protože DIM určuje očekávanou velikost odpovědi. U řetězcových proměnných odpovídá číslo v instrukci DIM maximálnímu očekávanému počtu znaků včetně mezer.

Změňte program, který obsahoval smyčku, na program, který dává otázku a očekává odpověď. Program není třeba přepisovat; Napište jen nové řádky - řádky 10, 120, 130 a 140. (Napsáním nového řádku 120 se automaticky vymaže starý řádek 120.)

```
10 DIM ODPOVED$ (100)
110 PRINT: PRINT"GRATULUJEME!"
115 PRINT"PRAVE JSTE VYHRAL V LOTERII."
120 PRINT: PRINT"JAK SE CITITE?"
130 INPUT ODPOVED$
140 PRINT"JA JSEM SI TO MYSLEL."
RUN
```

Řádek 10 říká počítači, aby rezervoval v paměti dostatek prostoru pro odpověď, jejíž maximální délka bude 100 znaků. Proměnná je v tomto programu nazvána ODPOVED. Proměnná očekává písmena, případně i čísla, musí to proto být řetězcová proměnná. Řetězcové proměnné jsou označeny po posledním písmenu v názvu proměnné znakem dolar.

Řádek 30 Vám umožňuje vložit do počítače odpověď. Po spuštění programu zobrazí počítač na obrazovce otázku, a Vy pak napíšete svou odpověď. Odpověď je uložena do řetězcové proměnné, nazvané ODPOVED$. Vynecháte-li řádek 10 s instrukcí DIM, objeví se zpráva o chybě a instrukce INPUT nebude pracovat.

?: Zdvořilost instrukce INPUT

Spusťte opět program. Na obrazovce se objeví dva otazníky. Druhý otazník bude na dalším řádku u levého okraje. Vypište svůj program a všimněte

si, že jste napsali pouze jeden otazník. Instrukce INPUT vždy napíše na obrazovku otazník. Změňte řádek 120:

```
120 PRINT: PRINT "JAK SE CITITE";
```

Spusťte program a na požádání počítače napište odpověď. Nyní je na obrazovce již jen jeden otazník, dodaný instrukcí INPUT, a Vaše odpověď je napsána do téhož řádku bezprostředně za otázku. Vytvořte více takovýchto dialogů pomocí dimenzování více řetězcových proměnných a vložení více instrukcí INPUT. Instrukce DIM se dává na začátek programu, vždy musí předcházet sesterskou instrukci INPUT:

```
20 DIM DATUM$(25)
140 PRINT: PRINT"KDY BYSTE SI NEJRADEJI VYZVEDNUL SVOU VYHRU";
150 INPUT DATUM$
RUN
```

Řetězcové proměnné v instrukci PRINT

Počítačový program obsahuje nyní dvě otázky, ale nereaguje na Vaši poslední odpověď. Odezva počítače může být realizována formou vložení řetězcové proměnné do instrukce PRINT. Napište následující instrukce:

```
160 PRINT:PRINT"VELICE LITUJI,ALE NASE KANCELAR JE VZDY ";DATUM$;.
" ZAVRENA."
165 PRINT"TO JE VELICE MRZUTE!"
```

Středníky připojují řetězcovou proměnnou (její obsah) mezi dvě věty v uvozovkách. Na konci první věty a na začátku druhé věty v uvozovkách musí být mezera. Spusťte program a opravte případné chyby. Zaveďte ještě další řetězcovou proměnnou:

```
30 DIM JMENO$(1)
170 PRINT"NAKONCHODEM, JAK SE JMENUJETE";
180 INPUT JMENO$
190 PRINT"TAK TY JSI ";JMENO$;"! SAZIM SE, ZE BYS RAD VEDEL, KOLIK JSI VYHRAL."
```

195 PRINT"NEJSTRIVE ALE MUSIS ODPOVEDET NA JEDNU OTAZKU."

Spusťte program. Ačkoliv jste napsali celé jméno, počítač vytiskl jenom první iniciálu. Stalo se tak proto, že prostor, rezervovaný pro jméno v paměti RAM, byl příliš malý. Většina lidských jmen je delší než jedno písmeno. Změňte řádek 30 na rozumnější dimenzi a spusťte znovu program:

```
30 DIM JMENO$(25)
RUN
```

Vkládání číselných proměnných

Dosud jste pracovali s alfanumerickými řetězcovými proměnnými - proměnnými složenými z písmen, číslic nebo obojího. Počítač by například přijal jméno R2-D2 nebo 007 jako řetězcovou proměnnou. Avšak toto číselné jméno by mohlo být použito pouze jako jméno, nikoliv jako číslo v matematických výpočtech. Zkuste nyní vložení číselné proměnné, která může být použita v matematických výpočtech. Jednoduché číselné proměnné nevyžadují příkaz DIM a nemají označení $ (dolar). Vložte následující programové řádky:

```
200 PRINT: PRINT"KOLIK JE TI LET";
210 INPUT VEK
220 CENA = VEK*1000
230 PRINT: PRINT"PRAVE JSI VYHRAL V LOTERII ";CENA;" $."
240 PRINT"MUZES SI JE VYZVEDNOUT V UREDNICH HODINACH."
```

V tomto programu je vložená hodnota stáří osoby uložena v číselné proměnné VEK. Řádek 220 vytváří další proměnnou CENA. Na řádku 220 vypočítává kalkulátor, vestavěný v počítači, cenu výhry, která je tisícinásobkem věku výherce. (Hvězdička znamená pro počítač znaménko pro násobení.)

Cena výhry je uložena v proměnné CENA. V řádku 230 je uvnitř instrukce PRINT vložena číselná proměnná a to tímtéž způsobem, jako byla před tím vložena proměnná řetězcová.

Konverzační smyčky

Abyste mohli opakovat Vaši konverzaci s počítačem, přidejte opět instrukci pro vytvoření smyčky. Instrukce GOTO, umístěná na konci programu, donutí počítač opakovat program od začátku. Pro lepší čitelnost programu použijte na začátek opakované části programu instrukci REM. Instrukce REM (remark - poznámka) funguje pro programátora jako návěstí nebo poznámka. Počítač instrukci nevykonává, poznámky si nevšímá, pouze ji při výpisu programu vytiskne.

```
100 REM * * * KONVERZACNI SMYCKA * * *
250 GOTO 100
```

Počítač se musí vracet na řádek 100, nikoliv na řádek 10, protože nemůže jít dvakrát přes instrukci DIM pro řetězcové proměnné. Uděláte-li smyčku přes tutéž instrukci DIM, podá Vám počítač zprávu o chybě.

Použití náhodných čísel a matematických funkcí RND,+,-,*,/.

Původně byly počítače vyvinuty k rychlému a snadnému zpracování čísel. Abyste mohli využít schopností počítače, který dovede vypočítat matematickou úlohu v několika milisekundách, musíte se naučit k počítači hovořit.

Čísla

Napište následující instrukci a stlačte /RETURN/:

```
PRINT 10
```

Počítač vytiskne číslo 10. Při zadávání čísel dejte pozor na záměnu číslic za písmena (nejčastější chybou bývá záměna písmene O za nulu). Vyzkoušejte další čísla:

```
PRINT 1000000000
PRINT -100000000
```

Záporná čísla mají před sebou znaménko minus, které je na klávese se šipkou vzhůru. U čísel nepoužívejte čárku. Na následujících příkazech si ověřte, co udělají čárky mezi čísly:

```
PRINT 9,876,543,210
PRINT 9, 876, 543, 210
```

V obou případech počítač interpretuje čárky jako oddělovací znaménko v řadě čísel. Rozmístí čísla po obrazovce do tabelačních pozic, určených čárkami. V tomto příkladě neznamená devítka pro počítač 9 miliard, ale pouze číslo 9, následované řadou jiných čísel.

Vědecký zápis

Počítač nerozumí mezi čísly, ale rozumí exponentům. Často sám automaticky převádí čísla větší velikosti do exponenciálního tvaru. Vyzkoušejte následující čísla:

```
PRINT 999999999
PRINT 555555555
PRINT 111111111
PRINT -111111111
PRINT -987654321112
```

Tato čísla jsou dostatečně velka nebo malá, aby počítač dal přednost jejich exponenciálnímu vyjádření. Porozumění vědeckému zápisu není pro porozumění počítači základní otázkou, ale jen jeho částí.

Vědecký zápis vyjadřuje velká čísla jako čísla mezi nulou a 10 nasobené mocninou 10. Mocninu 10 určuje exponent. V následujícím příkladě znamená zápis E+13, že exponent je 13:

$2.5E+13 = 2.5 \times 10^{13} = 25\ 000\ 000\ 000\ 000$

Exponenty můžete používat i při konverzaci s počítačem. Symbolem pro exponent je vsuvkové znaménko ve tvaru střišky, které se nachází na klávese se šipkou vpravo. Toto znaménko je aktivováno pomocí /SHIFT/. Zkuste následující výpočty:

```
PRINT 2^1
PRINT 2^2
PRINT 2^3
PRINT 2^4
PRINT 2^64
```

První zápis znamená dvě na prvou; druhý dvě na druhou atd. Poslední zápis je dvě na čtyřiašedesátou, číslo dost velké na to, aby jej počítač vyjádřil v exponenciálním tvaru.

Nejste-li fyzik, počítající dobu oběhu elektronů po jejich obězných dráhách, nebo astronom, počítající velikost vesmíru, budete vědecký zápis zřídka kdy potřebovat. Budete-li však přece tento zápis potřebovat, počítač je připraven provést Vaše výpočty i s těmito obrovskými čísly.

Počítač jako kalkulačka

Počítač může vykonávat tytéž funkce jako kalkulačka. Pro sčítání použijte znaménko plus na klávese se šipkou vlevo. Napište instrukci:

PRINT 1+1

Počítač Vám odpoví ihned po stlačení /RETURN/, stejně jako kalkulačka. Vymyslete si sami nějaké příklady na sčítání. Tvořte velká i malá čísla. Zkuste přičítání dlouhé řady čísel. Vyzkoušejte mnoho možností.

Pro odčítání použijte znaménka mínus na klávese se šipkou nahoru. Zkuste nyní tři verze téhož příkladu:

```
PRINT 4 - 1
PRINT 4-1
PRINT4-1
```

Jakmile stlačíte /RETURN/ objeví se ve všech případech stejný výsledek. Mezery jsou v matematických příkladech pro počítač nepodstatné. Vyzkoušejte svoje vlastní příklady. Sestavte dlouhé příklady, obsahující sčítání a odčítání.

Znaménko násobení - hvězdička (*)- je umisténa na klávese se šipkou vpravo. Znaménko dělení je zlomková čára na klávese s otazníkem. Napište následující instrukce:

```
PRINT 2 * 2
PRINT (2 * 2)
PRINT 6/3
PRINT (6/3)
```

Počítač nejenže rozumí použití závorek v matematických úlohách, ale ve složitějších úlohách se bez nich neobejde. Všimněte si rozdílu ve výsledku v následujících příkladech:

```
PRINT 3 * (2 + 2)
PRINT 3 * 2 + 2
```

Výsledek prvního příkladu je 12, druhého 8. V prvním příkladu počítač nejdříve sečte 2 a 2, výsledek násobí 3. Ve druhém příkladu nejdříve počítač násobí 3 a 2, pak přičítá 2. Kdykoliv narazí počítač v matematické úloze na závorky, provede nejdříve výpočet uvnitř závorek a potom teprve zbývající část výpočtu. Vyzkoušením následujících příkladů objevíte další zajímavá fakta týkající se práce počítače. Zkuste před stlačením /RETURN/ předpovědět výsledek:

```
PRINT (2 + 2) * 3
PRINT 2 + 2 * 3
```

V prvním příkladu provede počítač nejprve úkon v závorce. Ve druhém příkladu provede počítač nejdříve násobení a až potom přičítání:

1. úkony uvnitř závorek
2. exponenciální funkce
3. násobení a dělení, přitom postupuje zleva doprava
4. přičítání a odčítání, opět zleva doprava.

Tato pravidla jsou přehledně uspořádána v následující tabulce:

Pořadí vykonávání matematických funkcí:

1.	() Výpočty v závorkách	
2.	Exponenciální funkce	
3.	* Násobení	v pořadí
	/ Dělení	zleva doprava
4.	+ Sčítání	v pořadí
	- Odčítání	zleva doprava

N á h o d n á č í s l a

Počítač může vykonávat další funkce, které nejspíše Vaše kalkulačka nedovede. Počítač například dovede vygenerovat náhodné číslo. Napište následující program:

```
NEW
10 PRINT RND(0)
20 GOTO 10
RUN
```

RND je příkaz pro generování náhodných čísel. Ve výše uvedeném programu jsou generována náhodná čísla v nekonečné smyčce. Smyčku můžete přerušit

klávesou .BREAK/. V programu můžete provést některé změny. Stačí si nechat program vypsat a pomocí kurzoru a vložit další znaky namísto přepisování celých řádků. Vyzkoušejte následující varianty programu:

```
10 PRINT RND(1)
RUN
10 PRINT RND(123)
RUN
10 PRINT RND(50)
RUN
10 PRINT RND(50000)
RUN
```

Všechny čtyři varianty programu generuji náhodná čísla mezi 0 a 1. U náhodného čísla je vždy před první platnou cifrou desetinná tečka. Některá náhodná čísla, která maji platnou cifru i na levé strané od desetinné tečky, jsou stále čísla mezi 0 a 1, ale jsou tak malá, že je počítač napsal v exponenciálnim tvaru.

Číslo v závorce se nazývá "fiktivní proměnná". Nezáleži na tom, jaké číslo se pro fiktivní proměnnou použije, důležité jsou závorky a že je něco v nich (libovolné číslo nebo písmeno). Pro snadnost zápisu je obvykle dosazována za fiktivní proměnnou nula. Změňte následovně řádek 10:

```
10 PRINT(RND(0) * 10)
RUN
10 PRINT(RND(0) * 100)
RUN
10 PRINT(RND(0) * 1000)
RUN
```

Každý program generuje rozdílný rozsah náhodných čísel. PRINT(RND(0) * 10) generuje čísla do 10, protože instrukce přikazuje počítači násobit náhodné číslo 10. Násobením deseti se posunula o jedno místo desetinná tečka. V další variantě je náhodné číslo násobeno stem a desetinná tečka je posunuta o dvě místa. Podobně v poslední ukázce násobení náhodného čísla tisícem posouvá desetinnou tečku o tři místa. Chcete-li, můžete násobit mnohem většími čísly a obdržet tak vyšší náhodná čísla.

Protože dlouhá čísla s mnoha číslicemi za desetinnou tečkou jsou

při dalším zpracování těžkopádná, je počítač vybaven instrukcí, pomocí níž se tisknou pouze celá čísla, bez desetinných míst. Instrukce INT říká počítači, aby si nevšímal číslic za desetinnou tečkou. Přeprogramujte tři varianty řádku 10 a porovnejte výsledky:

```
10 PRINT INT(RND(0) * 10)
RUN
10 PRINT INT(RND(0) * 100)
RUN
10 PRINT INT(RND(0) * 1000)
RUN
```

Programy generují čísla v témže rozsahu jako předtím, ale čísla jsou bez číslic za desetinnou tečkou čitelnější.

Zkuste následující příklady generování náhodných čísel v různých rozsazích:

```
10 PRINT INT(RND(0) * 3)
RUN
10 PRINT INT(RND(0) * 12)
RUN
10 PRINT INT(RND(0) * 25)
RUN
```

Program generuje náhodná čísla, která jsou vždy o jednotku menší než číslo, kterým jste náhodné číslo v programu násobili. První řádek 10 generuje náhodná čísla 0, 1 a 2. Aby program generoval náhodná čísla 0, 1, 2 a 3, musel by být napsán takto:

```
10 PRINT INT(RND(0) * 4)
RUN
```

Aby program generoval pouze čísla 1, 2 a 3, musel by vypadat takto:

```
10 PRINT INT(RND(0) * 3) + 1
RUN
```

Chcete-li, aby program generoval tři náhodná čísla počínaje číslem 20, napište program takto:

```
10 PRINT INT(RND(0) * 3) + 20
RUN
```

Náhodná čísla v hrách

Programy s náhodnými čísly jsou velice pružné. Můžete je dokonce použít při hraní her s Vaším počítačem. Napište následující program. Vzpomeňte si, jak napsat zahnutou šipku, která je na řádku 5. Stlačte /ESC/, pak stlačeném /SHIFT/ nebo /CONTROL/ stlačte /CLEAR/.

```
NEW
1 REM * * * CISLO.HRA * * *
5 PRINT " ↖ "
10 TAJ=INT(RND(0) * 3) + 1
20 PRINT :PRINT "MYSLIM SI CISLO, 1,2, nebo 3. ZKUS JEJ UHADNOUT."
30 INPUT CISLO
40 IF CISLO=TAJ THEN PRINT"VYHRAL JSI!"
50 IF CISLO<>TAJ THEN PRINT"NEUHODL JSI."
60 GOTO 10
```

Řádek 10 přiřazuje číselné proměnné TAJ hodnotu náhodného čísla. Řádek 30 zavádí do počítače hodnotu hádaného čísla a přiřazuje ji k proměnné CISLO. (Uvědomte si, že číselné proměnné nemusí být dimenzovány ani označeny $ jako řetězcové proměnné.) Řádek 40 porovnává vygenerované a hádané číslo. Rovnají-li se tato čísla, počítač vytiskne "VYHRAL JSI!". Řádek 50 rovněž porovnává vygenerované a hádané číslo. Nerovnají-li se čísla (symbol <> znamená nerovnost), počítač vytiskne "NEUHODL JSI." Řádek 60 vytváří smyčku, takže hru můžete opakovat. (Následující kapitola vysvětluje instrukce IF - THEN podrobněji.)

Matematické programy

Matematické funkce počítače můžete použít jak pro pracovní účely, tak pro hry. Kdybyste byl šéfkuchařem, který připravuje stravu pro bankety, mohl byste využít počítač k rozšíření Vašeho receptáře. Představte si, například, že se pokoušíte spočítat, kolik liber mořských ústřic máte

nakoupit, chcete-li servírovat k večeři Coquilles St. Jacques a budete mít 62 hostů. Váš předpis udává normu 1 1/2 libry ústřic pro 5 osob. Následující program Vám vypočítá, kolik liber ústřic máte koupit.

```
NEW
1 REM * * * COQUILLE * * *
10 PRINT " "
20 HOSTE=62
30 KOUPITLIBER=1.5/5 * HOSTE
40 PRINT :PRINT"KUP ";KOUPITLIBER;" LIBER USTRIC."
50 END
```

Program dává odpověď (18.6 liber ústřic), ale pomocí kalkulačky byste dosáhli téhož výsledku s vynaložením menší námahy. Program se stane užitečnějším, když místo počtu hostů vložíte do programu instrukci INPUT. Napište doplňující řádky:

```
15 PRINT: PRINT"KOLIK HOSTU OCEKAVAS";
20 INPUT HOSTE
```

Spusťte několikrát program a pokaždé dosaďte jiný počet hostů.Požadované množství ústřic bude pokaždé jiné. Pro 200 hostů 60 liber, pro 436 hostů 130.8 liber. S funkcí INPUT se program stává praktičtějši.

IF - THEN, FOR - NEXT

Příkazy IF - THEN a FOR - NEXT (když - pak a pro - další) Vám umožňují psát programy, které napodobují lidský přístup při tvoření rozhodnutí nebo řešení problému. Zvláště jsou užitečné pro hry a logické hádanky. Tyto příkazy umožňují programátorovi, tedy Vám, aby nechal rozhodovat počítač.

P ř í k a z y I F - T H E N

Pro procvičení instrukce IF - THEN si napište následující program:

```
NEW
1 REM *** BRNPROBE.QZ ***
5 PRINT " ¶ "
10 DIM RAIN$(3)
20 PRINT :PRINT "YES OR NO, IF IT WERE RAINING OUTSIDE, WOULD YOU GO OUT
WITH AN UMBRELLA";
30 INPUT RAIN$
40 IF RAIN$="YES" THEN PRINT "YOU HAVE A FORMIDABLE IQ."
50 IF RAIN$="NO" THEN PRINT "YOU ARE A BORN RISK TAKER."
```

Tento programový kvíz využívá Vaší odpovědi k rozhodování o dalším postupu. Je-li v řádku 40 odpověď, uložená v proměnné RAIN$,"YES", tiskne počítač zprávu, že máte nebezpečně velké IQ. Není-li odpověď "YES", přechází počítač k dalšímu řádku 50 a opět testuje proměnnou RAIN$. Je-li odpověď "NO", sděluje Vám počítač, že jste od narození riskér. Není-li odpověď ani YES ani NO, program prostě končí. Program nemá žádné instrukce pro případ neurčité odpovědi. Vyzkoušejte si to.

Jedním ze způsobu, jak docílit správné odpovědi, je vytvoření nekonečné smyčky. Vložte dodatečný řádek:

```
60 GOTO 20
```

Vyhodnocení pomocí IF - THEN

Jinou cestou docílení správné odpovědi je vybavení programu pokyny. Následující program používá k "vylákání" správné odpovědi číselné proměnné:

```
NEW
1 REM *** NUMBER.QZ ***
5 PRINT " " "
10 SECRETNUM=INT(RND(0)*10)+1
20 PRINT :PRINT "GUESS A SECRET NUMBER BETWEEN 1 AND 10."
30 PRINT
40 PRINT "YOUR GUESS";
50 INPUT GUESS
60 PRINT
70 IF GUESS=SECRETNUM THEN PRINT "YOU GOT IT!":END
80 IF GUESS<SECRETNUM THEN PRINT "TOO LOW. TRY AGAIN.":GOTO 40
90 IF GUESS>SECRETNUM THEN PRINT "TOO HIGH. TRY AGAIN.":GOTO 40
```

Řádky 80 a 90 vyhodnocují odpověď jako větší nebo menší než tajné číslo. Instrukce PRINT dává hádajícímu pokyn, podle kterého se může přibližovat ke správné odpovědi. V případě, že pokračujete v hádání nesprávnými odpověďmi, ubírá se program po nekonečných smyčkách, které vytvářejí instrukce GOTO v řádcích 80 a 90.

Ukončení programu

Program Number Quiz se zastaví pouze při nalezení tajného čísla. Po vložení správné odpovědi dá řádek 70 počítači instrukci k ukončení. END ukončuje program a na obrazovce se objeví Ready.

Ošetření chyb

Vložíte-li neúmyslně do proměnné GUESS místo čísla písmeno, zobrazí počítač chybové hlášení a program náhle končí. Udělejte úmyslnou chybu

tím, že napíšete písmeno, nebo pouze stlačíte /RETURN/. Chcete-li se vyhnout takovémuto nechtěnému ukončení programu, můžete použít instrukci TRAP, která zachycuje zprávu o chybě. Přidejte k programu následující řádky a spusťte jej:

```
45 TRAP 100
100 PRINT: PRINT"PLEASE ENTER A NUMBER ONLY."
110 GOTO 30
```

Řádek 45 s instrukcí TRAP říká počítači, aby v případě chyby nezastavoval program, ale přešel na řádek 100. V řádku 100 dává počítač pokyn, jak napravit chybu. Řádek 110 vrací počítač do místa, ze kterého lze napravit chybu. Instrukce TRAP se dává vždy před instrukcí INPUT a vždy obsahuje číslo řádku, ve kterém je uvedeno řešení vzniklého problému.

Kviz pomocí IF - THEN

Program může snadněji poskytovat pokyny, je-li správná odpověď číslo, včetně data. Následující program používá k vyhodnocení odhadu letopočtu instrukci IF - THEN a TRAP.

```
NEW
1 REM *** LOVELACE.QZ ***
5 PRINT " ¶ "
10 PRINT :PRINT "ADA LOVELACE, DAUGHTER OF HTE POET LORD BYRON, WAS
MATHEMATICALLY BRILLIANT."
20 PRINT
30 PRINT "IN WHAT YEAR DID SHE WRITE HER AMAZIGLY ACCURATE DESCRIPTION
OF THE FUTURE USES OF THE COMPUTER";
40 TRAP 200
50 INPUT GUESS
60 IF GUESS=1842 THEN GOTO 100
70 IF GUESS<1842 THEN GOTO 110
80 IF GUESS>1842 THEN GOTO 120
100 PRINT :PRINT "CONGRATULATIONS! YOU GUESSED THE YEAR CORRECTLY.":END
110 PRINT :PRINT "THAT WAS TOO ERALY. TRY AGAIN.":GOTO 20
120 PRINT :PRINT "THAT WAS TOO LATE. TRY AGAIN.";GOTO 20
```

```
200 PRINT :PRINT"PLEASE ENTER A NUMBER ONLY."
210 GOTO 20
```

V tomto kvizu je umístění instrukcí PRINT, sdružených s instrukcemi IF - THEN a obsahujících pokyny, odlišné od předchozích dvou kvizů. Tento rozdíl ukazuje, že ke stejnému výsledku vedou v programování často různé cesty.

Počítačové "mouchy"

Instrukce TRAP činí předchozí kviz odolnějším proti chybám, avšak stále ochrana není dokonalá. Protože počítač vyhodnocuje datum jako číslo, přijme 1842.78 jako správnou hodnotu a 1842,78 jako nesprávnou. Většina programů má své "mouchy", neboli slabá místa nebo problémy. Umíte-li tato slabá místa určit a ošetřit, naučili jste se skutečně programovat. Každý začátečník se při programování setkává s mnoha problémy a dělá mnoho chyb. Studujete tuto příručku, abyste se stali lepšími programátory, čerpejte i z dalších pramenů, podívejte se do popisu jazyka ATARI BASIC. Příležitostně si nechte nakouknout přes rameno zkušenějším programátorem. Naučíte se, jak odhalovat slabá místa v programu, takže se v budoucnu budete moci vyvarovat podobných chyb.

C y k l u s F O R - N E X T: počítající smyčka

V nekonečných smyčkách GOTO jste již jako doma. Dalším druhem smyček je cyklus FOR-NEXT. Cyklus FOR-NEXT si sám počítá, kolikrát proběhnul. Počet cyklů je narozdíl od GOTO konečný. Napište NEW a vložte do počítače následující program:

```
NEW
10 FOR X=1 TO 4
20 PRINT "BRAMBORA"
30 NEXT X
RUN
```

Řádek 10 znamená "Pro X od jedné do čtyř". Řádek 30 znamená "Další X".

slovo BRAMBORA bude na obrazovku vytištěno čtyřikrát. Změňte nyní řádek 10:

```
10 FOR X=1 TO 7
```

Když spustíte program tentokrát, budou na obrazovce brambory sedmkrát. Počítač proběhl sedmkrát smyčku, sestávající z řádků 10, 20 a 30. FOR říká počítači, kterou hodnotu má dosadit do proměnné X jako první a kterou hodnotou má končit. Určuje tedy, kolikrát smyčka proběhne. NEXT říká počítači, aby se vrátil na začátek a dosadil do proměnné X v pořadí další hodnotu. Jako proměnnou můžete použít libovolnou kombinaci abecedněčíslicových znaků. Zkuste tuto proměnnou:

```
10 FOR CISLO=1 TO 7
30 NEXT CISLO
```

Spustíte-li program, nebudete pozorovat proti předchozímu žádnou změnu. Změňte proměnnou ještě jednou:

```
10 FOR JKL=1 TO 7
30 NEXT JKL
```

JKL je název, který nedává proměnné v cyklu FOR-NEXT žádný smysl. Přesto s ním bude program pracovat naprosto stejně. Vyzkoušejte si to a pak přidejte další řádek:

```
15 PRINT JKL,
RUN
```

Řádek 15 říká počítači, aby vytiskl hodnotu proměnné JKL. (Pro lepší čitelnost tisku je instrukce ukončena čárkou.) Pokaždé, když počítač probíhá smyčkou FOR-NEXT, nabývá proměnná následující hodnotu v řadě, určené řádkem 10. První hodnotou je 1; druhou hodnotou 2; atd. Po dosažení hodnoty posledního čísla v instrukci FOR (v tomto případě 7) proběhne počítač smyčkou naposled a více se nevrací. Změňte v řádku 10 toto číslo:

```
10 FOR JKL= 1 to 50
RUN
10 FOR JKL = 1 TO 200
```

```
RUN
10 FOR JKL = 1 TO 500
RUN
```

Počátecní bod

Vypište program.. První číslo v instrukci FOR je počátečním bodem v počítání průchodů smyčkou. Posledni číslo je ukončovacím bodem. Jako počátečního (i ukončovacího) bodu může být použito dokonce i záporného čísla. Vyzkoušejte tyto obmény řádku 10:

```
10 FOR JKL = 1 TO 5
RUN
10 FOR JKL = 0 TO 5
RUN
10 FOR JKL = 3 TO 5
RUN
10 FOR JKL = -10 TO 5
```

STEP (Krok): počítání s přírůstkem

Vypište program, vymažte řádek 20 s instrukcí PRINT, zrušte čárku v instrukci na řádku 15 a spusťte program. Počítač počítá a tiskne čísla velmi rychle. Použijete-li příkazu STEP, bude počítač měnit hodnoty proměnné ne po jedné, ale se stanoveným přírůstkem. vyzkoušejte následujici program:

```
10 FOR JKL = 0 TO 500 STEP 5
RUN
10 FOR JKL = 0 TO 500 STEP 2
RUN
10 FOR JKL = 0 TO 500 STEP 100
RUN
10 FOR JKL = 0 TO 500 STEP 7
RUN
```

Počítač bude poslušně počítat tak, jak mu určíte.

Počítání dozadu

Použijete-li v instrukci FOR správného pořadí čísel a příkaz STEP se zápornou hodnotou kroku, bude počítač počítat pozpátku od většího čísla k menšímu, Příklad:

```
10 FOR JKL = 500 TO 0 STEP -1
RUN
10 FOR JKL = 10 TO 0 STEP -1
RUN
10 FOR JKL = -1 TO -19 STEP -1
RUN
```

Počítač umí počítat dozadu i v přírůstcích:

```
10 FOR JKL = 500 TO 0 STEP -20
RUN
10 FOR JKL = 500 TO 0 STEP -3
RUN
10 FOR JKL = 0 TO -500 STEP -50
RUN
```

I zde můžete počítání začít a ukončit na libovolném čísle:

```
10 FOR JKL = 500 TO 300 STEP -10
RUN
10 FOR JKL = 25 TO 0 STEP -1
RUN
```

Nyní víte, jak máte Váš počítač instruovat, aby počítal dopředu a dozadu, aby počítal po jedné nebo se stanoveným přírůstkem, a aby začal a skončil počítání na stanoveném čísle.

"Sendvičová" smyčka FOR-NEXT

Vypište Váš program. FOR je na horním řádku, NEXT je na spodním řádku.

Vše, co chcete mít v této smyčce vykonáno, je vloženo, podobně jako v sendviči, mezi FOR a NEXT. Napište tyto řádky:

```
10 FOR JKL = 1 TO 5
20 PRINT "HRUSKY"
```

Počítač provede libovolnou instrukci nebo libovolné množství instrukcí, které jsou mezi instrukcemi FOR a NEXT, tolikrát, kolikrát mu určíte. Nechte počítač napsat tato slova:

```
16 PRINT "SYR"
17 PRINT "MAJONEZA"
18 PRINT "HORCICE"
19 PRINT "RAJCATA"
21 PRINT "SLANINA"
22 PRINT "SALAT"
23 PRINT:PRINT
RUN
```

Počítač počítá a tiskne tak rychle, že není možné text na obrazovce v průběhu tisku přečíst. Nicméně tiskne pětkrát přesně všechno dle příkazů PRINT. Součástí "sendviče" smyčky FOR-NEXT mohou být i jiné instrukce, jako matematické operace nebo instrukce INPUT.

Zpožďovací smyčky

Vymažte veškeré instrukce PRINT tak, aby v sendviči nebylo naprosto nic vyjma instrukcí FOR a NEXT:

```
15
16
17
18
19
20
21
22
23
```

```
LIST
```

Spusťte program a dívejte se, co se stane:

```
RUN
```

Nic se nestalo. Změňte číslo v řádku 10 a opět se pozorně dívejte:

```
10 FOR JKL = 1 TO 500
RUN
```

Slovo Ready se objevilo na obrazovce až po chvíli. Změňte opět řádek 10:

```
10 FOR JKL = 1 TO 5000
RUN
```

Tentokrát se slovo Ready objevilo značně později. Počítač počítá, ale nic netiskne, jako když si počítáte potichu pro sebe. Čas, který uplyne do zobrazení slova Ready, je čas, který počítač spotřeboval na počítání do 5000.

Cykly FOR-NEXT jsou výborným prostředkem pro dočasné "zastavení" počítače. Cykly FOR-NEXT jsou vskutku tak často pro tyto účely používány, že jsou někdy nazývány "zpožďovacími smyčkami". Název proměnné bývá obvykle DELAY (zpoždění). Přepište cyklus FOR-NEXT a použijte přitom proměnnou DELAY a jiná čísla v instrukci FOR:

```
NEW
10 FOR DELAY  1 TO 300
20 NEXT DELAY
LIST
RUN
```

Někdy bývá zpožďovací smyčka umístěna v jednom řádku:

```
NEW
10 FOR DELAY = 1 TO 300:NEXT DELAY
LIST
RUN
```

Ukázkové programy

Následující programy používají různými způsoby cyklů FOR-NEXT. První program používá cyklu FOR-NEXT jako jednoduché zpožďovací smyčky, pomocí které zůstanou určená slova chvíli na obrazovce, aby mohla být vůbec před vymazáním řádkem 30 přečtena"

```
NEW
1 REM *** DLAYLOOP ***
5 PRINT " ↰ "
10 PRINT "HI"
20 FOR DELAY=1 TO 800:NEXT DELAY
30 PRINT " ↰ "
40 PRINT "BYE"
50 FOR DELAY=1 TO 800:NEXT DELAY
```

Následující program používá v cyklu FOR-NEXT číselné proměnné. Rovněž používá příkazu TRAP, který bez podání chybového hlášení vrací počítač na předchozí řádek:

```
NEW
1 REM *** HOWNIGH? ***
10 DIM A$(1),HH$(1)
20 PRINT " ↰ "
30 PRINT :PRINT "HOW HIGH DO YOU WANT TO COUNT";
40 TRAP 30
50 INPUT HH
55 HH$=STR$(HH):IF HH$="0" THEN GOTO 30
60 FOR COUNT=1 TO HH
70 PRINT COUNT
80 NEXT COUNT
90 PRINT :PRINT "PLEASE ANSWER (Y/N). WOULD YOU LIKE TO COUNT AGAIN";
100 TRAP 90
110 INPUT A$
120 IF A$="Y" THEN GOTO 30
130 IF A$="N" THEN PRINT :PRINT "BYE":END
140 GOTO 90
```

Poslední program je parafrází na rock n'roll a používá vloženého cyklu FOR-NEXT. Vloženým cyklem je menší zpožďovací smyčka uvnitř většího cyklu FOR-NEXT. Pro rozšíření podmínek v instrukci IF-THEN je použito operátoru OR:

```
NEW
1 REM *** CLOCKROCK ***
5 PRINT " "
10 FOR X=1 TO 9
20 PRINT X;
30 PRINT "O'CLOCK"
40 FOR DELAY=1 TO 500:NEXT DELAY
50 IF  X=3 OR X=6  OR X=9  THEN  PRINT "ROCK!": FOR  PAUSE=1 TO 500:NEXT
PAUSE
60 NEXT X
70 PRINT :PRINT "WE'RE GOING TO ROCK"
80 PRINT "AROUND THE CLOCK"
90 PRINT "TONIGHT!"
```

Tvoření zvuků a grafika SOUND, SETCOLOR, COLOR

Na některých počítačích je tvoření zvuků a grafiky velmi komplikované, ne však na ATARI. Vše, co potřebujete, jsou zvukové příkazy jazyka ATARI BASIC spojené s některými jednoduchými programovacími technikami.

ATARI může hrát až čtyři zvuky najednou. Každému zvuku, nebo hlasu, je přiřazen jede zvukový registr. Tyto registry jsou očíslovány 0,1,2 a 3. Instrukci SOUND 1 volíte první hlas, SOUND 1 druhý hlas, SOUND 2 třetí a SOUND 3 čtvrtý hlas.

Příkaz SOUND má v ATARI BASICu 4 parametry:
číslo hlasu (0 - 3)
výška tónu (0 - 255)
zkreslení (0 -14)
hlasitost (0 - 15)

Výška tónu, nebo jeho frekvence, je určena číslem mezi 0 a 255, což dává celkem 256 frekvencí, z kterých si můžete vybrat. Výška tónu je druhým parametrem (číslem) v příkazu SOUND. Instrukce SOUND 1,50 určuje druhý hlas s výškou tónu 50. Přesvědčte se, že nemáte na televizoru nebo monitoru staženou hlasitost a napište instrukci:

```
SOUND 1,50,0,8
```

Stlačte /RETURN/. Pořádný výbuch, že. Zvuk vypnete buď stažením hlasitosti na televizoru, nebo napsáním jednoho z následujících příkazů a stlačením /RETURN/:

```
END
SOUND 1,0,0,0
```

Zkreslení, nebo čistota zvuku, je určena sudým číslem v rozmezí 0 a 14. V příkazu SOUND určuje čistotu zvuku třetí parametr. Zkuste takovýto zvuk:

```
SOUND 1,50,10,8
```

Číslo 10 určuje čistý tón, bez zkreslení. Malé zkreslení dosáhnete záměnou čísla 10 za číslo 6:

```
SOUND 1,50,6,8
```

Počítač vydává velice nepříjemný zvuk. Napište END, než si začnou sousedé ztěžovat.

Poslední číslo v příkazu SOUND řídí hlasitost. Číslo se musí pohybovat v mezích 0-15. Pro většinu použití je vhodné číslo 8. Při větším čísle než 12 riskujete poškození reproduktoru a vašich uší.

Pomocí následujících příkazů vytvořte čtyřhlasý akord:

```
SOUND 0,50,10,8
SOUND 1,100,10,8
SOUND 2,150,10,8
SOUND 3,200,10,8
```

Zastavte znění akordu napsáním END.

Určení zvuku pomocí proměnných parametrů

Svůj program oživíte použitím proměnných v instrukci SOUND. Pomocí proměnných parametrů můžete programovat změnu hlasu, výšky tónu, zkreslení, a hlasitosti vydržovaných tónů. Zavedte a spusťte následující program"

```
NEW
10 REM * NASTAVENI HODNOT PROMENNYCH ZVUKOVYCH PARAMETRU
20 HLAS = 0:LADENI = 100:TON = 8:HLASITOST = 8
30 SOUND HLAS,LADENI,TON,HLASITOST
40 GOTO 20
RUN
```

Zvuk zastavíte stlačením /BREAK/ a napsáním END. K prodloužení zvuku potřebujete v programu opakovat příkaz SOUND. Obvykle se k tomuto účelu používá dvou metod - smyčky FOR-NEXT, nebo jako v předchozím příkladě, smyčky s GOTO. Následující program používá ve smyčce FOR-NEXT proměnnou výšku tónu a předvádí celý rozsah ladění:

```
NEW
10 REM * SOUND EFFECTS WITH FOR-NEXT LOOP
20 VOICE=0:PITCH=0:TONE=10:VOL=8
30 FOR PITCH=0 TO 255
40 SOUND VOICE,PITCH,TONE,VOL
50 NEXT PITCH
RUN
```

Změna hlasitosti zvuků v programu vytváří zvukovou rozmanitost. Změňte VOL=8 na VOL=0 a stlačte /RETURN/. Pak přidejte následující řádek:

```
35 VOL=INT(RND(0) * 16 )
```

Tento řádek volí náhodně hodnotu parametru hlasitosti v rozmezí 0 a 15. Spusťte tento program a zjistíte, jak ovlivňuje náhodná změna hlasitosti zvuk.

Hudební tóny

Příkaz SOUND umí také vytvářet hudební tóny. Následující tabulka obsahuje hudební tóny a jejich příslušné hodnoty parametru výšek tónu. Frekvence tónu se určí ze vztahu:

Frekvence = 31 960/(hodnota parametru+1)

TABULKA HUDEBNÍCH TÓNŮ

Tón	parametr	Tón	parametr
C	243	Fis 1	85
Cis	230	G 1	81
D	217	Gis 1	76
Dis	204	A 1	72
E	193	Ais 1	68
F	182	H 1	64
Fis	173	C 2	60
G	162	Cis	57
Gis	153	D	53
A	144	Dis	50
Ais	136	E	47
H	128	F	45
C 1	121	Fis	42
Cis 1	114	G	40
D 1	108	Gis	37
Dis 1	102	A	35
E 1	96	Ais	33
F 1	91	H	31
		C 3	29

Napište a spusťte následující program:

```
NEW
10 REM ** SIMPLE SONG
15 DIM PITCH$(1)
20 VOICE=0:PITCH=0:TONE=10:VOL=8
30 REM ** C=121:D=108:E=96:F=91
40 TRAP 300
50 PRINT "¶"
60 PRINT "NOTES FOR SIMPLE SONG"
65 FOR NOTE=1 TO 8
70 READ PITCH
80 SOUND VOICE,PITCH,TONE,VOL
90 GOSUB 200
```

```
100 PRINT :PRINT PITCH$
110 FOR PAUSE=1 TO 500:NEXT PAUSE
120 SOUND 0,0,0,0
130 NEXT NOTE
140 GOTO 30
150 REM ** DATA FOR NOTES
160 DATA 121,121,108,96,96,91,108,121
200 REM ** PRINT NOTES
210 IF PITCH=121 THEN PITCH$="C"
220 IF PITCH=108 THEN PITCH$="D"
230 IF PITCH=96 THEN PITCH$="E"
240 IF PITCH=91 THEN PITCH$="F"
250 RETURN
300 PRINT :PRINT "END OF SIMPLE SONG":END
RUN
```

Příkazy GOSUB-RETURN a READ-DATA umožňují počítači vložením řady hodnot do proměnné PITCH (ladění) vytvářet různé tóny. Instrukce GOSUB říká počítači: jdi do podprogramu, který začíná na řádku 200 a pokračuje až k řádku 250. Příkaz RETURN říká počítači, aby se vrátil na následující příkaz za příkazem GOSUB. Příkaz READ říká počítači, aby vyzvedl z instrukce DATA jednu položku a její hodnotu přiřadil proměnné PITCH. Položky jsou vybírány postupně v pořadí, v jakém jsou zapsány, až je použit celý řádek s instrukcí DATA.

Cyklus FOR-NEXT je v programu použit také k určení délky trvání tónů. Zkuste modifikovat program tvořením celých, půlových a dalších druhů not pomocí dalších cyklů FOR-NEXT.

Barevná grafika

ATARI 130XE má 16 grafických módů a pracuje s 256 barvami. Tato část popisuje 6 různých módů a některé ze základních grafických příkazů.

Následující tabulka uvádí 16 základních barev a jim odpovídající číselné hodnoty. (Barevné podání závisí ovšem také na nastavení příslušných ovladacích prvků televizoru.)

0	Šeda	8	Modrá
1	Zlatá	9	Světle modrá
2	Oranžová	10	Tyrkysová
3	Červenooranžová	11	Modrozelená
4	Růžová	12	Zelená
5	Purpurová	13	Žlutozelená
6	Červenooranžová	14	Oranžovozelená
7	Modrá	15	Světle oranžová

Zbývajících 112 barev je vytvořeno z těchto základních barev pomocí různých hodnot jasu. Jas musí být vyjádřen sudým číslem v rozmezí 0 - 14. Čím vyšší jasové číslo, tím světlejší a jasnější barva.

Důležitou součástí grafiky ATARI jsou barvové registry. Můžete si je představit jako kalíšky s barvami. V každém registru může být obsažena libovolná ze 128 barev. Protože barvových registrů je 5, může být současně zobrazeno pět barev. Barvovým registrům odpovídají čísla 0,1,2,3 a 4. Důležitým příkazem je SETCOLOR. Formát instrukce je SETCOLOR x,x,x. Prvním parametrem je číslo barvového registru; druhým je číslo barvy a třetím je jas.

Grafický mód 0

Barvové registry mají v různých grafických módech různou funkci. V následující tabulce je ukázána funkce barvových registrů v grafickém módu 0 (textový mód):

Standardní barvy	Registr	Funkce
	0	Není použit
Světle modrá	1	Jas textu
Tmavě modrá	2	Pozadí
	3	Není použit
Černá	4	Orámování

Nepřikážete-li počítači použít jiných barev, používá automaticky standardních barev. Pomocí instrukce SETCOLOR můžete barvy změnit. Napište:

```
SETCOLOR 2,3,4
```

Po stlačení /RETURN/ bude obrazovka oranžová. Změna barvy nastala následkem instrukce SETCOLOR, kde číslo 2 představuje pozadí obrazovky, číslo 3 oranžovou barvu a číslo 4 stupeň jasu. Změňte číslo 4 na číslo 6. Oranžová se mění na světleji oranžovou. Změňte 6 na 7. Nic se nestane, protože jas je určen pouze sudými čísly v rozsahu 0 - 14. Napíšete-li liché číslo, počítač použije barvu předešlého sudého čísla. Změňte 7 na 8. Barva bude ještě světlejší. Následující program ukazuje všech 128 kombinací barev a jasů:

```
NEW
10 REM ** 128 ATARI COLORS
20 REM ** 16 COLORS
30 FOR COLOR=0 TO 15
40 REM ** 8 LUMININACES
50 FOR LUMININACE=0 TO 14 STEP 2
60 SETCOLOR 2,COLOR,LUMINANCE
65 PRINT "COLOR=";COLOR;"LUMINANCE=";LUMINANCE
70 REM ** PAUSE TO SEE COLOR
80 FOR PAUSE=1 TO 600:NEXT PAUSE
90 NEXT LUMINANCE
100 NEXT COLOR
RUN
```

Dosáhne-li jas čísla 10, text zmizí, protože standardní jas textu je rovněž 10. (Není-li počítači přikázáno jinak, používá standardního jasu.) Vždy když je jas pozadí na obrazovce stejný s jasem textu, text zdánlivě zmizí. Používáme-li grafický mód 0, dejte pozor na jas pozadí a jas textu. K normálním (standardním) barvám obrazovky se vrátíte napsáním GR.0 (zkratka pro grafický mód 0).
Změňte v řádku 60 SETCOLOR 2 na SETCOLOR 4 a spusťte znovu program. Protože registr 4 je určen pro orámování, bude se teď měnit místo barvy pozadí barva orámování. K normálním barvám obrazovky se vrátíte napsáním GR.0.

Grafické módy 1 a 2

V grafických módech 1 a 2 je text větší s možností volby barvy. Grafický mód 2 je stejný jako grafický mód 1, až na to, že má dvakrát tak vysoká písmena. V módu 1 se na obrazovku vejde 24 řádků, v módu 2 jen 12. Vstupte do grafického módu 1:

```
NEW
10 GRAPHICS 1
20 PRINT #6;"GRAFICKY MOD 1"
```

Spusťte program. Nahoře na obrazovce bude oranžový text GRAFICKY MOD 1. Dole na obrazovce je modrý pruh se slovem Ready. Modrý pruh je textové okénko a zobrazuje text v grafickém módu 0. Napsáním GR.0 se vraťte do textového módu.

Velký text v grafických módech 1 a 2 se vytiskne pomocí instrukce PRINT #6; následované uvozovkami a požadovaným textem zakončeným opět uvozovkami. Tato instrukce je druhem instrukce PRINT, kterou jste se naučili dříve.

Vypište si nyní program, změňte "MOD" na "mod" a spusťte program. Slovo "MOD" bude nyní zelené. Pomocí klávesy pro inverzní zobrazení změňte v řádku 20 "mod" na "MOD" (zobrazené inverzně) a spusťte program. Slovo "MOD" bude nyní modré. Vypište si znovu řádek 20 a změňte inverzně napsané "MOD" na inverzně napsané "mod". Po spuštění programu bude slovo "MOD" červené.

Napište a spusťte následující program:

```
NEW
10 REM ** COLORFUL TEXT
20 GRAPHICS 1
30 PRINT #6;"ORANGE"
40 PRINT #6;"green"
50 PRINT #6;"[redacted]"
60 PRINT #6;"[redacted]"
70 PRINT "COLORFUL TEXT"
RUN
```

Jak vidíte, v grafickém módu 1 můžete zobrazit najednou pět barev - text ve čtyřech různých barvách a jednu barvu pozadí. Tyto barvy můžete použitím instrukce SETCOLOR změnit. V grafickém módu 1 volíte barvový registr druhem písma (malé, velké, inverzní). V následující tabulce jsou uvedeny standardní barvy v jednotlivých registrech pro grafický mód 1 a 2:

Registr	Standart. barva	Druh písma	Barva #	Jas
0	Oranžová	Velké	2	8
1	Světle zelená	Malé	12	10
2	Tmavě modrá	Velké inverzně	9	4
3	Červená	Malé inverzně	4	6
4	Černá	Pozadí	0	0

Napište SETCOLOR 4,15,5. Registr 4 (a i pozadí) bude nyní obsahovat oranžovou barvu. Avšak tmavěmodrý text je teď těžko čitelný. Změňte jej pomocí SETCOLOR. Podle tabulky pochází tmavěmodrá z registru 2. SETCOLOR 2,8,6 změní tmavěmodrý text na text o něco světlejší. Přidejte k programu Colorful Text následující řádky:

```
100 FOR COLOR=0 TO 15
110 SETCOLOR 2,COLOR,0
120 FOR DELAY=1 TO 400:NEXT DELAY
130 NEXT COLOR
```

Spusťte program. Spolu se změnou barvy modrého textu se mění i barva textového okénka v dolní části obrazovky, protože registr 2 ovládá vedle barvy zobrazovaného textu i barvu textového okénka.

Vypuštění textového okénka

V některých případech nebudete ve Vašem programu chtít, aby se na obrazovce objevilo textové okénko. Toho docílíte přičtením čísla 16 k číslu grafického módu. Změňte řádek 20 na GRAPHICS 17 a vypusťte řádek 70. Příkaz PRINT bude vždy tisknout v grafickém módu 0. Máte-li nastaven

grafickým mód 1 nebo 2 bez textového okénka a použijete instrukci PRINT a PRINT #6;, počítač je zmaten a vše tiskne v módu 0. Přidejte tento řádek:

```
70 PRINT "TEST OKENKA"
```

Spusťte program a pozorujte, co se stane. Chcete-li mít zobrazen text v módu 1 a použijete instrukcí PRINT a PRINT #6;, musíte použít textového okénka.

Vypusťte řádky 100, 110, 120 a 130 a spusťte program. V horní části obrazovky se objeví TEST OKENKA a pak Ready. Vypište program. Řádek 20 určuje mód 17 (což je mód 1 bez textového okénka). Kam zmizel? Nahraďte řádek 70 řádkem:

```
70 GOTO 70
```

Po spuštění programu se opět vrátí na obrazovku zobrazení v módu 1. Použijete-li mód 1 nebo 2 bez textového okénka, musíte použít smyčky GOTO, abyste udrželi zobrazení na obrazovce. Jinak se mihne na obrazovce příliš rychle na to, aby jste jej zahlédli. Stlačení klávesy /BREAK/ Vás vrátí do módu 0.

Chcete-li vidět, jak vypadá tento příklad v módu 2, vypište program Colorful Text a změňte řádek 20 na:

```
20 GRAPHICS 18
```

GRAPHICS 18 představuje mód 2 plus 16 (bez textového okénka). Spusťte program. Barevný text je nyní větší.

Chcete-li přejít zpět k původním barvám, stlačte /RESET/ a napište SETCOLOR 2,9,4. Stlačíte-li v ATARI BASICu /RESET/, program se nevymaže. Neplatí to však u jiných jazyků nebo programů.

G r a f i c k ý m ó d 3

V grafickém módu 3 je obrazovka rozdělena rastrem sestávajícím ze 40 sloupců a 24 řádků (20 při použití textového okénka). Napište a spusťte následující program:

```
NEW
10 GRAPHICS 3
20 COLOR 1
30 PLOT 0,0
RUN
```

V levém horním rohu obrazovky je oranžový čtvereček. Tento čtvereček je základní jednotkou grafického zobrazení neboli obrazovým elementem. Barvu obrazového elementu určuje instrukce COLOR. Číslo v instrukci COLOR určuje, který barvový registr bude pro barvu obrazového elementu použit. Příkaz COLOR neurčuje barvu registru; to dělá instrukce SETCOLOR. Instrukce COLOR prostě volí, který barvový registr bude použit pro znázornění obrazového elementu. Obrazový element pak bude mít takovou barvu, která je momentálně v barvovém registru. Aby to bylo jasnější, změňte řádek 20 na:

```
20 COLOR 2
```

Spusťte program. Původně oranžový obrazový element je nyní světle zelený. Považujte obrazový element za textový znak. V módech 1 a 2 jste používali k určení barvy textu velkých a malých písmen a inverzního zobrazení. V módech 3 a vyšších se k určení barvy obrazového elementu používá instrukce COLOR.

PLOT: umístění obrazového elementu do rastru

PLOT je jako PRINT #6;, avšak místo písmen a číslic tiskne obrazové elementy. Parametry v instrukci PLOT určují souřadnice místa v rastru, do kterého bude obrazový element umístěn. Instrukce COLOR určuje registr, jehož barvu bude mít obrazový element. Standardní barvy jsou oranžová, světle zelená, tmavě modrá a černá. Instrukcí SETCOLOR se mění barvy v barvových registrech.

Barvové registry jsou jako kalíšky s barvami. SETCOLOR určuje, kterou barvou budou naplněny a COLOR vybírá kalíšek, do kterého bude namočen štětec. PLOT určuje místo na obrazovce, do kterého bude štětec položen.

DRAWTO: Spojování bodů

Přidejte tento řádek:

```
40 DRAWTO 39,0
```

Spusťte program. Na obrazovce je zelená čára. Instrukce DRAWTO spojuje úsečkou posledně umístěný obrazový element s bodem na obrazovce,určený souřadnicemi instrukce DRAWTO.Řádek 40 říká počítači, aby zakreslil do sloupce 39 a řádku 0 obrazový element a spojil jej s místem poslední polohy kurzoru. Napište nyní:

```
DRAWTO 39,19
```

Tento příkaz zakresluje obrazový element do pravého dolního rohu obrazovky, těsně nad textové okénko a spojuje úsečkou body o souřadnicích 39,0 a 39,19. Napište:

```
DRAWTO 0,19
```

Dokončete obdélník:

```
DRAWTO 0,0
```

Napište nyní GR.0 a vypište program. Přidejte následující řádky:

```
50 DRAWTO  39,19
60 DRAWTO 0,19
70 DRAWTO 0,0
```

SETCOLOR a COLOR

Po spuštění programu počítač opět nakreslí zelený obdélník. Zvýrazněte obrázek:

```
35 COLOR 1
45 COLOR 2
```

```
55 COLOR 1
65 COLOR 3
```

Spusťte program a uvidíte obdélník v barvách.

Chcete-li změnit barvu v registru, použijte instrukce SETCOLOR. Mohli byste si udělat závěr, že COLOR 1 volí barvový registr 1 a COLOR 2 volí barvový registr 2. Bohužel tento závěr není zcela pravdivý. Mód 3 má čtyři registry a čtyři barvy - ale registry jsou očíslovány 0,1,2 a 3. Správné přiřazení ukazuje následující tabulka:

```
COLOR 0  Registr 4 černá
COLOR 1  Registr 0 oranžová
COLOR 2  Registr 1 světle zelená
COLOR 3  Registr 2 tmavé modrá
```

Napiš GR.0, vypište program a v řádku 20 změňte COLOR 2 na COLOR 1. COLOR 1 volí registr 0 a standardní barva v registru 0 je oranžová. Barvu v registru 0 změníte instrukcí SETCOLOR. Přidejte následující řádek:

```
15 SETCOLOR 0,4,6
```

Po spuštění programu se oranžové čáry změní na narůžovělé. Barvu jste změnili změnou barvy v jednom kalíšku (barvovém registru) použitím instrukce SETCOLOR, nikoliv použitím jiného kalíšku (registru) pomocí instrukce COLOR. Jas barvy registru 0 ovlivňuje také jas textu v textovém okénku. Přidejte nyní:

```
42 SETCOLOR 1,2,8
```

Barva pravé strany rámečku se změní ze světle zelené na zlatou. Přidejte další řádek:

```
62 SETCOLOR 2,11,4
```

Spusťte program. Na zelenou barvu se změní nejen levá strana rámečku, ale i textové okénko, neboť registr 2 určuje také barvu textového okénka.

Nyní byste měli již být schopni používat instrukce SETCOLOR a COLOR a pomocí Vašich programů dosahovat široké palety barev a odstínů.

Objasnění rozdílu mezi módy 3, 5 a 7 je velmi snadné. Změňte řádek 10 na:

```
10 GRAPHICS 5
```

Spusťte program. Obdélník je mnohem menší, protože i obrazové elementy jsou menší. S textovým okénkem má mód 3 40 sloupců a 20 řádků v rastru. Mód 5 má v rastru 80 sloupců a 40 řádků.

Změňte nyní řádek 10 na:

```
10 GRAPHICS 7
```

Po spuštění programu se objeví ještě menší obdélník. Rastr v módu 7 má 150 sloupců a 80 řádků.

Čím menší obrazové elementy, tím vyšší rozlišení. Z těchto tří módů má mód 3 nejnižší a mód 7 nejvyšší rozlišení. Zkuste nakreslit podobný obdélník, ohraničující zobrazovací pole, v módech 5 a 7.

Následující program ilustruje vše, co jste v této části vyzkoušeli. Napište jej a vyzkoušejte:

```
NEW
5 REM ** BILL'S BOX (PLOT AND DRAW)
10 PRINT "WHICH MODE (3,5,OR 7)";
20 LEFT=0:TOP=0
30 INPUT MODE
40 IF MODE=3 THEN RIGHT=39:BOTTOM=19
50 IF MODE=5 THEN RIGHT=79:BOTTOM=39
60 IF MODE=7 THEN RIGHT=159:BOTTOM=79
70 GRAPHICS MODE
80 PRINT "    GRAPHICS MODE";MODE
90 FOR COUNT=1 TO 1000
```

```
100 COLOR 2
110 TRAP 240
115 REM ** DRAW BOX
120 PLOT LEFT,TOP
130 COLOR 1
140 DRAWTO RIGHT,TOP
150 COLOR 2
160 DRAWTO RIGHT,BOTTOM
170 COLOR 1
180 DRAWTO LEFT,BOTTOM
190 COLOR 3
200 DRAWTO LEFT,TOP
205 REM ** DELAY LOOP
210 FOR DELAY=1 TO 500:NEXT DELAY
215 REM ** SIZE OF NEXT BOX
220 LEFT=LEFT+2:TOP=TOP+2:RIGHT=RIGHT-2:BOTTOM=BOTTOM-2
230 NEXT COUNT
240 PRINT "    THAT'S ALL FOLKS!"
250 END
```

Vyzkoušejte použití instrukce SETCOLOR ke změně barev v programu Bill s Box.

UKÁZKOVÉ PROGRAMY

S trochou pomoci správné programovací techniky a Vaší představivosti může Váš počítač ATARI dělat zázraky. Tyto ukázkové programy ukazují mnohostrannost Vašeho počítače a budou Vás motivovat k napsání Vašich vlastních programů.

Napište každý program přesně podle předlohy nezapomeňte po napsání každého programového řádku stlačit /RETURN/. Po napsání celého programu napište slovo RUN (běž), stlačte /RETURN/ a sledujte, jak se Váš počítač ATARI má k světu.

Poznámka: Jsou-li v některých řádcích programu důležité mezery v textu, určuje jejich požadovaný počet poznámka pod programem.

ATARI CHOO-CHOO

Specialitou ATARI jsou zvukové efekty. Když zavřete oči a spustíte program ATARI CHOO-CHOO, bude se Vám zdát, že jste v Marrakesh expresu.

```
10 POKE 764,255:POKE 580,1
20 GRAPHICS 17:POKE 712,148:POSITION 1,10:PRINT #6;"THE ATARI CHOO-CHOO"
30 FOR X=15 TO 0 STEP -1-P:SOUND 1,0,0,X
40 R=INT(RND(0)*300)+1
50 IF R=30 THEN SOUND 3,36,10,10:SOUND 2,48,10,10:GOSUB 90:SOUND
3,0,0,0:SOUND 2,0,0,0
60 NEXT X:P=P+0.03
70 IF P>=5 THEN P=5
80 GOTO 30
90 POKE 77,0:POSITION 8,12:PRINT #6;"toot":FOR A=1 TO 400:NEXT
A:POSITION 8,12:PRINT #6;" ":RETURN
```

Poznámka: Mezi uvozovkami v řádku 90 dejte 4 mezery.

BIG BANG

Před spuštěním následujícího programu zavřete dveře, abyste nerušili sousedy.

```
10  POKE  764,255:POKE  580,1
20  GRAPHICS  17
30  FOR     X=10     TO     100:SOUND:0,X,10,10:SOUND     1,X-2,10,8:SOUND
2,X+2,10,12:NEXT X
40  SOUND 1,0,0,0:SOUND 2,0,0,0
50  POSITION 4,11:PRINT #6;"BARGOOOMMM!"
60  FOR DECAY=15 TO 0  STEP -0.5:FOR B=1  TO 20:SOUND 0,100,B,DECAY:POKE
712,B:NEXT B:NEXT DECAY
70  GRAPHICS 1+32:POKE 712,148
80  POKE 752,1:PRINT :PRINT "PRESS START TO SET OFF ANOTHER EXPLOSION."
90  IF PEEK(53279)<>6 THEN GOTO 90
100 GOTO 20
```

TŘÍDĚNÍ SLOV

Tento třídící program seřazuje slova v abecedním pořádku. Nahraďte slova v příkazu DATA v řádcích 10 a 20 jinými dle Vašeho vlastního výběru a program Vám je seřadí. Nezapomeňte každé slovo oddělit od dalšího čárkou.

```
10 DATA ATARI,DISK DRIVE,MONITOR,COMPUTER,TOUCH TABLET,PRINTER,KEYBOARD
20 DATA  SOFTWARE,PROGRAM  RECORDER,WORD  PROCESSING,ACCOUNTING,DATA
BASE,FUN
30 DIM Z$(1000),A(50),A$(20),S(10)
40 S(1)=1:FOR L=1 TO 9:S(L+1)=S(L)*3+1:NEXT L
50 TRAP 80:GRAPHICS 0:? "HERE IS THE LIST:"
60 READ A$:B=LEN(Z$):C=LEN(A$):Z$(B+1,B+1)=CHR$(C):? A$
70 Z$(B+2,B+1+C)=A$:Q=Q+1:A(Q)=B+1:GOTO 60
80 ? :? "READY TO SORT...",:P=0
90 P=P+1:IF S(P+2)<Q THEN 90
100 FOR I=P TO 1 STEP -1:S=S(I):FOR J=S+1 TO Q:L=J-S:A=A(J):B=A(L)
110 IF Z$(A+1,A+ASC(Z$(A,A)))>Z$(B+1,B+ASC(Z$(B,B))) THEN 130
120 A(L+S)=B:L=L-S:IF L>0 THEN B=A(L):GOTO 110
130 A(L+S)=A:NEXT J:NEXT I:? :? "SORTED."
140 FOR L=1 TO Q:A=A(L):? Z$(A+1,A+ASC(Z$(A,A))):NEXT L
```

Tento program využívá techniky nazývané Player Missile Graphics k vytvoření růžového monstra, které se pohybuje po Vaší obrazovce před svislým modrým pruhem. Chcete-li, aby monstrum lítalo za modrým pruhem, změňte 150 na 150 POKE 623,4.

```
10 POKE 764,255:POKE 580,1
20 GRAPHICS 3+16
30 FOR X=16 TO 24:FOR Y=0 TO 23:COLOR 3:PLOT X,Y:NEXT Y:NEXT X
40 MEMTOP=PEEK(741)+256*PEEK(742)-1
50 PMBASE=INT((MEMTOP-1024)/1024)*1024
60 ADJTOP=PMBASE+384
70 POKE 742,INT(ADJTOP/256):POKE 741,ADJTOP-256*PEEK(742)
80 POKE 54279,PMBASE/256
90 POKE 53277,2
100 POKE 559,34+8
110 P0=PMBASE+512
120 FOR A=P0 TO P0+128:POKE A,0:NEXT A
130 FOR A=P0+60 TO P0+67:READ B:POKE A,B:NEXT A
140 POKE 53256,3
150 POKE 623,1
160 POKE 704,100
170 POKE 53248,PEEK(20):GOTO 170
180 DATA 60,126,129,153,255,36,66,129
```

TOPSY-TURVY

Až spustíte tento program, uvidíte na obrazovce nějaké cizí nápisy. Stlačením tlačítka /START/ je zlidštíte. Do původní podoby je dostanete stlačením /SELECT/.

```
10 POKE 764,255:POKE 580,1
20 GRAPHICS 18:POKE 712,128:POKE 755,5
30 POSITION 5,3:PRINT #6;"WELCOME TO"
40 POSITION 2,5:PRINT #6;"THE TOPSY-TURVY":POSITION 6,7:PRINT #6;"WORLD
OF":POSITION 6,9
50 PRINT #6;"COMPUTERS"
60 IF PEEK(53279)=5 THEN POKE 755,5:POKE 712,128
```

```
70 IF PEEK(53279)=6 THEN POKE 755,1:POKE 712,99
80 GOTO 60
```

NAPSANÁ PÍSNIČKA

Tento program přikazuje klávesám v horní řadě klávesnice tóny. Nestlačujte více kláves současně.
Přiřazení tónů klávesám:

Klávesa	Tón
Insert	B
Clear	$B^{\flat}$ (or A♯)
0	A
9	$A^{\flat}$ (or G♯)
8	G
7	F♯ (or $G^{\flat}$)
6	F
5	E
4	$E^{\flat}$ (or D♯)
3	D
2	$D^{\flat}$ (or C♯)
1	C

```
10 DIM CHORD(37),TUNE(12)
20 GRAPHICS 0:? :? "   TYPE-A-TUNE PROGRAM"
25 ? :? "PRESS KEYS 1-9,0,<,> TO PRODUCE NOTES.";
27 ? :? "RELEASE ONE KEY BEFORE PRESSING THE   NEXT."
28 ? :? "OTHERWISE, THERE MAY BE A DELAY."
30 FOR X=1 TO 37:READ A:CHORD(X)=A:NEXT X
40 FOR X=1 TO 12:READ A:TUNE(X)=A:NEXT X
50 OPEN #1,4,0,"K:"
55 OLDCHR=-1
60 A=PEEK(764):IF A=255 THEN 60
63 IF A=OLDCHR THEN 100
65 OLDCHR=A
70 FOR X=1 TO 12:IF TUNE(X)=A THEN SOUND 0,CHORD(X),10,8:GOTO 100
80 NEXT X
100 I=INT(PEEK(53775)/4):IF (I/2)=INT(I/2) THEN 60
110 POKE 764,255:SOUND 0,0,0,0:OLDCHR=-1:GOTO 60
```

```
200 DATA  243,230,217,204,193,182,173,162,153,144,
136,128,121,114,108,102,96,91,85,81,76,72,68,64,60
210 DATA 57,53,50,47,45,42,40,37,35,33,31,29
220 DATA 31,30,26,24,29,27,51,53,48,50,54,55
```

Chcete-li nahrát "Marie měla jehňátko", stlačujte následující klávesy:

5,3,1,3,5,5,5 3,3,3 5,8,8 5,3,1,3,5,5,5 5,3,3,5,3,1

Poznámka: V řádku 27 vložte mezi THE a NEXT tři mezery.

VYŠŠÍ MATEMATIKA

Váš počítač ATARI je fantastický kalkulátor. Když vložíte do následujícího programu dvě čísla, počítač Vám vypočítá jejich největšího společného dělitele. Vložíte-li například čísla 690 911 a 11214017, zjistíte brzy, že jejich největším společným dělitelem je číslo 53147.

```
10 ? CHR$(125):? "ENTER TWO NUMBERS.PRESS RETURN AFTER EACH ENTRY."
20 INPUT N1,N2
30 GOSUB 90
40 ? "THEIR GCD IS ";:? AN
50 POKE 752,1:POSITION 10,10:? "PRESS START TO CONTINUE."
60 IF PEEK(53279)<>6 THEN GOTO 60
70 POKE 752,0:? CHR$(125):GOTO 10
80 REM ****SUBROUTINE****
90  AN=0:POKE  195,0:TRAP 130:M=(N1>=N2)*N1+(N2>N1)*N2:N=(M=N1)*2+(M=N2)
*N1
100 IF INT(N1)<>N1 OR INT(N2)<>N2 THEN RETURN
110 P=M-INT(M/N)*N:M=N:N=P
120 IF P<>0 THEN GOTO 110
130 AN=M*(PEEK(195)=0):RETURN
```

COMPUTER BLUES

Tento program generuje náhodné hudební tóny a skládá zajímavé melodie na naprogramovaný doprovod.

```
1 GRAPHICS 0:? :? "  COMPUTER BLUES";:?
2 I=R=1
```

```
3 THNOT=1
5 CHORD=1
6 PRINT "BASS TEMPO(1=FAST)";
7 INPUT TEMPO
8 GRAPHICS 2+16:GOSUB 2000
10 DIM BASE(3,4)
20 DIM LOW(3)
25 DIM LINE(16)
26 DIM JAM(3,7)
30 FOR X=1 TO 3
40 FOR Y=1 TO 4
50 READ A:BASE(X,Y)=A
60 NEXT Y
70 NEXT X
80 FOR X=1 TO 3:READ A:LOW(X)=A
90 NEXT X
95 FOR X=1 TO 16:READ A:LINE(X)=A:NEXT X
96 FOR X=1 TO 3
97 FOR Y=1 TO 7
98 READ A:JAM(X,Y)=A:NEXT Y:NEXT X
100 GOSUB 500
110 T=T+1
115 GOSUB 200
120 GOTO 100
200 REM PROCESS HIGH STUFF
205 IF RND(0)<0.25 THEN RETURN
210 IF RND(0)<0.5 THEN 250
220 NT=NT+1
230 IF NT>7 THEN NT=7
240 GOTO 260
250 NT=NT-1
255 IF NT<1 THEN NT=1
260 SOUND 2,JAM(CHORD,NT),10,NT*2
280 RETURN
500 REM PROCESS BASE STUFF
510 IF BASS=1 THEN 700
520 BDUR=BDUR+1
530 IF BDUR<>TEMPO THEN 535
531 BASS=1:BDUR=0
535 SOUND 0,LOW(CHORD),10,4
```

```
540 SOUND 1,BASE(CHORD,THNOT),10,4
550 RETURN
700 SOUND 0,0,0,0
710 SOUND 1,0,0,0
720 BOUR=BOUR+1
730 IF BOUR<>1 THEN 800
740 BOUR=0:BASS=0
750 THNOT=THNOT+1
760 IF THNOT<>5 THEN 800
765 THNOT=1
770 PTR=PTR+1
780 IF PTR=17 THEN PTR=1
790 CHORD=LINE(PTR)
800 RETURN
1000 DATA 162,144,136,144,121,108,102,108,108,96,91,96
1010 DATA 243,182,162
1020 DATA 1,1,1,1,2,2,2,2,1,1,1,1,3,2,1,1
1030 DATA 60,50,47,42,40,33,29
1040 DATA 60,50,45,42,40,33,29
1050 DATA 81,68,64,57,53,45,40
2000 PRINT #6:PRINT #6:PRINT #6
2005 PRINT #6;"  COMPUTER"
2006 PRINT #6
2010 PRINT #6;"   BLUES"
2030 RETURN
```

VLAJKA SPOJENÝCH STÁTŮ

Tento program používá k vytvoření pruhů "přepínání" barev. Používá grafického módu 7+16, takže obrazu je vyhrazena celá obrazovka. Všimněte si vzájemné shody příkazů COLOR a SETCOLOR.

```
10 REM DRAW THE UNITED STATES FLAG
20 REM HIGH RESOLUTION 4-COLOR GRAPHICS,NO TEXT WINDOW
30 GRAPHICS 7+16
40 REM SETCOLOR 0 CORRESPONDS TO COLOR 1
50 SETCOLOR 0,4,4:RED=1
60 REM SETCOLOR 1 CORRESPONDS TO COLOR 2
70 SETCOLOR 1,0,14:WHITE=2
80 REM SETCOLOR 2 CORRESPONDS TO COLOR 3
```

```
90 BLUE=3:REM DEFAULTS TO BLUE
100 REM DRAW 13 RED & WHITE STRIPES
110 C=RED
120 FOR I=0 TO 12
130 COLOR C
140 REM EACH STRIPE HAS SEVERAL HORIZONTAL LINES
150 FOR J=0 TO 6
160 PLOT 0,I*7+J
170 DRAWTO 159,I*7+J
180 NEXT J
190 REM SWITCH COLORS
200 C=C+1:IF C>WHITE THEN C=RED
210 NEXT I
300 REM DRAW BLUE RECTANGLE
310 COLOR BLUE
320 FOR I=0 TO 48
330 PLOT 0,I
340 DRAWTO 79,I
350 NEXT I
360 REM DRAW 9 ROWS OF WHITE STARS
370 COLOR WHITE
380 K=0:REM START WITH ROW OF 6 STARS
390 FOR I=0 TO 8
395 Y=4+I*5
400 FOR J=0 TO 4: REM 5 STARS IN A ROW
410 X=K+5+J*14:GOSUB 1000
420 NEXT J
430 IF K<>0 THEN K=0:GOTO 470
440 REM ADD 6TH STAR EVERY OTHER LINE
450 X=5+5*14:GOSUB 1000
460 K=7
470 NEXT I
500 REM IF KEY HIT THEN STOP
510 IF PEEK(764)=255 THEN 510
515 REM OPEN TEXT WINDOW WITHOUT CLEARING SCREEN
520 GRAPHICS 7+32
525 REM CHANGE COLORS BACK
530 SETCOLOR 0,4,4:SETCOLOR 1,0,14
550 STOP
1000 REM DRAW 1 STAR CENTERED AT X,Y
```

```
1010 PLOT X-1,Y:DRAWTO X+1,Y
1020 PLOT X,Y-1:PLOT X,Y+1
1030 RETURN
```

IGPAY ATINLAY

Tento krátký program komolí slova nebo věty do tzv. pig Latin. Nepoužívejte jednopísmenných slov jako A nebo I.

```
10 DIM A$(256):S=2
20 ? "TYPE  IN A  WORD OR  SENTENCE.PLEASE DON'T  EXCEED THREE LINES OF
TEXT."
30 INPUT A$
40 FOR X=1 TO LEN(A$)
50 IF A$(X,X)=CHR$(32) THEN PRINT A$(S,X-1);A$(S-1,S-1);"AY";" ";:S=X+2
60 IF X=LEN(A$) THEN PRINT A$(S,X);A$(S-1,S-1);"AY"
70 NEXT X
80 ? :? :? "THAT'S ALL FOLKS!"
```

GRAFICKÝ

Napište tento program a pozorujte, co počítač vytvoří.

```
10 DIM A$(35)
20 GRAPHICS 1
25 TRAP 90
30 A$="THIS IS A GRAPHICS DEMONSTRATION."
40 FOR I=1 TO 33:? #6;A$(I,I);
50 S=PEEK(53770)
60 SOUND 0,S,10,14
70 FOR DELAY=0 TO 100:NEXT DELAY
80 NEXT I
90 SOUND 0,0,0,0:END
```

Poznámka: V řádku 30 vložte mezi slova GRAPHICS a DEMONSTRATION dvě mezery.

ESREVER

Název tohoto programu je slovo REVERSE napsané pozpátku. Tento program píše program pozpátku. Po jeho spuštění se objeví na obrazovce otazník. Napište slovo nebo krátkou větu a další nechte na Vašem počítači ATARI.

```
10 DIM A$(180)
20 PRINT "Enter a word or short sentence and press Return."
30 INPUT A$
40 FOR X=LEN(A$) TO 1 STEP -1
50 PRINT A$(X,X);
60 NEXT X
70 PRINT :PRINT ;GOTO 20
```

RACEK NAD OCEÁNEM

Tento program kombinuje zvuky s grafikou. Zvuky nejsou "čisté" napodobují hřmot oceánu a křik racka. Symboly v řádku 20 dostaneme stlačením /CONTROL G/, /CONTROL F/, /CONTROL R/ a /CONTROL R/.

```
10 DIM BIRD$(4)
20 BIRD)="\/--"
30 FLAG=1:ROW=10:COL=10
40 GRAPHICS 1:POKE 756,226:POKE 752,1
50 SETCOLOR 0,0,0:SETCOLOR 1,8,14
60 PRINT #6;"   the ocean"
70 R=INT(RND(0)*11)
80 POSITION 17,17
90 FOR T=0 TO 10
100 SOUND 0,T,8,4
110 FOR A=1 TO 50:NEXT A
120    IF   RND(0)>0.8    THEN   FOR    D=10   TO    5   STEP   -1:SOUND
1,0,10,INT(RND(0)*10):NEXT D:SOUND 1,0,0,0
130 GOSUB 200
140 NEXT T
150 FOR T=10 TO 0 STEP -1
160 SOUND 0,T,8,4
170 FOR A=1 TO 50:NEXT A
175  IF  RND(0)>0.8  THEN  FOR  D=10  TO  5  STEP -1:SOUND 1,D,10,8:NEXT
```

```
D:SOUND 1,0,0,0
180 FOR H=1 TO 10:NEXT H
185 GOSUB 200
190 NEXT T
195 GOTO 70
200 GOSUB 300
210 POSITION COL,ROW
220 PRINT #6;BIRD$(FLAG,FLAG+1)
230 FLAG=FLAG+2:IF FLAG=5 THEN FLAG=1
240 RETURN
300 IF RND(0)>0.5 THEN RETURN
310 POSITION COL,ROW
320 PRINT #6;"  "
330 A=INT(RND(0)*3)-1
340 B=INT(RND(0)*3)-1
350 ROW=ROW+A
360 IF ROW=0 THEN ROW=1
370 IF ROW=20 THEN ROW=19
380 COL=COL+B
390 IF COL=0 THEN COL=1
400 IF COL>18 THEN COL=18
410 RETURN
```

Poznámka: V řádku 320 musí být mezi uvozovkami dvě mezery.

POHYBOVÉ UMĚNÍ

Tento program vytváří neustále se měnící a po obrazovce rozprostírající se čáry s měnícími se barvami.

```
10 REM KINETIC ART BY NEIL HARRIS
20 GRAPHICS 10
30 DIM A(3,50)
35 FOR L=0 TO 3:FOR M=0 TO 50:A(L,M)=0:NEXT M:NEXT L
40 HUE=INT(RND(1)*8+1):POKE 704+HUE,INT(RND(1)*8)*16+INT(RND(1)*4+4)
50 X1=INT(RND(1)*80):X2=INT(RND(1)*80):Y1=INT(RND(1)*192):Y2=INT(RND(
1)*192)
60 COLOR 0:PLOT A(0,WHICH),A(1,WHICH):DRAWTO A(2,WHICH),A(3,WHICH)
70 BOUNCE=BOUNCE-1:IF BOUNCE>0 THEN 90
80 BOUNCE=INT(RND(1)*10+10):BX1=INT(RND(1)*9-4):BX2=INT(RND(1)*9-4):BY1=
```

```
INT(RND(1)*13-6):BY2=INT(RND(1)*13-6)
90 CHANGE=CHANGE-1:IF CHANGE>0 THEN 110
100 CHANGE=INT(RND(1)*10+5):HUE=INT(RND(1)*8+1):POKE 704+HUE,INT(RND(1)
*256)
110 COLOR HUE:PLOT X1,Y1:DRAWTO X2,Y2
120 A(0,WHICH)=X1:A(1,WHICH)=Y1:A(2,WHICH)=X2:A(3,WHICH)=Y2
130 WHICH=WHICH+1:IF WHICH>50 THEN WHICH=0
140 X1=X1+BX1:IF X1<0 OR X1>79 THEN BX1=-BX1:GOTO 140
150 X2=X2+BX2:IF X2<0 OR X2>79 THEN BX2=-BX2:GOTO 150
160 Y1=Y1+BY1:IF Y1<0 OR Y1>191 THEN BY1=-BY1:GOTO 160
170 Y2=Y2+BY2:IF Y2<0 OR Y2>191 THEN BY2=-BY2:GOTO 170
180 GOTO 60
```

Rozmer. V Gr

Přemýšleli jste někdy o tom, jak ochránit Vaše programy proti zvědavým očím a nenechavým prstům? Zde je několik typů.

Nejdříve napište tento program:

```
10 FOR X=1 TO 50:POKE 710,X:NEXT X:GOTO 10
```

Přidáte-li k tomuto programu další řádek, uvedený níže, vyřadíte z funkce klávesu /BREAK/. Tento řádek brání přerušení běžícího programu a jeho vypisování. Nebo máte-li navržen program, který vyžaduje vstup z klávesnice, chrání vyřazení klávesy /BREAK/ před přerušením programu neúmyslným stlačením klávesy /BREAK/.

Vymažte z řádku 10 instrukci GOTO 10 a přidejte další řádek:

```
20 POKE 16,64:POKE 53774,64:GOTO 10
```

Spusťte nyní Váš nový program a zkuste jej stlačením klávesy /BREAK/ zastavit. Nejde to.

Aby příkazy POKE byly účinné, musí být vloženy v programu po každém příkazu grafického módu.

Program můžete zastavit a poté vypsat nejen klávesou /BREAK/, ale také klávesou /RESET/. Tomu lze zabránit přidáním následujícího řádku k Vašemu programu:

```
5 POKE 580,1
```

Nyní když někdo stlačí /RESET/, vymaže se program z paměti. Žádný program - žádný výpis. Příkaz POKE by vždy měl být na začátku Vašeho programu.

ČÁST II

ATARI BASIC

Operátory a jejich priorita

Jako první jsou prováděny operace v nejvnitřnějších závorkách. Pak se pokračuje dalšími "nejvnitřejšími" závorkami. Závorkám, které jsou uzavřeny v jiných závorkách, se říká "vložené" závorky. Operace na téže úrovni vložených závorek jsou prováděny v následujícím pořadí:

Od nejvyšší k nejnižší priorité

<,>, =,<=,>=,<>
Relační operátory, používané v řetězcových výrazech, mají tutéž prioritu a jsou vykonávány zleva doprava.

-
Mínus (označuje záporné číslo).

^
Umocňování.

*,/
Násobení a dělení má tutéž prioritu a je vykonáváno zleva doprava.

+,-
Sčítání a odčítání má tutéž prioritu a je vykonáváno zleva doprava.

<,>,=,<=,>=,<>
Relační operátory v numerických výrazech mají tutéž prioritu a jsou vykonávány zleva doprava.

NOT
Negace

AND
Logický součin

OR
Logicky součet

Řízení systému

(Přípustné zkratky jsou uvedeny v závorce.)

Následující slova mohou být použita jako programové instrukce, nebo jako přímé příkazy bez čísla řádku se stlačením RETURN.
Tato slova nesmějí být použita v názvech proměnných.

BYE (B.)

Přechází z BASICu do funkce SELF TEST (samočinný test).

DOS

Zobrazuje menu DOS (používá se pouze s disketovou jednotkou).

CSAVE (CS.)

Ukládá na kazetu programový soubor.

CLOAD

Zavádí programový soubor z kazety do počítače.

SAVE (S.)

Ukládá program v jazyce BASIC na výstupní zařízení.
Příklad: SAVE "D: MYFILE.BAS"

LOAD (LO.)

Zavádí programový soubor ze vstupního zařízení do počítače.
Příklad: LOAD"D:MYFILE.BAS"

LIST (L.)

Posílá na obrazovku nebo na výstupní zařízení výpis programu.
Příklad: LIST (vypisuje celý program)
LIST 10 (vypíše na obrazovku řádek 10)
LIST 10,20 (vypíše vše od řádku 10 po řádek 20)
LIST"P:" (vypisuje na tiskárnu)
LIST"P:",10,20 (vypíše na tiskárnu vše od řádku 10 do řádku 20)
LIST"D:MYFILE.LST" (vypisuje na disketu)
LIST"D:MYFILE.LST",10,20 (vypisuje na disketu řádky 10-20)
LIST"C:" (vypisuje na kazetu)

ENTER (E.)

Zavádí program z výpisu na vstupní zařízení
Příklad: ENTER"C:"
ENTER"D:MYFILE.LST"
Poznámka: Řádky téhož čísla v již zavedeném programu budou přepsány řádky zaváděného programu.

NEW

Maže z paměti program.

RUN

Spouští vykonávání BASIC programu. Program může být v paměti nebo zaváděn z diskety nebo kazety. (Nastavuje proměnné na nulu a ruší dimenzování polí a stringů.)
Příklad: RUN (spouští program, uložený v paměti)
RUN"D:MYFILE.BAS" (zavádí program z diskety a spouští jej)

CONT

Spouští program, jehož průběh byl přerušen klávesou BREAK, nebo který byl zastaven příkazem STOP nebo END. Program je spuštěn od následující řádky v programu. Příkazy v téže řádce, ve které došlo k zastavení programu, nejsou vykonány.

SOUND(SO.)

Nastavuje parametry jednoho ze čtyř tónových generátorů, jejichž zvuk můžete slyšet z reproduktorů televizoru. Zvuk, nastavený tímto příkazem trvá do té doby, než přijde další instrukce SOUND adresovaná témuž generátoru, nebo až je vykonán příkaz END, RUN, nebo NEW. Generátory jsou programovány nezávisle a mohou být ve funkci všechny najednou. Tato instrukce musí být následována čtyřmi parametry (čísla, proměnné nebo výrazy).

Příklad: SOUND A,B,C,D

kde:

A...číslo generátoru (0 - 3)

B...perioda (0 - 255) Čím vyšší hodnota, tím nižší frekvence tónu. Frekvence = 31 960/(perioda + 1)
Viz tabulku přiřazení tónů.

C...zkreslení (0 - 14, pouze sudé hodnoty). Čisté tóny jsou pouze 10 a 14. Ostatní čísla představují šumy.

D...hlasitost (0 - 15). Čím vyšší číslo, tím vyšší hlasitost. 0 znamená vypnuto. Přesahuje-li úhrnná hlasitost všech čtyř zvuků 32, může dojít při reprodukci k přebuzení reproduktoru a ke zkreslení reprodukce.

Přiřazení tónů

Tón	perioda	Tón	perioda	Tón	perioda
C	243	Gis	153	E	96
Cis	230	A	144	F	91
D	217	B	136	F	85
Dis	204	H	128	G	81
E	193	C1	121	Gis	76
F	182	Cis	114	A	72
Fis	173	D	108	B	68
G	162	Dis	102	H	64

C2	60	E	47	Gis	37
Cis	57	F	45	A	35
D	53	Fi	42	B	33
Dis	50	G	40	H	31
				C3	29

Programové řízení

GOTO(G.)

Provádění programu pokračuje na určeném řádku.

Příklad: GOTO 30 (pokračuj v programu od řádku 30).
GOTO A+10 (přípustné, ale může být těžko odladitelné).

ON...GOTO

Provádění programu pokračuje na určeném řádku, v závislosti na hodnotě proměnné nebo výrazu za ON.

Příklad: ON A GOTO 10,300,50 nahrazuje příkazy:
IF A=1 THEN GOTO 10
IF B=2 THEN GOTO 300
IF C=3 THEN GOTO 50

GOSUB(GOS.)

Provádění programu pokračuje na určeném řádku. Instrukce RETURN vrátí řízení programu na instrukci, následující po GOSUB.

Příklad: GOSUB 30 (pokračuj podprogramem na řádku 30).
GOSUB A+10 (přípustné, ale může být velice těžko odladitelné).

ON...GOSUB

Provádění programu pokračuje na určeném řádku, v závislosti na hodnotě proměnné nebo výrazu za ON.
Instrukce RETURN vrátí řízení programu na instrukci následující po

ON...GOSUB.

Příklad: ON A+1 GOSUB 10,300,50
(IF A+1=1 THEN GOSUB 10
IF A+1=2 THEN GOSUB 300
IF A+1=3 THEN GOSUB 50)

Upozornění: Je-li hodnota výrazu menší než 1 nebo větší než počet řádků uvedených za GOSUB, je výsledek nepředvídatelný.

RETURN(RET.)

Ukončuje program a vrací řízení na instrukci bezprostředně následující za posledně vykonanou instrukcí GOSUB.

Příklad: RETURN

FOR(F.)

Tato instrukce nastavuje počáteční a konečnou hodnotu proměnné a hodnotu, která má být při každém průchodu cyklem FOR...NEXT přičtena. Přičítaná hodnota je standardně 1, jinak se musí specifikovat instrukcí STEP (krok). Instrukce NEXT přičítá určenou hodnotu a způsobuje opakování cyklu FOR...NEXT, v případě, že proměnná nepřesáhla svou určenou konečnou hodnotu.

Příklad: FOR A=1 TO 10 (A bude začínat s hodnotou 1, při každém průchodu cyklem se zvětší o 1 a skončí na 10).
FOR A=10 TO Q STEP - 2 (záporný krok).
FOR A=B/T TO B*T STEP X (rovněž platný zápis).

NEXT(N.)

Ukončuje cyklus FOR...NEXT. Kontroluje, zda hodnota parametru nepřesáhla konečnou hodnotu, přičítá hodnotu kroku k hodnotě parametru a vrací řízení na instrukci, následující po FOR. Přesáhl-li parametr konecnou hodnotu, přechází řízení na instrukci následující po NEXT.

Příklad: NEXT A (A je tentýž parametr jako u FOR).

POP

Ruší návratovou informaci příslušející poslední vykonané instrukci FOR nebo GOSUB. Užívá se pro předčasné opuštění cyklu FOR - NEXT, nebo pro odskok z programu bez průchodu instrukcí RETURN.

Příklad: POP:GOTO 10

IF...THEN

Při splnění podmínek mezi IF a THEN je vykonána instrukce, která následuje za THEN. Jinak řízení přechází na následující řádek programu.

Příklad: IF A C THEN GOTO 300
IF A=B THEN PRINT "A=B":PRINT"HOW ABOUT THAT!":
LET A=5:GOTO 20
IF A THEN PRINT "A JE NENULOVE!" (instrukce za THEN je vykonána, když A je nenulové).

TRAP(T.)

Vyskytne-li se chyba - error - přechází řízení na určený řádek programu. TRAP zůstává nastaveno, dokud chyba trvá, nebo dokud nepřijde další instrukce TRAP. Chybový kód obdržíte pomocí instrukce PEEK(195).

Příklad: TRAP 30 (v případě chyby jdi na řádek 30).
TRAP 40000 (číslo vyšší než 32 767 ruší účinek předchozí instrukce TRAP).

STOP

Zastavuje běh programu a vytiskne číslo řádku, na kterém byl program zastaven. Neuzavírá soubor ani nevypíná zvuk. Program může být znovu spuštěn příkazem CONT.

Příklad: IF A=B THEN STOP

END

Zastavuje běh programu, uzavírá všechny otevřené soubory, vypíná zvuk.

Vstupy a výstupy

Názvy vstupních/výstupních zařízení

V systému ATARI má každé zařízení svoje samostatné jméno. Disketové jednotky a ATARI 850 Interface moduly (manipulační program RS232) vyžadují ještě číslo zařízení (1 - 4). Disketové jednotky dále požadují jména souboru. Jména souboru musí být v uvozovkách, nebo obsažena v řetězcových proměnných. Zde je několik příkladů:

K: Klávesnice. Pouze vstup.
P: Tiskárna. Pouze výstup.
C: Magnetofon. Vstup i výstup.
S: Televizní obrazovka. Pouze výstup.
E: Obrazový editor (klávesnice kombinovaná s obrazovkou). Vstup i výstup.
R: Manipulační program RS232 (Interface modul ATARI 850). Vstup i výstup.
D: JMENOSOU.PRO Obslouží soubor "JMENOSOU.PRO" na disketové jednotce #1.
D2:JMENOSOU.PRO Totéž pro disketovou jednotku #2.

Jména souborů na disketě začínají písmenem a mohou obsahovat až 8 znaků. Jméno souboru může být prodlouženo o nepovinné prodloužení (extender). Prodloužení je odděleno od jména tečkou a může obsahovat až 3 znaky - libovolná písmena nebo číslice.

Příklady některých praktických prodloužení (extendrů):

.BAS programy BASIC uložené pomocí SAVE
.LST výpisy programů BASIC, uložené pomocí LIST
.DAT datové soubory
.OBJ soubory ve strojovém kódu
.TXT textové soubory

Vstupní/výstupní instrukce

OPEN(0.)

Připravuje zařízení pro vstup nebo výstup. Čísla vstupních/výstupních

řídicích bloků (IOCB) jsou 1 - 7. Používají následujícího kódování.

Typ	Kód	Činnost
vstup	4	pouze čtení
výstup	8	pouze zápis
aktualizace	12	čtení i zápis
připisování	9	připisování na konec souboru
adresář	6	pouze adresář disketové jednotky

```
Příklad: OPEN #1,4,0,"K:"(Otevři klávesnici pro vstup na IOCB #1.)
         OPEN #2,8,0,"P:"(Otevři tiskárnu pro výstup na IOCB #2.)
         OPEN #1,12,0,"D:MYFILE.DAT"(Otevři diskový soubor
                        "MYFILE.DAT" pro aktualizaci na IOCB #1.)
         OPEN  #1,6,0,"D:*.*" (Otevři  disketovou  jednotku  pro
                          adresář diskety na IOCB #1.)
```

CLOSE(CL.)

Po vstupní/výstupní operaci uzavírá zařízení a uvolňuje IOCB. Čísla IOCB jsou, stejné jako u OPEN, 1 - 7.

```
Příklad: CLOSE #1 (Uzavírá soubor otevřený na IOCB #1 a uvolňuje IOCB.)
```

INPUT(I.)

Obstarává data ze vstupního zařízení. Data musí být ukončena znakem RETURN.

```
Příklad: INPUT A (Obdrží číslo a uloží jej do A.)
         INPUT A,B,C (Obdrží tři čísla oddělená čárkami a uloží je do A,
                  B a C.)
         INPUT  A$ (Obdrží řetězec  znaků a uloží  jej do A$.  A$ nebude
                  obsahovat znak RETURN.)
         INPUT #1,A$,B (Obdrži řetězec  znaků a číslo z  IOCB #1 a uloží
                  je do A$ a B.)
```

PRINT(PR.) nebo (?)

Posílá data na obrazovku nebo na jiné zařízení.

Příklad: PRINT Pošle prázdný řádek.
PRINT"A=";A (Vytiskne text v uvozovkách a obsah proměnné A do jednoho řádku bez mezery.)
PRINT"CISLO",A (Vytiskne text v uvozovkách a obsah proměnné do jednoho řádku; čárka místo středníku způsobí rozdělení tisku do sloupců. Šíře sloupců je standardně 10 znaků, při požadované šíři K znaků použité instrukci POKE 201,K.)
PRINT A$; (Další tisk bude umístěn do téhož řádku těsně za obsah A$.)
PRINT #1,A$ (Pošle A$ do zařízení otevřeného na IOCB #1.)

LPRINT(LP.)

Tiskne data na tiskárnu. Tiskárnu není třeba otevírat ani zavírat. Narozdíl od instrukce PRINT nezabraňuje středník na konci řádku odesílání řádku příkazem RETURN.

Příklad: LPRINT (Odešle na tiskárnu prázdný řádek.)
LPRINT A$ (Vytiskne obsah A$.)
LPRINT A$;B (Vytiskne obsah B hned za obsahem A$.)
LPRINT A$,B (Obsah B bude vytištěn do samostatného sloupce. Standardní šíře sloupců je 10 znaků. Při požadované šíři K znaků použijte instrukci POKE 201, K.)

GET

Vybírá s určeného zařízení jeden byte a ukládá jej do určené proměnné.

Příklad: GET #1,A (Vybírá byte ze zařízení, otevřeného na IOCB #1 a ukládá jej do proměnné A.)

PUT

Posílá jeden byte z určené proměnné do určeného zařízení.

Příklad: PUT #1,A (Posílá byte, který je obsahem A do zařízení, otevřeného na IOCB #1.)

NOTE(NO.)

Používá se při práci s disketou k určení oblasti, z níž bude čten, nebo do níž bude zapisován další byte.

Příklad: NOTE #1,SEC,BYTE (Odešle číslo sektoru SEC a číslo byte BYTE. Vztahuje se k souboru otevřenému na IOCB #1.)

POINT(P.)

Používá se při práci s DOS k určení oblasti, z níž bude čten, nebo do níž bude ukládán následující byte.

Příklad: POINT #1,SEC,BYTE (Ukáže DOS na sektor číslo SEC a byte číslo BYTE.)

STATUS(ST.)

Čte stav určeného zařízení. Obdržený stavový kód může být nalezen v seznamu chybových hlášení (error code).

Příklad: STATUS #1,A (Čte stav zařízení, otevřeného na IOCB #1 a ukládá jej do A.)

Zpracovatelské instrukce

LET

Přiřazuje hodnoty numerickým nebo řetězcovým proměnným.

Příklad: LET A=B (Hodnota B je přiřazena A.)
LET A$="NAZDAR"
A=B:A$="NAZDAR" (LET může být vynecháno.)

POKE

Ukládá čísla v rozmezí 0 až 255 do určené oblasti paměti v rozmezí 0 až

65535. Opačnou funkcí má PEEK, používá se ke čtení určené oblasti paměti.
Desetinná čísla budou zaokrouhlena.

Příklad: POKE 82,0 (Uloží 0 na adresu 82.)
A=PEEK(82) (Čte obsah paměti na adrese 82 a ukládá jej do proměnné A.)

DIM

Vyhražuje v paměti prostor pro číselná a řetězcová pole. Pro každý znak řetězce vyhražen 1 byte; pro každé číslo je vyhraženo 6 byte.

Příklad: DIM A$ (10) (Řetězcová proměnná o délce 10 znaků.)
DIM B (10) (Číselné pole; B obsahuje deset elementů.)
DIM B (10,10) (Dvourozměrné pole 10 krát 10 elementů.)
DIM A$ (10),B (10) (Různé proměnné oddělte čárkou.)

COM

Totéž jako DIM.

CLR

Ruší vyhražení všech polí, maže řetězcové proměnne, nuluje číselné proměnné (opak DIM).

Příklad: CLR

DATA(D.)

Vytváří seznam čísel a (nebo) písmen, který se použije v instrukci READ.

Příklad: DATA 1,2,3,4,A,B,C,D (Seznam informací, které mají být čteny.)

READ

Čte následující položku v příkladu DATA a přiřazuje ji proměnné. Byla-li data obsažena v instrukci DATA přečtena, pokračuje READ ve čtení

dat v další instrukci DATA v programu.

Příklad: READ A (Proměnné A bude přiřazena další položka v instrukci DATA.)
READ A$ (Lze použít i pro řetězce.)
READ A,A$,B,B$ (Více položek oddělte čárkami.)

RESTORE(RES.)

Používá se v souvislosti s instrukcemi READ a DATA. Ukazuje na instrukci DATA.

Příklad: RESTORE (Následující položka dat bude první položkou v prvním příkazu DATA.)
RESTORE 10 (Následující položka dat bude první položkou v instrukci DATA na řádku 10.)

REM(R.) nebo ([mezera])

Umožňuje dělání poznámek pro lepší orientaci v programu. BASIC ignoruje vše od REM až do konce řádku.

Příklad: REM POZNAMKA

G r a f i k a

GRAPHICS GR

Volí grafický mód. Celkem je přístupno 16 módů (0 - 15). Mód +16 rezervuje celou obrazovku pro grafiku (žádné textové okénko), mód +32 nemaže obrazovku.

Příklad: GRAPHICS 8 (Grafický mód 8 s textovým okénkem.)
GRAPHICS 8+16 (Mód 8 bez textového okénka.)
GRAPHICS 8+32 (Nemaže obrazovku.)
GRAPHICS 8+16+32 (Kombinace obou.)

Tabulka módů a formátů obrazu

Graf. mód	typ módu	sloupců	řád- dělená obraz.	řád- celá obraz.	počet barev	požadov. paměť RAM (byte) dělená	celá obr.
0	text	40	-	24	1-1/2	-	992
1	text	20	20	24	5	674	672
2	text	20	10	12	5	424	420
3	grafika	40	20	24	4	434	432
4	grafika	80	40	48	2	694	696
5	grafika	80	40	48	4	1174	1176
6	grafika	160	80	96	2	2174	2184
7	grafika	160	80	96	4	4190	4200
8	grafika	320	160	192	1-1/2	8112	8138
9	grafika	80	-	192	1	- -	8138
10	grafika	80	-	192	9	- -	8138
11	grafika	80	-	192	16	- -	8138
12	grafika	40	20	24	5	1154	1152
13	grafika	40	10	12	5	664	660
14	grafika	160	160	192	2	4270	4296
15	grafika	160	160	192	4	8112	8138

SETCOLOR(SR.)

Nastavuje odstín a jas zvoleného barvového registru. Číslo registru není totožné s číslem v příkazu COLOR.

Příklad: SETCOLOR 1,2,4 (Nastavuje registr 1 na barevný odstín 2 a jas 4.)

Registry 0 - 4
Barevné odstíny 0 - 15
Jas 0 - 14 pouze sudá čísla (s výjimkou módů GTIA, které používají sudá i lichá čísla až do 15.)

COLOR(C.)

V mřížových módech 3 - 11 volí barvový registr pro použití v instrukci

PLOT. Tyto registry nejsou totožné s registry v instrukci SETCOLOR.

Příklad: COLOR 2 (Volí barvový registr 2 nebo znak ASCII, jehož hodnota je 2.)

PLOT(PL.)

Ukládá do určeného místa na obrazovce obrazový element nebo znak.

Příklad: PLOT X,Y (X a Y musí být kladné souřadnice.)

POSITION(POS.)

Určuje místo na obrazovce, používá se v souvislosti s instrukcí PRINT, která vytiskne text na místo, určené instrukcí POSITION.

Příklad: POSITION X,Y

LOCATE(LOC.)

Vyhledává data, uložená na určeném místě obrazovky. V módech 0 - 2 předává znaky, v módech 3 - 11 čísla barev.

Příklad: LOCATE X,Y,D (Ukládá data, příslušející souřadnicím X,Y na obrazovce, do proměnné D.)

DRAWTO(DR.)

Kreslí úsečku mezi poslední polohou kurzoru a určenými souřadnicemi X a Y. Kurzor je po vykonání instrukce na nových souřadnicích X, Y.

Příklad: DRAWTO X,Y (Kreslí úsečku z místa poslední polohy kurzoru do místa o souřadnicích X,Y.)

Tabulka standardních barev

Tyto barvy jsou nastaveny, není-li jinak určeno instrukcí SETCOLOR.

Barvový registr (Color register)	standardní barva	jas	barva
0	2	8	oranžová
1	12	10	zelená
2	9	4	tmavě modrá
3	6	6	růžová nebo červená
4	0	0	černá

Barevné odstíny ATARI

Nastavují se instrukcí SETCOLOR.

Barva	Číslo (2.parametr v SETCOLOR)
šedá	0
světle oranžová (zlatá)	1
oranžová	2
červeno-oranžová	3
růžová	4
purpurová	5
purpurovo-modrá	6
modrá	7
modrá	8
světle modrá	9
tyrkysová	10
zeleno-modrá	11
zelená	12
žluto-zelená	13
oranžovo-zelená	14
světle oranžová	15

Poznámka: Barevné odstíny závisí na typu nastavení televizoru nebo monitoru.

Funkce

Funkce přiřazují jedné nebo několika hodnotám jinou hodnotu. Hodnotami mohou být buď řetězce (znaky) nebo čísla. V příkladech je použito A jako číselné proměnné a A$ jako řetězcové proměnné. Funkci můžete použít

téměř všude, kde se používají hodnoty (včetně uvnitř jiných funkcí).

Aritmetické funkce

ABS

Absolutní hodnota čísla.

Příklad: A=ABS (B)

CLOG

Dekadický logaritmus.

Příklad: A=CLOG (B)

EXP

Exponenciální funkce se základem "e" (přibližně 2.718). V některých případech je počítána s přesností na 6 platných cifer. Je inverzní k funkci LOG.

Příklad: A=EXP (B)

INT

Nejvyšší celé číslo, které je menší nebo rovno výchozí hodnotě.

Příklad: A=INT (B)

LOG

Přirozený logaritmus. Je inverzní k EXP.

Příklad: A=LOG (B)

RND

Náhodné číslo mezi 0 a 1. Nikdy nenabývá hodnoty 1.

Příklad: A=RND (0) (A je číslo větší nebo rovno nule a menší než 1.)
A=RND (0)*8 (A je číslo větší nebo rovno nule a menší než 8.)

SGN

-1, je-li číslo záporné, nula, je-li číslo 0 a 1, je-li číslo kladné.

Příklad: A=SGN (A)

SQR

Druhá odmocnina kladného čísla.

Příklad: A=SQR (B)

T r i g o n o m e t r i c k é f u n k c e

ATN

Arctangens argumentu v radiánech nebo stupních.
Příklad: A=ATN (B)

COS

Cosinus
Příklad: A=COS (B)

DEG

Všechny následující trigonometrické funkce budou ve stupních.
Příklad: DEG

RAD

Všechny následující funkce budou v radiánech. Počítač předpokládá, že

požadujete údaje v radiánech. Chcete-li údaje ve stupních, musíte použít funkce DEG.
Příklad: A=SIN (B)

TAN

Tangens.
Příklad: A=TAN (B)

Speciální funkce

ADR

Desítková adresa začátku řetězce.
Příklad: A=ADR (B)
A=ADR ("TOHOTO RETEZCE")

FRE

Počet volných byte v uživatelské paměti RAM. Požadovaná je formální proměnná (0).
Příklad: A=FRE (0)

PEEK

Obsah určeného místa paměti. Adresa paměti musí být číslo v rozmezí 0 až 65535. Hodnota funkce je mezi 0 až 255.
Příklad: A=PEEK (B)

USR

Volá z BASICu podprogramy ve strojovém kódu. Například USR (ADDR, P1, P2, ...PN) ukládá argumenty P1 až PN v obráceném pořadí do sklípku. Nejprve je uložen argument PN.

Jako poslední je do sklípku uložen počet argumentů - reprezentován jedním byte. Nejsou-li ve funkci určeny žádné argumenty, je uložena do sklípku nula. Pak je volán podprogram ve strojovém jazyku na adrese

(ADDR). Má-li podprogram ve strojovém kódu vrátit hodnotu v BASICu, musí být horní a spodní byte uloženy v paměti na adrese $D4 resp. $D5.

Před návratem do BASICu musí program odstranit argumenty a jejich počet, jinak se systém zhroutí.

Příklad: A=USR (8,C,D) (Je volán program na adrese 8, parametry C a D budou v podprogramu uplatněny přes sklípek.)

Řetězcové funkce

ASC

Desitkové číslo v kódu ATASCII prvního znaku v řetězci.

Příklad: A=ASC ("A") (A bude 65.)
A=ASC (B$) (Mohou být použity řetězcové proměnné.)

CHR$

Znak, příslušející určenému číslu v ATASCII kódu. Inversní k ASC.

Příklad: A$=CHR$ (65) (A$ bude "A".)

LEN

Číslo, představující délku řetězcové proměnné.

Příklad: A=LEN (A$)

STR$

Řetězec představující určenou hodnotu. Převádí číslo na řetězcovou proměnnou.

Příklad: A$=STR$ (65) (A$ bude "65", což nyní není číslo, ale řetězec.)

VAL

Číslo, reprezentující specifikovaný řetězec. (Převádí řetězec na číslo.)

Příklad: A=VAL ("100") (A je rovno číslu 100.)

Manipulace s řetězcovými proměnnými

ATARI BASIC nepožívá při práci s řetězci formátu řetězcového pole. Může však vybírat z řetězce dílčí řetězec, provádět spojování řetězců a hledat v řetězci dílčí řetězec.
Příklady: Dílčí řetězec.

```
50 A$="OLDADAVIDMICHAL"
60 B$=A$ (10, 15) (B$ je " MICHAL")
```

Spojování.

```
50 A$="AHOJ"
60 B$="FREDE"
70 A$ (LEN(A$)+1)=B$      (A$ je "AHOJ FREDE")
```

Hledání.

```
50 FOR Z=1 TO LEN (A$)
60 IF A$ (Z,Z) = "E" THEN PRINT"EMIL"
70 NEXT Z
```

Funkce ovladačů her

PADDLE

Udává polohu ovladače "paddle". Ovladače jsou očíslovány zpředu dozadu 0-3. Udávané číslo leží mezi 1 a 228, zvětšuje se při otáčení knoflíku doleva (proti směru hodinových ručiček).
Příklad: A=PADDLE (0)

PTRIG

Nabývá hodnoty 0 při stlačení tlačítka ovladače "paddle" a hodnoty 1, není-li tlačítko stlačeno. Ovladače jsou očíslovány zpředu dozadu 0-3.

Příklad: A=STRIG (0)

STICK

Udává stav ovladače "joystick". Ovladače jsou číslovány zpředu dozadu 0-1. Podrobnější viz diagram na str. 10 orig. návodu.
Příklad: A=STICK (0)

STRIG

Nabývá hodnoty 0 při stlačení spouštěcího tlačítka ovladače "joystick" a hodnoty 1, není-li tlačítko stlačeno. Ovladače jsou číslovány 0 a 1.
Příklad: A=STRIG (0)

K l á v e s y s p e c i á l n í c h f u n k c í

ESC

Použití závisí na programu. V některých programech se používá pro přechod z jednoho menu do druhého. Ve spojitosti s tiskárnou se používá pro tisk mezinárodních znaků.

BREAK

Zastavuje běh programu BASIC.

RESET

Zastavuje běžící program, nastavuje grafický mód 0, maže obrazovku, ukončuje, ale neuzavírá soubory; nemaže program.

SET - CLR - TAB

Posouvá kurzor na následující nastavenou značku.

SHIFT SET - CLR - TAB

Nastavuje značku.

CTRL SET - CLR - TAB

Maže značku.

CTRL 1

Zastavuje a znovu spouští cyklický posuv řádků na obrazovce.

CTRL 2

Rozezní tón podobný bzučáku.

CTRL 3

Vytváří označení "konec souboru" (EOF) při vstupu z klávesnice.

U p r a v o v á n í

SHIFT INSERT

Vkládá mezeru pro řádek.

CTRL INSERT

Vkládá mezeru pro znak.

DELETE BACK SPASE

Vypouští znak nalevo od kurzoru a posouvá kurzor na jeho místo.

SHIFT DELETE BACK SPACE

Vypouští řádek.

CTRL DELETE BACK SPACE

Vypouští znak v místě kurzoru a posouvá zbytek řádku o jedno místo

vlevo.

SHIFT CLEAR nebo CTRL CLEAR

Maže obrazovku.

CTRL ŠIPKA VZHŮRU

Posouvá kurzor nahoru.

CTRL ŠIPKA DOLŮ

Posouvá kurzor dolů.

CTRL ŠIPKA VLEVO

Posouvá kurzor vlevo.

CTRL ŠIPKA VPRAVO

Posouvá kurzor vpravo.

Chybová hlášení

Chybový kód	Obsah kódu
2	Nepostačující paměť: V paměti RAM není již místo k uložení další instrukce nebo další proměnné, nebo k dimenzování nové řetězcové proměnné.
3	Chyba v hodnotě: Hodnota čísla je záporná, zatímco je očekáváno kladné celé číslo; hodnota není v určeném rozsahu.
4	Příliš mnoho proměnných: Bylo překročeno maximální množství 128 různých názvů proměnných.
5	Chyba v délce řetězce: Uživatel se pokouší vložit řetězcovou proměnnou, jejíž délka přesahuje dimenzovanou délku.
6	Mimo data: Instrukce READ požaduje více položek dat, než poskytuje instrukce DATA.
7	Číslo řádku větší než 32767: Použité číslo řádku je větší než 32767.
8	Chyba ve vstupní instrukci: Uživatel se snaží vložit do číselné proměnné nečíselnou hodnotu.
9	Chyba v dimenzování pole nebo řetězce: Rozsah dimenzovaných číselných polí přesahuje 5460 nebo rozsah dimenzovaných řetězcových proměnných přesahuje 32767; pole nebo řetězcová proměnná byla znovu dimenzována; pole nebo řetězec byly dimenzovány.
10	Přetečení zásobníku.

11 Přetečení nebo podtečení v pohyblivé řádové čárce: Uživatel se pokouší dělit nulou, použít čísla, většího než 1×10^{98}, nebo menšího než 1×10^{-99}.

12 Řádek nenalezen: Instrukce GOSUB, GOTO nebo THEN volají neexistující číslo řádku.

13 Nesdružení instrukcí FOR-NEXT: Program narazil na instrukci NEXT bez předchozí instrukce FOR, nebo včleněné instrukce FOR-NEXT nejsou správně sdruženy. (Zpráva o chybě je vydávána u instrukce NEXT, nikoliv FOR.)

14 Chyba v délce řádku: Instrukce je příliš složitá nebo příliš dlouhá pro zpracování v BASICu.

15 Vynechán řádek GOSUB nebo FOR: Program narazil na instrukci RETURN nebo NEXT, avšak odpovídající instrukce GOSUB nebo FOR byla po posledním příkazu RUN vynechána.

16 Chyba v návratu: Program narazil na instrukci RETURN bez odpovídající instrukce GOSUB.

17 Syntaktická chyba: Počítač narazil na řádek s syntaktickou chybou.

18 Neplatný znak řetězce: Řetězec v instrukci VAL není číselným řetězcem.

Poznámka:

Následující chyby jsou vstupní/výstupní chyby, které se mohou vyskytnout při práci s disketovými jednotkami, tiskárnami, nebo jinými periferními zařízeními. Další informace jsou poskytovány s příslušným hardware.

19 Příliš dlouhý zaváděný program.

Paměť nestačí k zavedení celého programu.

20 Chyba v čísle zařízení: Číslo zařízení je větší než 7 nebo je rovno 0.

21 Chyba v zaváděném souboru: Uživatel se pokouší zavést do paměti počítače soubor, který není kompatibilní s BASICem. Soubor, který lze zavést, byl vytvořen pomocí příkazu SAVE.

128 Chybné BREAK: Uživatel během vstupní / výstupní operace zavadil o tlačítko BREAK.

129 IOCB[1] již otevřen: Vstupní / výstupní řídící kanál je již otevřen.

130 Neexistující zařízení: Uživatel se pokouší dosáhnout nedefinovaného zařízení (zařízení není v tabulce manipulačního programu).

131 IOCB[1] pouze pro zápis: Příkaz READ byl poslán na zařízení, určené pouze pro zápis (tiskárna).

132 Neplatný příkaz: Pro toto zařízení je příkaz neplatný.

133 Zařízení nebo soubor nebyl otevřen: Zařízení nebylo otevřeno příkazem OPEN.

134 Špatné číslo IOCB[1]: Číslo zařízení je neplatné.

135 IOCB[1] pouze pro čtení: Příkaz WRITE byl poslán na zařízení, určené pouze pro čtení.

136 Konec souboru: Počítač dosáhl konce souboru.

137 Záznam uříznut: Typický případ

výskytu této chyby je při čtení záznamu, který je delší než maximální určená délka. (V BASICu je maximální délka záznamu 119 byte.)

136 Zařízení blokováno: Zařízení neodpovídá.

139 Záporný potvrzovací signál zařízení: Chyba je buď v sériovém kanálu nebo v periferním zařízení.

140 Chyba při čtení vstupních dat: Ke ztrátě informace došlo na cestě z periferního zařízení do počítače.

141 Kurzor mimo dosah: Kurzor se dostal mimo oblast, vymezenou jednotlivými módy.

142 Překročen rámec dat na sériové sběrnici: Ke ztrátě informace došlo na cestě z periferního zařízení do počítače.

143 Chybný kontrolní součet datového rámce na sériové sběrnici: Ke ztrátě informace došlo na cestě z periferního zařízení do počítače.

144 Chyba v zařízení: Uživatel se pokouší zapisovat na disketu, chráněnou proti zápisu.

145 Nesprávný grafický mód: Uživatel se pokouší otevřít obrazový editor s neplatným číslem grafického módu.

146 Funkce nerealizována: Funkce nebyla v manipulačním programu realizována.

147 Nepostačující paměť: Paměť RAM

nepostačuje pro operace ve zvoleném grafickém módu.

160 Chybné číslo jednotky: Uživatel určil chybné číslo jednotky.

161 Příliš mnoho otevřených souborů: Vyrovnávací paměť je vytížena.

162 Plná disketa: Na disketě již nejsou žádné volné sektory.

163 Chyba systému vstup/výstup: Na disketě mohou být znehodnoceny soubory DOS.

164 Nesouhlasí číslo souboru: Může být znehodnocen soubor na disketě.

165 Chybné jméno souboru: Specifikace souboru obsahuje chybné znaky.

166 Chybná délka dat v POINT: Druhý parametr příkazu POINT je příliš dlouhý.

167 Chráněný soubor: Uživatel se pokouší použít chráněný soubor jinak než pro čtení.

168 Neplatný příkaz: Chybný příkaz ve zvláštním operačním kódu.

169 Plný adresář: Již je použit veškerý prostor, přidělený pro adresář (64 jmen souborů).

170 Soubor nenalezen: Uživatel se pokouší pracovat se souborem, jehož jméno není obsaženo v adresáři diskety.

171 Neplatná instrukce POINT: Uživatel se pokouší ukázat na byte v souboru,

který nebyl otevřen pro aktualizaci.

172 Neplatné připojení k souboru: Uživatel se pokouší pomocí DOS II otevřít pro doplnění soubor v DOS I. Překopírujte soubor DOS I pomocí DOS II na disketu DOS II.

173 Vadné sektory: Disketová jednotka našla při formátování diskety vadné sektory. Použijte jinou disketu, protože disketa s vadnými sektory nemůže být naformátována. Vyskytne-li se tato chyba u více disket, může se jednat o závadu disketové jednotky.

1) IOCB znamená Inpu/Output Control Block - Vstupní /výstupní řídící blok. Číslo zařízení je totéž jako číslo IOCB.

Některé důležité adresy paměti RAM

Obsah paměťové buňky leží v rozsahu 0 až 255 a může být čten pomocí instrukce PEEK. Pomocí instrukce POKE lze do paměťové buňky uložit novou hodnotu. Větší číselné hodnoty než 255 se ukládají do dvou paměťových buněk se sousedními adresami. Obsah těchto dvou buněk se pak stanoví součtem obsahu nižší buňky a 256 násobku obsahu vyšší buňky. Chceme-li např. určit sloupec, ve kterém je kurzor, vypočítáme PEEK (85) + 256 * PEEK (86). Chceme-li polovinu kurzoru změnit do sloupce SLOUP, provedeme instrukci POKE 85, SLOUP - INT (SLOUP / 256) * 256 a POKE 86, INT (SLOUP / 256). V následujícím přehledu paměťových buněk a jejich funkcí jsou jejich adresy uvedeny v desítkové reprezentaci.

Adresa	Funkce
ROZSAHY PAMĚTI	
14, 15	konec uživatelské paměti RAM pro program a proměnné
18,19,20	interní časovač
88, 89	počátek videopaměti (levý horní obrazový element)
128, 129	konec paměti pro BASIC
144, 145	začátek paměti pro BASIC
741, 742	nejvyšší volná buňka RAM
734, 744	nejnižší volná buňka RAM
OBRAZOVÝ EDITOR	
82	levý okraj textu (0-39, standardně 2)
83	pravý okraj textu (0-39, standardně 39)
84	řádková souřadnice textového kurzoru
85, 86	sloupcová souřadnice textového kurzoru
90	řádková souřadnice grafického kurzoru - výchozí bod pro DRAWTO nebo XIO 18
91, 92	sloupcová souřadnice grafického kurzoru
93	kód znaku, na kterém je kurzor
94, 95	adresa paměti textového kurzoru
96	řádková souřadnice konce operace DRAWTO nebo XIO 18

87, 88	sloupcová souřadnice konce operace DRAWTO nebo XIO 18
201	počet sloupců, o které bude posunut kurzor při užití TAB
622	listing 0-normální 255-pomalý
656	řádková souřadnice textového kurzoru v textovém okénku
657, 658	sloupcová souřadnice textového kurzoru v text. okénku
694	po stlačení libovolné klávesy klávesnice se vytvoří kód ATASCII 0-normálně zobrazený znak jiná hodnota - inverzně zobrazený znak
752	0-viditelný kurzor 1-neviditelný kurzor
755	0-neprůhledný kurzor 2-průhledný kurzor (standardní hodnota)
756	volba znakového souboru 224-velká písmena a číslice 226-malá písmena a graf.znaky 204-mezinárodní abeceda
763	ATASCII kód posledně psaného nebo čteného znaku
765	barvový registr pro operaci XIO 18
708-712	barvové registry 0-4

BUFFER MAGNETOFONU

61	ukazatel volné pozice (0 - plný buffer)
63	0 - potvrzený konec souboru
649	0 - čtení 128 - zápis
650	obsazení bufferu v bytech (max. 128)
1021 - 1151	buffer
54018	52 - zapnutý magnetofon 60 - vypnutý magnetofon

BUFFER TISKÁRNY

29	ukazatel aktuální pozice v bufferu
30	obsazení bufferu v bytech (max. 40)
960 - 999	buffer

KLÁVESNICE

16	0 - klávesnice zapnutá

	255 - klávesnice vypnutá
17	0 - stlačena klávesa BREAK
621	0 - normální stav 255 - povolení vypnutí klávesnice
702	64 - velká písmena 0 - režim CAPS 128 - režim CONTROL
731	0 - akustický signál klávesnice zapnut 1 - akustický signál klávesnice vypnut
732	0 - HELP nestlačeno 17 - HELP stlačeno
764	kód posledně stlačené klávesy 255 - žádná klávesa nestlačena
53279	stlačeny klávesy 0 - OPTION, SELECT, START 1 - OPTION, SELECT 2 - OPTION, START 3 - OPTION 4 - SELECT, START 5 - SELECT 6 - START 7 - žádná

RŮZNÉ

65	0 - vypnutý zvuk po dobu přenosu dat jiná hodnota - zapnutý zvuk
186, 187	číslo řádku v programu BASIC, ve kterém došlo k přerušení stlačením BREAK, instrukcí STOP nebo v důsledku chyby
195	kód chyby
212, 213	návratová adresa do BASICu při použití USR
251	výpočet prováděn 0 - v radiánech 1 - ve stupních

Poznámky:

1. ATASCII znamená ATARI ASCII. Písmena a číslice mají stejné hodnoty jako v ASCII, avšak některé speciální znaky jsou jiné.
2. Znaky od 128 do 255 jsou inverzně zobrazené znaky od 1 do 127.
3. Dekadický kód velkého písmene zvětšený o 32 dá dekadický kód téhož písmene malého.
4. Na příkaz PRINT ASC ("__") vytiskne počítač ATASCII kód znaku v uvozovkách.
5. Bílé symboly na černém pozadí představují inverzní zobrazení a černé symboly na bílém pozadí představují normální zobrazení.

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
0	0		Control ,	á
1	1		Control A	ú
2	2		Control B	Ñ
3	3		Control C	É
4	4		Control D	Ç
5	5		Control E	ä
6	6		Control F	ö
7	7		Control G	ï
8	8		Control H	£
9	9		Control I	?
10	A		Control J	ü
11	B		Control K	ß
12	C		Control L	Ö
13	D		Control M	ó
14	E		Control N	ò
15	F		Control O	ô
16	10		Control P	Ü
17	11		Control Q	ñ

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
18	12		Control R	ü
19	13		Control S	ï
20	14		Control T	é
21	15		Control U	ë
22	16		Control V	ö
23	17		Control W	ê
24	18		Control X	ä
25	19		Control Y	â
26	1A		Control Z	Ä
27	1B		Esc Esc	
28	1C		Esc Control -	
29	1D		Esc Control =	
30	1E		Esc Control +	
31	1F		Esc Control *	
32	20		Space bar	
33	21		Shift 1	
34	22		Shift 2	
35	23		Shift 3	
36	24		Shift 4	
37	25		Shift 5	
38	26		Shift 6	
39	27		Shift 7	
40	28		Shift 8	
41	29		Shift 9	
42	2A		*	
43	2B		+	
44	2C		,	
45	2D		-	
46	2E		.	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
47	2F		/	
48	30		0	
49	31		1	
50	32		2	
51	33		3	
52	34		4	
53	35		5	
54	36		6	
55	37		7	
56	38		8	
57	39		9	
58	3A		Shift :	
59	3B		;	
60	3C		<	
61	3D		=	
62	3E		>	
63	3F		Shift /	
64	40		Shift 8	
65	41		A	
66	42		B	
67	43		C	
68	44		D	
69	45		E	
70	46		F	
71	47		G	
72	48		H	
73	49		I	
74	4A		J	
75	4B		K	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
76	4C	L	L	
77	4D	M	M	
78	4E	N	N	
79	4F	O	O	
80	50	P	P	
81	51	Q	Q	
82	52	R	R	
83	53	S	S	
84	54	T	T	
85	55	U	U	
86	56	V	V	
87	57	W	W	
88	58	X	X	
89	59	Y	Y	
90	5A	Z	Z	
91	5B	[	Shift ,	
92	5C	\	Shift ↑	
93	5D	]	Shift .	
94	5E	^	Shift *	
95	5F	_	Shift —	
96	60	♦	Control ,	
97	61	a	a	
98	62	b	b	
99	63	c	c	
100	64	d	d	
101	65	e	e	
102	66	f	f	
103	67	g	g	
104	68	h	h	
105	69	i	i	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
106	6A		j	
107	6B		k	
108	6C		l	
109	6D		m	
110	6E		n	
111	6F		o	
112	70		p	
113	71		q	
114	72		r	
115	73		s	
116	74		t	
117	75		u	
118	76		v	
119	77		w	
120	78		x	
121	79		y	
122	7A		z	
123	7B		Control ;	
124	7C		Shift =	
125	7D		Esc Control < or Esc Shift <	
126	7E		Esc Delete Bk Sp	
127	7F		Esc Tab	
128	80		Control ,	
129	81		Control A	
130	82		Control B	
131	83		Control C	
132	84		Control D	
133	85		Control E	
134	86		Control F	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystroke	European Character
135	87		Control G	
136	88		Control H	
137	89		Control I	
138	8A		Control J	
139	8B		Control K	
140	8C		Control L	
141	8D		Control M	
142	8E		Control N	
143	8F		Control O	
144	90		Control P	
145	91		Control Q	
146	92		Control R	
147	93		Control S	
148	94		Control T	
149	95		Control U	
150	96		Control V	
151	97		Control W	
152	98		Control X	
153	99		Control Y	
154	9A		Control Z	
155	9B	Esc	Return	
156	9C		Esc Shift Delete (BS)	
157	9D		Esc Shift >	
158	9E		Esc Control Tab	
159	9F		Esc Shift Tab	
160	A0		Space bar	
161	A1		Shift 1	
162	A2		Shift 2	
163	A3		Shift 3	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
164	A4	$	Shift 4	
165	A5	%	Shift 5	
166	A6	&	Shift 6	
167	A7	'	Shift 7	
168	A8	(	Shift 9	
169	A9	)	Shift 0	
170	AA	*	*	
171	AB	+	+	
172	AC	,	,	
173	AD	-	-	
174	AE	.	.	
175	AF	/	/	
176	B0	0	0	
177	B1	1	1	
178	B2	2	2	
179	B3	3	3	
180	B4	4	4	
181	B5	5	5	
182	B6	6	6	
183	B7	7	7	
184	B8	8	8	
185	B9	9	9	
186	BA	:	Shift	
187	BB	;	;	
188	BC	<	<	
189	BD	=	=	
190	BE	>	>	
191	BF	?	Shift /	
192	C0	@	Shift 8	
193	C1	A	A	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
194	C2		CTRL B	
195	C3		CTRL C	
196	C4		CTRL D	
197	C5		CTRL E	
198	C6		CTRL F	
199	C7		CTRL G	
200	C8		CTRL H	
201	C9		CTRL I	
202	CA		CTRL J	
203	CB		CTRL K	
204	CC		CTRL L	
205	CD		CTRL M	
206	CE		CTRL N	
207	CF		CTRL O	
208	D0		CTRL P	
209	D1		CTRL Q	
210	D2		CTRL R	
211	D3		CTRL S	
212	D4		CTRL T	
213	D5		CTRL U	
214	D6		CTRL V	
215	D7		CTRL W	
216	D8		CTRL X	
217	D9		CTRL Y	
218	DA		CTRL Z	
219	DB		CTRL ;	
220	DC		CTRL =	
221	DD		CTRL ,	
222	DE		CTRL *	
223	DF		CTRL -	

Decimal Code	Hexadecimal Code	ATASCII Character	Keystrokes	European Character
224	E0		Control	
225	E1		a	
226	E2		b	
227	E3		c	
228	E4		d	
229	E5		e	
230	E6		f	
231	E7		g	
232	E8		h	
233	E9		i	
234	EA		j	
235	EB		k	
236	EC		l	
237	ED		m	
238	EE		n	
239	EF		o	
240	F0		p	
241	F1		q	
242	F2		r	
243	F3		s	
244	F4		t	
245	F5		u	
246	F6		v	
247	F7		w	
248	F8		x	
249	F9		y	
250	FA		z	
251	FB		Control ;	
252	FC		Shift =	
253	FD		Esc Control 2	
254	FE		Esc Control Delete Bk Sp	
255	FF		Esc Control	

ABECEDNÍ PŘEHLED SLOV, VYHRAZENÝCH PRO ATARI BASIC

Poznámka: Po všech zkrácených slovech je povinná tečka.

Rezervované slovo	Zkratka	Stručný význam instrukce v BASICu
ABS		Absolutní hodnota proměnné nebo výrazu.
ADR		Adresa řetězcové proměnné.
AND		Logický operátor pro logický součin. Výraz je pravdivý, jsou-li výrazy na obou stranách operátoru pravdivé.
ASC		Číselná hodnota - desítkové číslo v kódu ATASCII - prvního znaku řetězcové proměnné.
ATN		Arcustangens čísla nebo výrazu v radiánech nebo stupních.
BYE	B.	Návrat z BASICu do rezidentního operačního systému.
CLOAD	CLOA.	Zavádí data z magnetofonu do počítače.
CHR$		Znak, příslušející v ATASCII kódu určenému číslu v rozmezí 0 - 255.
CLOG		Dekadický logaritmus výrazu.
CLOSE	CL.	Ve spojení s vstupní / výstupní operací uzavírá soubor. Pracuje jako vstupní / výstupní příkaz.
CLR		Vykonává opačnou funkci než DIM : ruší dimenzování všech polí a nuluje všechny proměnné.
COLOR	C.	Volí barvový registr pro použití v instrukci PLOT.
COM		Vykonává tutéž funkci DIM.
CONT.	CON.	Znamená "pokračuj!". Znovu spouští program, jehož průběh byl přerušen klávesou BREAK, nebo který byl zastaven příkazem STOP nebo END. Program je spuštěn od následujícího řádku programu.
COS		Cosinus proměnné nebo výrazu ve stupních nebo radiánech.
CSAVE		Ukládá data z RAM na magnetofonovou kazetu.
DATA	D.	Vytváří seznam čísel a (nebo) písmen, která se používají v instrukci READ. Položky seznamu musí

		být oddělený čárkou.
DEG	DE.	Říká počítači, že argumenty následujících trigonometrických funkcí budou místo v radiánech ve stupních (standardní výpočet je v radiánech).
DIM	DI.	Vyhrazuje v paměti určený prostor pro matice, pole a řetězcové proměnné. (Veškeré řetězcové proměnné, pole a matice musí být dimenzovány.)
DOS	DO.	Znamená "Diskový Operační Systém". Zobrazuje menu DOS. (Viz návod DOS.)
DRAWTO	DR.	Kreslí úsečku mezi poslední polohou kurzoru a určeným bodem. END Zastavuje běh programu, uzavírá všechny otevřené soubory, vypíná zvuk. V programu může být použito víckrát. (Program může být znovu spuštěn instrukcí CONT.)
ENTER	E.	Ukládá data nebo program ve zdrojové formě. Funguje jako vstupní/výstupní příkaz.
EXP		e (2.7182818) povýšené na určené číslo.
FOR	F.	Společné s instrukcí NEXT vytváří cyklus FORNEXT. Nastavuje počáteční a konečnou hodnotu proměnné, která řídí počet průchodů cyklem.
FRE		Počet volných byte v uživatelské paměti RAM.
GET	GE.	Používá se většinou při práci s disketou k výběru jednotlivých byte.
GOSUB	GOS.	Odskok do podprogramu, začínajícího určeným číslem řádku.
GOTO	G.	Nepodmíněný odskok na řádek určeného čísla.
GRAPHICS	GR.	Určuje jeden ze 16 grafických módů. (Pro vymazání obrazovky může být použito GR.0.)
IF		Podmíněný odskok nebo podmíněné vykonání operace na témže řádku (pouze když je první výraz pravdivý).
INPUT	I.	Požaduje vložení vstupních dat z klávesnice. Vykonání instrukce pokračuje po vložení vstupních dat a stlačení klávesy /RETURN/. Pracuje také jako vstupní/ výstupní příkaz.
INT		Nejvyšší celé číslo, které je menší nebo rovno určené hodnotě. (Zaokrouhlení je vždy dolů i při záporné hodnotě čísla.)
LEN		Délka určené řetězcové proměnné v bytech nebo znacích. (Jeden byte obsahuje jeden znak.)

LET	LE.	Přiřazuje hodnotu číselným a řetězcovým proměnným. (V ATARI BASICu může být vynecháno.)
LIST	L.	Zobrazuje na obrazovce nebo posílá na výstupní zařízení výpis programu.
LOAD	LO.	Zavádí program z diskety do počítače.
LOCATE	LOC.	Do určené proměnné ukládá data, příslušející určenému bodu na obrazovce.
LOG		Přirozený logaritmus čísla.
LPRINT	LP.	Příkazuje řádkové tiskárně vytisknout určený text.
NEW		Maže veškerý obsah uživatelské paměti.
NEXT	N.	Příkaz k ukončení nebo pokračování cyklu FOR-NEXT v závislosti na hodnoté parametru, jehož počáteční a koncovou hodnotu nastavuje instrukce FOR.
NOT		Není-li výraz pravdivý, dává hodnotu 1; je-li výraz pravdivý, dává hodnotu 0.
NOTE	NO.	Užívá se pouze při práci s disketou. (Viz příručku DOS.)
ON		Používá se ve spojitosti s GOTO a GOSUB k provedení podmíněných odskoků. (Jsou možné i vícenásobné odskoky v závislosti na hodnotě proměnné nebo výrazu za ON.)
OPEN	O.	Otevírá určený soubor pro vstupní nebo výstupní operace.
OR		Operátor pro logický součet. Výraz je roven jedné, je-li pravdivý jeden nebo oba výrazy po obou stranách operátoru. Jsou-li oba výrazy nepravdivé, je výsledkem nula.
PADDLE		Udává polohu ovladače PADDLE.
PEEK		Dekadická hodnota obsahu určeného paměťového místa. (RAM nebo ROM.)
PLOT	PL.	Ukládá do místa, určeného souřadnicemi X, Y obrazový element.
POINT	P.	Používá se pouze při práci s disketou. Viz příručku DOS.
POKE	POK.	Vkládá do určeného paměťového místa určený byte. Může být použit pouze pro paměť RAM.
POP		Ruší návratovou informaci, příslušející poslední vykonané instrukci FOR nebo GOSUB. Používá se při

		odskoku z cyklu nebo podprogramu.
POSITION	POS.	Umisťuje kurzor do určeného místa na obrazovce.
PRINT nebo ?	PR.	Posílá data z počítače na specifikované výstupní zařízení, pracuje jako vstupní/ výstupní příkaz.
PTRIG		Udává stav spouštěcího tlačítka na ovladači her.
PUT	PU.	Posílá jeden byte z počítače do určeného zařízení.
RAD		Říká počítači, aby dával údaje u trigonometrických funkcí v radiánech. (Výpočty se prováději standardně v radiánech. Viz DEG.)
RED	REA.	Čte položku ze seznamu v instrukci DATA a přiřazuje ji určené proměnné.
REM	R.	Znamená "poznámky". Nevykoná nic, ale umožňuje vytvoření komentářů, které jsou vytištěny ve výpisu programu.
RESTORE	RES.	Dovoluje vícenásobné čtení dat.
RETURN	RET.	Ukončuje podprogram a vrací program na instrukci, bezprostředně následující instrukci GOSUB, která podprogram vyvolala.
RND		Generuje náhodné číslo mezi 0 a 1, ale menší než 1.
RUN	RU.	Spouští vykonávání programu; nuluje proměnné a ruší dimenzování polí a řetězců.
SAVE	S.	Ukládá na určené jméno na disketu data a programy. Pracuje jako vstupní/výstupní příkaz.
SETCOLOR	SE.	Naplňuje jednotlivé barvové registry daty odstínů a jasů.
SGN		Dává hodnotu 1, je-li hodnota proměnné kladné číslo, nulu, je-li nula a -1, je-li záporné číslo.
SIN		Sinus hodnoty udané ve stupních nebo radiánech.
SOUND	SO.	Volí zvukový registr, výšku tónu, zkreslení a hlasitost.
SQR		Odmocnina z určeného čísla.
STATUS	ST.	Čte stav určeného zařízení.STEP Užívá se společně s FOR-NEXT. Určuje hodnotu, která má být přeskočena mezi dvěma po sobě jdoucími průchody cyklem.
STICK		Udává polohu ovladače typu joystick.
STRIG		Při stlačení spouštěcího tlačítka na joysticku

		dává nulu, jinak jedničku.
STOP	STO.	Zastavuje program, ale neuzavírá soubory a nevypíná zvuk. STR$ Převádí číslo na řetězcovou proměnnou (například STR$ (65) je řetězec "65".)
THEN		Používá se s IF. Je-li výraz pravdivý, je instrukce vykonána. Není-li výraz pravdivý, přechází se na následující řádek.
TO		Používá se společně s FOR, například "FOR X=1 TO 10". Určuje krajní hodnotu, do které bude v průběhu cyklu počítáno.
TRAP	T.	Vyskytne-li se v průběhu programu chyba - error - přechází řízení na určený řádek programu.
USR		Volá programy ve strojovém kódu.
VAL		Převádí řetězec na číslo.
XIO	X.	Používá se při operacích s disketou (viz příručku DOS) a při práci s grafikou. Působí jako univerzální instrukce vstupu/ výstupu.

FLOP

Pro všechny uživatele ATARI XL/XE zajišťujeme tyto služby:

- vydávání disketového zpravodaje FLOP
- kazetová verze zpravodaje FLOP
- tisknutý zpravodaj OBČASNÍK
- informační BULLETIN FLOPu
- úpravy počítačů, datasetů, doplňky
- poštovní zásilková služba publikací a doplňků
- ediční a vydavatelská činnost pro kluby i jednotlivce
- konzultace, poradenská činnost

Ze starších publikací a zpravodajů pořídíme na požádání kopie.

FLOP
1.máje 1009
756 61 Rožnov p.R.

FLOP

ATARI BASIC

Uživatelská příručka

Vydal :	FLOP ROŽNOV 1991
Odpovědný redaktor :	Lýdie Kolaříková
Rozsah :	120 stran
Tisk :	FLOP ROŽNOV 1991
Náklad :	800 ks

MANUÁLY PROGRAMŮ ATARI XL/XE