# ATARI®

# SPRAVODAJ 4/89

# OBSAH

## PRE ZAČIATOČNÍKOV



## SYSTÉMOVÉ A UŽÍVATEĽSKÉ PROGRAMY



## TIPY A TRIKY

# PRE ZACIATOCNIKOV

**********
*KOPIROVANIE PROGRAMOV NA DISKETACH - 2.časť*
**********

V treťom čísle nášho Spravodaja sme uverejnili prvú časť popisu disketových kopírovacích programov. Dnes uverejňujeme druhú časť.

Popíšeme si dva kopírovacie programy - SUPER-COPY (na disketách sa rozširuje pod názvom SUPCOPY.COM alebo UNICOPY.COM), a TRACK COPIER (uložený pod názvom TRACKCPY.COM). TRACK COPIER je nový kopírovací program, ktorý priniesol jeden z našich kolegov z Poľska spolu s novou zrýchľovacou úpravou disketovej jednotky.

SUPERCOPY aj TRACK COPIER sa do počítača nahrávajú povelom LOAD cez DOS.

**********
* SUPERCOPY *
**********

Po nahratí SUPERCOPY do počítača sa vypíšu oznamy:

```
Copy programm
(1)  LOAD
(2)  SAVE
(3)  DOS
(4)  Directory
(5)  Formatieren
```

Stlačením čísla od 1 do 5 si zvolíte jemu zodpovedajúcu funkciu.

Popis jednotlivých funkcií:

a) **Funkcia (1) - LOAD:**

Táto funkcia umožňuje načítať do pamäte počítača program. Po jej vyvolaní sa zobrazí nasledujúce menu:

```
    LOAD Programm
(1) Von Boottape
(2) Von bootdisk
(3) Von EXE-File
(4) Von FCOPY-File
```

Stlačením klávesy 1 budete nahrávať program z kazety. Stlačením klávesy 2 nahráte BOOT-program diskety. Stlačením 3 alebo 4 nahráte jednotlivý súbor do pamäte.

(1) - Load von Boottape. Táto funkcia umožňuje prevádzať tzv. BOOT-programy z kazety na diskety do tvaru DOSového súbora. Po jej zvolení sa vypíše výzva:

PLAY und RETURN druecken.

a po stlačení PLAY na magnetofóne a RETURN na klávesnici sa začne samotné čítanie z kazety. Po načítaní programu do pamäte sa zobrazí hlavné menu.

(2) - Load von Bootdisk. Táto voľba umožňuje načítať BOOT-program z disku, vloženého do disketovej jednotky číslo 1. BOOT-program je taký program, ktorý sa nahráva z disku hneď po prevedení studeného štartu. Po nahratí BOOT-programu z disku sa zobrazí hlavné menu.

(3) - Load von EXE-File. Táto voľba načíta súbor, ktorý je nahrávateľný aj cez DOS. Po jej zvolení sa zobrazí otázka:

Filename fuer INPUT ? D: - zadaj vstupné zariadenie a meno. Predvolené zariadenie je D: a kurzor je nastavený na písmeno D v názve zariadenia. Názov sa zadáva klasicky pomocou klávesnice. Stlačením klávesy RETURN ukončíte zadávanie názvu a zadaný súbor sa načíta do pamäte počítača. Po načítaní sa zobrazí hlavné menu.

(4) - Load von FCOPY-File. Táto voľba umožňuje načítať súbor, ktorý sa v DOSe po načítaní spustí. Ostatný postup (zobrazenie výzvy a zadanie mena) je rovnaký, ako pri podfunkcii 3. Medzi funkciou Load von EXE-File a Load von FCOPY-File nie je prakticky žiadny rozdiel.

b) Funkcia <2> SAVE:

Táto funkcia umožňuje zaznamenať program v pamäti počítača na zvolené médium. Jej činnosť je podobná ako činnosť funkcie LOAD. Po jej zvolení sa zobrazí podmenu:

SAVE Program
(1) Als Boottape
(2) Als Bootdisk
(3) Als EXE-File
(4) Als FCOPY-File

Stlačením príslušnej klávesy si zvolíte jednu z poskytovaných podfunkcií.

(1) Save als Boottape. Táto funkcia zaznamená program vo forme BOOT-programu na kazetu. Po skončení záznamu sa zobrazí hlavné menu. Ešte pred záznamom sa musí program trochu upraviť (ide o zmeny v hlavičke). Preto po zvolení tejto funkcie sa chvíľočku (necelú pol sekundu) nič nedeje, a až potom sa vypíše výzva PLAY RECORD und RETURN druecken.

(2) Save als Bootdisk. Táto funkcia umožňuje zaznamenať načítaný program z pamäte na disketu, vloženú do disketovej jednotky číslo 1, vo forme BOOT-programu. Pri zápise budú prepísané aj BOOT-sektory (sektory 1,2, a 3). Budú prepísané tak, aby ukazovali na začiatok sektoru 5, na ktorom začne SUPERCOPY ukladať načítaný program. Po zvolení funkcie sa vypíše výzva Diskette fuer OUTPUT Bootdisk einlesen und RETURN druecken - Vlož disketu, na ktorú chceš zapísať program, a stlač klávesu RETURN. Stlačením inej klávesy než RETURN sa vrátite do hlavného menu.

(3) Save als EXE-File. Táto funkcia zapíše načítaný program formou súboru, čitateľného aj z DOSu. Takýto EXE-súbor sa po načítaní nespustí, a treba to previesť manuálne zadaním M a adresy.

(4) Save als FCOPY-File. Táto funkcia zaznamená program vo forme automaticky štartovaného súboru, čitateľného aj DOS-om. Po načítaní sa takýto súbor automaticky spustí.

c) Funkcia <3> DOS:

Spôsobuje návrat do DOSu. Nie je položená žiadna kontrolná otázka, prevedie sa iba návrat do DOSu.

d) **Funkcia (4) Directory:**

Táto funkcia spôsobuje vypísanie obsahu diskety, nachádzajúcej sa v disketovej jednotke číslo 1, na obrazovku. Po vypísaní obsahu je potrebné stlačiť niektorú klávesu pre návrat do hlavného menu.

e) **Funkcia (5) Formatieren:**

Táto funkcia naformátuje disketu v disketovej stanici jedna na strednú hustotu. Po jej zvolení sa vypíše výzva Diskette, die formatiere werden soll, einlesen und RETURN druecken - Disketu, ktorú chceš formátovať, vlož do jednotky 1, a stlač RETURN. Stlačením inej klávesy ako RETURN sa vrátite do menu.

**********
* TRACKCOPY *
**********

Ďalej budeme popisovať program TRACK COPIER. Do počítača ho nahráte pomocou SUPERDOSu 4.3T. Po jeho nahratí sa vypíšu údaje o programe a o jeho autorovi. Stlačením klávesy START sa dostaneme z úvodnej hlavičky do samotného módu kopírovania. V rámčeku sú zobrazené tieto údaje:

TURBO: - absent/present (neaktívny/aktívny Turbo systém - funguje iba BERHARDs TURBO 1050 a TOP-DRIVE),

RAM: xx kBYTE (veľkosť pamäte, použitej pre kopírovanie),

DENSITY: (hustota načítavanej diskety. Zobrazí sa až po prečítaní prvého sektora),

TRACK: xxx (číslo označuje stopu, ktorú práve číta kopírovací program).

SELECT: No formatting/Normal 1050 format/Turbi 1050 format (fomátovanie cieľovej diskety - žiadne, normálne a ak je aktívny TURBO, tak aj formátovanie TURBO-systémom)

START: Read source (čítanie zdrojového disku)

Pod týmito správami sa zobrazí výzva "insert source disk" - vložte zdrojovú disketu.

Po stlačení klávesy START sa začne samotné čítanie diskety. V dolnej časti obrazovky sa graficky zobrazuje počet načítaných stôp. Po načítaní jednej stopy sa zväčší číslo za TRACK, ktoré označuje číslo práve načítavanej stopy. Po zaplnení pamäte, alebo po dočítaní cieľovej diskety sa zobrazia výzvy:

SELECT: No formatting... (možnosť prepínania formátovania)

START: Write destination (zapísať načítaný obsah na cieľovú disketu)

Stlačením klávesy START sa začne zápis na disketu. Po skončení kopírovania sa zobrazí správa "copy completed" - kopírovanie ukončené.

Pri kopírovaní sa môže stať, že príde k chybe. Chybové hlásenie by sa nemalo kam zobraziť, preto sa číslo chyby zobrazuje do čísla stopy - za slovo TRACK. Ak je číslo za TRACK väčšie ako 128, znamená to číslo chyby. Pri chybe je čítanie alebo zápis prerušené, a je zobrazené pomocné menu:

SELECT: Retry (znovuopakovanie operácie, pri ktorej prišlo ku chybe)

START: Restart (spustenie kopírovacieho programu spolu s vymazaním pôvodného obsahu)

Stlačením START alebo SELECT si zvolíme príslušnú funkciu.

Stlačením klávesy RESET sa dostaneme do základného režimu. Avšak obsah kopírovacieho programu bude zničený.

# SYSTEMOVE A UZIVATELSKE PROGRAM

```
**************************
* APPEND START ADDRESS *
**************************
```

Drahomír Volný, Atari klub Bratislava

Program Append Start Address - ASA je určený na dodatočné úpravy štartovacej a inicializačnej adresy binárnych súborov. Po zavedení programu z DOSu sa na obrazovke objaví hlavné menu a výzva aby si užívateľ vybral čo bude robiť. Výber sa uskutočňuje stlačením prvého písmena názvu podprogramu.

Directory
Umožňuje zobrazenie adresára diskety na obrazovke. Po zavolaní podprogramu sa na obrazovke objaví výzva "Drive 1,2,3,4,8". Po stlačení jednej z týchto číslic dostaneme výpis adresára danej diskety. Ak stlačíme ľubovoľnú inú klávesu, obdržíme výzvu aby sme zadali spresnenie názvov súborov ktorých výpis chceme (ako pri voľbe A z DOSu 2.5).

Výpis adries
Umožňuje výpis počiatočnej a koncovej adresy programu (všetkých častí) a tiež štartovaciu a inicializačnú adresu. Po zavolaní tohoto podprogramu sa program spýta či má výpis smerovať aj na tlačiareň. Výber sa uskutočňuje stlačením Z-zapnúť a V-vypnúť. Ak si vyberieme výpis aj na tlačiarni, potom je potrebné zadať hlavičku (max. 40 znakov), ktorá sa vytlačí na papier pred celý výpis. Ak sme zadali hlavičku alebo chceme výpis iba na obrazovke, program nás vyzve aby sme zadali celé meno súboru, ktorého výpis chceme. Ak je výpis adries príliš dlhý, môžeme ho zastaviť stlačením Control-1 (to platí i pre tlačiareň).

Pridanie adresy
Umožňuje pridať štartovaciu alebo inicializačnú adresu programu. Najprv si vyberieme či chceme pridať štartovaciu alebo inicializačnú adresu. Potom zadáme celé meno súboru. Program prehľadá celý súbor a vypíše štartovaciu a inicializačnú adresu, ak nejakú nájde. Potom vypíše počiatočné adresy všetkých častí a vyzve užívateľa aby si jednu z nich vybral ako štartovaciu (inicializačnú). Výber sa uskutočňuje zadaním čísla pri adrese. U väčšiny programov treba zadať adresu poslednej časti t.j. najväčšie poradové číslo adresy. Takto sa dajú upraviť programy z DOSu XL do DOSu 2.5 (takýmto spôsobom som si upravil do DOS 2.5 aj program MAC65, BUG65, ktoré spolupracujú s DOS XL). Ak nechceme pridať ani jednu z ponúkaných adries, zadáme nulu. V takom prípade nás program vyzve, aby sme sami zadali hexadecimálnu adresu. Ak nechceme pridať žiadnu adresu, tak zadáme nulu. Ak sme nejakú adresu zadali, program ju pre kontrolu ešte vypíše a potom pridá k zvolenému súboru. Ak pridanie prebehlo bez chýb, objaví sa "ok".

Zmazanie adresy

Tento podprogram umožňuje zmazať všetky štartovacie adresy. Pracuje tak, že kopíruje jeden súbor na druhý dovtedy, kým nenájde štartovaciu alebo inicializačnú adresu. Takto sa dá chybne zadaná adresa znovu zmazať. Tu treba zadať staré a nové meno súboru, pričom staré meno reprezentuje pôvodný súbor a nové meno skopírovaný súbor bez štartovacích adries (nový súbor je výhodné kopírovať do Ramdisku). Ak všetko prebehne v poriadku objaví sa "ok".

Návrat do DOSu
Umožňuje návrat do DOSu. Pred návratom sa znovu spýta, či chceme prácu ukončiť.

Ak pri niektorej činnosti nastane chyba, program vypíše jej číslo a činnosť preruší. Program automaticky zisťuje či spracovávaný súbor je binárny alebo nie. Ak sa pokúsime spracovať iný súbor než binárny, program vypíše oznam "Nie je binárny súbor" a prácu preruší. Program ASA je napísaný v Jazyku DEEP BLUE C.

**********
* TEXTOVÉ EDITORY *
**********

Bohatý výber programov umožňuje používať počítače ATARI nielen v domácnosti, ale aj poloprofesionálnom nasadení. Ako sa ukazuje, ponuka programov je taká dobrá, že si môžeme vyberať. Často býva problémom, ktorý program si vybrať, vzhľadom na ich rozmanitosť a rôznorodosť úžitkových funkcií.

Jedným z najobľúbenejších využití počítačov je spracovanie textov. Na počítače Atari XE,XL je spracovaných niekoľko textových editorov. Tento článok vás zoznámi s niektorými z nich.

**AtariWriter+**

Je to vylepšená verzia prvého editoru textu na ATARI. Jeho výhodou je spolupráca so všetkými tlačiarňami (pre ktoré však musíte, samozrejme, mať postavený príslušný interface pre zapojenie). Dokáže (podľa druhu pripojenej tlačiarne) vytlačiť až 200 znakov na riadok. Pri vymazávaní a hľadaní časti textu možno použiť znak "?" ako jokera, ktorý zastupuje každý iný znak. Umožňuje aj prihrať k textu nejaké tabuľky, vytvorené pomocou programu SynFile+. Neumožňuje vytlačiť zväčšené písmená (pri tlači na tlačiarni), a jeho ďalšou nevýhodou je malý priestor pre ukladanie textu na počítačoch ATARI 800XL, 800XE a 65XE. Pri použití počítača ATARI 130XE využíva pre text priestor asi 47kB. Producentom programu je ATARI Corp.

**PaperClip**

PaperClip je na Západe uznávaný za najlepší editor pre osembitové počítače ATARI (existuje už aj vo verzii pre ATARI ST a MEGA ST). Má veľké redakčné možnosti a príkazy sú ľahko zapamätateľné. Medzi jeho ďalšie výhody patria aj zabudované matematické funkcie, automatické zapisovanie textu, použitie funkcií Macro a Undo, spolupráca so všetkými tlačiarňami, možnosť pridania grafiky k textu i použitie hociakého druhu písmen (hociakej znakovej sady). Dĺžka riadku na obrazovke je definovaná cez užívateľa. Medzi vady patrí dlhý čas vymieňania častí textu. Na systémovej diskete programu PaperClip sa nachádza aj program pre transformáciu programov zapísaných v editore AtariWriter+ do formátu PaperClipu. Producentom tohoto programu je Batteries Included.

## SpeedScript

Je to pomerne krátky a jednoduchý editor textu, napísaný Charlesom Brannonom. Pre prezeranie textu umožňuje vytlačiť tento text na obrazovku, ale iba formátom 40 znakov na riadok. Nemá zabudované kódy tlačiarní, čo umožňuje k nemu pripojiť tiež všetky druhy tlačiarní. Kódy pre tlač textu sa musia zadávať priamo do textu. Najväčšou výhodou je to, že má veľký bufer pre spracovanie a uloženie textu (asi 28kB) a že dokáže spolupracovať s magnetofónom.

## Word Magic

Je to program napísaný firmou Blue Collar Software podľa amerického mesačníka ANTIC. Môže pracovať s hociakou tlačiarňou a umožňuje tiež tlačiť grafiku. Preklad textu prebieha v grafickom móde (jeden bod = jeden znak), teda možno spraviť preklad jednej strany textu. Pre vytlačenie textu sa načíta špeciálny program, ktorý potom prečíta z diskety text, a tento text vytlačí. Presúvanie (pohyb) kurzora sa môže dosiahnuť pomocou klávesnice, alebo joysticku.

## Home Text

Je to program, zahrnutý v celodisketovom programe HomePak. Malý priestor bufera umožňuje napísať iba osem strán klasického strojopisu. Je veľmi nekomfortný a neprispôsobený v obsluhe, navyše aj napísaný text sa dá z obrazovky len prečítať. Producentom je Batteries Included.

## HomeWord

Je to program producenta Sierra On-Line. Veľmi malý bufer umožňuje zachovať necelé štyri strany textu. Pri zápise vykonáva kópie súborov. Má špeciálny formát zápisu a jeho text nemôže byť čítaný iným editorom.

## First Xlent Processor

Je to najnovší editor pre počítače ATARI. Svojimi poskytovanými možnosťami ustupuje len PaperClipovi. Neumožňuje automatický zápis, používať matematické funkcie, nepozná funkcie Macro a Undo. Umožňuje kopírovanie textu dlhého asi 800 znakov. Presúvanie blokov textu a vymieňanie textu je veľmi rýchle. Xlent môže pracovať so všetkými druhmi tlačiarní, a pri použití iných znakových sád je možné vytlačiť aj tieto sady.

## Stat Texter

Je to program nemeckého producenta SYBEX-Verlag GMBH. Umožňuje nadefinovanie hociakej znakovej sady. Umožňuje spoluprácu s hociakými tlačiarňami. Avšak možnosťami ustupuje programom PaperClip, First Xlent Processor, AtariWriter+. Navyše má aj dosť malý priestor pre ukladanie textu - asi 20kB. Jeho výhoda je práca na 80 znakovej obrazovke, hoci na riadok je zobrazovaná iba časť z tohto osemdesiat znakového módu.

## ČAPEK 2.1

Naprogramoval ho Ing.Petr Jandík z Prahy. Svojimi možnosťami už ďaleko predčí svojho predchodcu - Speedscript. Umožňuje písať texty dlhé 26.5kB, (v prípade použitia s magnetofónom ešte viac), na obrazovke zobrazuje 40 znakov, ale má zabudovanú funkciu zobrazenia 80 znakov na obrazovke, čo umožňuje si prakticky pozrieť formát, v akom bude vytlačený text na tlačiareň, čo je zvlášť výhodné pri editovaní textu.

Umožňuje vyhľadávanie a menenie textu, veľký buffer umožňuje kopírovanie textu o dĺžke asi 3.5 kilobytu. Umožňuje tiež tlač českých a slovenských znakov na tlačiarni ATARI 1029. Umožňuje nastaviť ľavý okraj, pravý okraj, veľkosť strany na tlačiarni (v riadkoch). Umožňuje tiež zarovnávanie pravého okraja. Môže vytlačiť aj nami definované znaky. Je to rozhodne najlepší československý editor textu pre osembitové počítače ATARI.

V súčasnosti je rozšírená verzia Čapka 3.0, ktorá má ďalšie

vylepšenia a doplnky. Medzi najzaujímavejšie patrí možnosť tlačenia grafiky a vylepšenú mierku pri móde 80 znakovom.

Ako príklad práce Čapka môžu poslúžiť aj Spravodaje ATARI klubu Bratislava, ktoré sú od čísla 1/1989 tlačené cez program Čapek.

V nasledujúcej tabuľke sú prehľadným spôsobom uvedené charakteristické znaky jednotlivých popisovaných textových editorov.

| EDITOR | AW+ | PC | SS | WM | HT | HW | FX | ST | CAP |
|---|---|---|---|---|---|---|---|---|---|
| médium | D | D | D/K | D | D | D | D | D | D/K |
| dĺžka textu XL | 12.3 | 25.0 | 27.9 | 23.7 | 6.5 | 3.9 | 28.3 | 19.5 | 26.5 |
| dĺžka textu XE | 45.5 | 30 | - | RD | - | - | RD | RD | 26.5 |
| funkcia HELP | N | A | N | A | N | A | A | N | N *1 |
| automatic. zápis | N | A | N | N | N | N | N | N | N |
| tlač časti textu | A | A | A *2 | N | N | N | A | N | A *3 |
| funkcia Undo | N | A | N | N | N | N | N | N | N |
| počet okien | 1 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| presuv pozície | A | A | N | N | N | N | N | A | A |
| nast. riadkov. | N | A | N | N | N | N | A | N | A |
| nast. šírky | A | A | A | A | A | A | A | A | A |
| 2-stĺpcová tlač | A | A | N | N | N | N | A | N | N |
| zab. kódy tlač. | A | A | N | A | N | N | N | A | A |
| tab. zo SynFile+ | A | A | N | A | A | N | A | N | N |
| možnosť prezer. | 40 | 40 | N | GR | N | GR | 80 | 80 | 80 |
| tlač grafiky | N | A | N | A | N | N | A | A | A |
| def. znak. sád | N | A | N | N | N | N | A | A | N *4 |

Pozn.:
*1 - Popis funkcií sa nachádza v manuáli, dodávanom s programom Čapek.
*2 - Len od začiatku vybranej strany do konca súboru
*3 - Len od pozície kurzora do konca textu, alebo do prerušenia tlače textu
*4 - V programoch Čapek 2.1... je definovaná sada s českými a slovenskými písmenami

Skratky jednotlivých editorov textu:

- AW+ - AtariWriter+
- PC - PaperClip
- SS - SpeedScript
- WM - WordMasic
- HT - Hometext
- HW - Homeword
- FX - First Xlent Procesor
- ST - Startexter
- CAP - Čapek

## ******** * OKNÁ * ********

Nové počítače firmy ATARI - ATARI ST, pracujúce pod systémom GEM, používajú pre komunikáciu s užívateľom metódu takzvaných okien - všetky texty, ktoré majú byť oznámené užívateľovi, sa vypisujú v oknách. Technika okien sa dá vytvoriť aj na osembitových ATARI počítačoch.

Osembitové počítače ATARI poskytujú možnosť práce s periférnymi zariadeniami - s magnetofónom, s disketou, tlačiarňou, obrazovkou a klávesnicou. Každé zariadenie má svoje označenie, napr. práca s magnetofónom je definovaná pod zariadením C:. Zoznam použiteľných zariadení sa nachádza v tzv. tabuľke HATABS - v tabuľke zariadení. Táto tabuľka poskytuje možnosť práce s trinástimi zariadeniami. Obsadených je iba 6 zariadení, čiže 7 ich môže užívateľ definovať. Ako príklad definovania môže poslúžiť modul RS-232C. Do tabuľky HATABS sa zariadenia ukladajú v takomto poradí : Názov zariadenia, dvojbajtová adresa, ukazujúca na začiatok obsluhy daného zariadenia. Zoznam zariadení sa v tabuľke HATABS číta zdola nahor, čiže najprv by sa prečítali užívateľom definované zariadenia, a až potom štandartné periférie. Obsluha daného zariadenia musí obsahovať tieto podrutiny : OPEN, CLOSE, PUT, GET a môže obsahovať STATUS, SPECIAL. Môže tiež obsahovať skok na nejakú ďalšiu podinicializačnú rutinu.

Príklad : príkaz OPEN #1,4,0,"C:" v Atari BASICu spôsobí vyhľadanie ATASCII znaku C (Jeho kód je 67) v tabuľke HATABS. Po nájdení čísla 67 v tabuľke HATABS sa prečítajú nasledujúce dva bajty a skočí na podprogram, začínajúci na adrese, vypočítanej z týchto dvoch bajtov. Na tejto adrese musí začínať podprogram pre obsluhu magnetofónu ako periférneho zariadenia.

Dostávame sa do hlavnej časti - k utvoreniu zariadenia pre prácu s oknami. Toto zariadenie je O: (Okna). V inštrukcii OPEN sa miesto parametrov, zaužívaných pri práci s magnetofónom, alebo s disketou, zadáva ľavý horný roh okna. Napríklad OPEN #1,28,5,"O:" otvorí kanál #1 pre prácu s okienkami, pričom ľavý horný roh novootvoreného okna bude na pozícii 28,5. Lenže program musí vedieť aj rozmery okna a keďže do inštrukcie OPEN možno zadať len dva parametre, rozmery okna sa zapíšu do adries 205 a 206 na nultej stránke. Do adresy 205 sa zapisuje šírka a do adresy 206 výška okna. Po otvorení niektorého kanála pre prácu s oknom je najskôr skúšané, či užívateľ pracuje v grafickom móde 0. Ak nie, je oznámená chyba 255. Pokiaľ by bolo okno malé, alebo by nastavené parametre nevyhovovali podmienkam pre otvorenie okna, je vypísaná chyba 200.

Po otvorení okna sa kurzor umiestni do jeho ľavého horného rohu. V okne nemožno kurzorom pohybovať pomocou inštrukcie POSITION, ale pomocou ATASCII kódov, pohybujúcich kurzorom (to znamená, že CHR$(28) posunie kurzor v okienku o jednu pozíciu hore, CHR$(29) o jednu pozíciu dole, CHR$(30) doľava a CHR$(31) o jednu pozíciu doprava). Príkazy PRINT a PUT budú pracovať iba s posledne otvoreným oknom - to znamená, že pô otvorení niektorého ďalšieho okna sa znedprístupnia všetky ostatné.

Program sa nevošiel do šiestej strany pamäti. Je umiestnený na 158-mej strane, a keby ho chcel užívateľ premiestniť na iné miesto, musí zmeniť všetky čísla 158 v riadkoch 100 a 101 a číslo 158 (prvú položku) v riadku 129.

Prvý z BASICovských programov, ktorý uvádzame na konci Spravodaja vytvára strojový program pre obsluhu okien, ktorý je uvedený za týmto textom. Pre úsporu miesta je výpis uvedený v dvoch stĺpcoch. Druhý basicovský program je demo-programom, ktorý slúži pre ukážku techniky okien.

Výpis strojového programu pre obsluhu okien

```
0100 RX = 203
0110 RY = 204
0120 DX = 205
```

```
0130 DY = 206
0140 W1 = 207
0150 W2 = 208
0160 KX = 4
0170 KY = 5
0180 PUTCH = 62128
0190 * = 40448
0200      .WORD OPE-1,SUK-1,SUK-1
0210      .WORD PUT-1,SUK-1,SUK-1
0220      JMP SUK
0230 OPE  LDA 842,X
0240      STA RX
0250      LDA 843,X
0260      STA RY
0270      LDA 87
0280      BEQ DL1
0290      LDY #255
0300      RTS
0310 DL1  SEC
0320      LDA DX
0330      CMP #3
0340      BCS DL2
0350      LDY #200
0360      RTS
0370 DL2  LDA DY
0380      CMP #3
0390      BCS DL3
0400      LDY #200
0410      RTS
0420 DL3  CLC
0430      LDA RX
0440      ADC DX
0450      STA DX
0460      SEC
0470      CMP #40
0480      BCC DL4
0490      LDY #200
0500      RTS
0510 DL4  LDA RY
0520      ADC DY
0530      STA DY
0540      SEC
0550      CMP #24
0560      BCC OTW
0570      LDY #200
0580      RTS
0590 OTW  LDA 88
0600      STA W1
0610      LDA 89
0620      STA W2
0630      LDX #0
0640 SK1  INX
0650      CLC
0660      LDA #40
0670      ADC W1
0680      STA W1
0690      LDA #0
0700      ADC W2
0710      STA W2
0720      CPX RY
0730      BNE SK1
0740 SK2  LDA #0
0750      LDY RX
0760 SK3  STA (W1),Y
0770      INY
0780      CPY DX
0790      BNE SK3
0800      CLC
0810      LDA #40
0820      ADC W1
0830      STA W1
0840      LDA #0
0850      ADC W2
0860      STA W2
0870      INX
0880      CPX DY
0890      BNE SK2
0900      DEC 6
0910      DEC DX
0920      LDA 88
0930      STA W1
0940      LDA 89
0950      STA W2
0960      LDX #0
0970 SK4  INX
0980      CLC
0990      LDA #40
1000      ADC W1
1010      STA W1
1020      LDA #0
1030      ADC W2
1040      STA W2
1050      CPX RX
1060      BNE SK4
1070      LDA #81
1080      LDY RX
1090      STA (W1),Y
1100      INY
1110 SK5  LDA #82
1120      STA (W1),Y
1130      INY
1140      CPY DX
1150      BNE SK5
1160      LDA #69
1170      STA (W1),Y
1180 SK6  CLC
1190      LDA #40
1200      ADC W1
1210      STA W1
1220      LDA #0
1230      ADC W2
1240      STA W2
1250      LDA #124
1260      LDY RX
1270      STA (W1),Y
1280      LDY RX
1290      STA (W1),Y
1300      INX
```

```
1310        CPX DY
1320        BNE SK6
1330        LDY RX
1340        LDA #90
1350        STA (W1),Y
1360        INY
1370        LDA #82
1380 SK7    STA (W1),Y
1390        INY
1400        CPY DX
1410        BNE SK7
1420        LDA #67
1430        STA (W1),Y
1440        LDA RX
1450        STA KX
1460        LDA RY
1470        STA KY
1480        INC KX
1490        INC KY
1500 SUK    LDY #1
1510        RTS
1520 ;
1530 PUT    TAX
1540        LDA RX
1550        STA 82
1560        LDA DX
1570        STA 83
1580        LDA 84
1590        PHA
1600        LDA 85
1610        PHA
1620        LDA KX
1630        STA 85
1640        LDA KY
1650        STA 84
1660        TXA
1670        CMP #125
1680        BNE NIE
1690        PLA
1700        STA 85
1710        PLA
1720        STA 84
1730        INC DX
1740        JMP OTW
1750 NIE    JSR PUTCH
1760        LDA 85
1770        STA KX
1780        LDA 84
1790        STA KY
1800        LDA KX
1810        CMP RX
1820        BNE PB1
1830        LDA DX
1840        STA KX
1850        DEC KX
1860        DEC KY
1870 PB1    CMP DX
1880        BNE PB2
1890        LDA RX
1900        STA KX
1910        INC KX
1920        INC KY
1930 PB2    LDA KY
1940        CMPRY
1950        BNE PB3
1960        LDA DY
1970        STA KY
1980        DEC KY
1990 PB3    CMP DY
2000        BNE PB4
2010        LDA RY
2020        STA KY
2030        INC KY
2040 PB4    PLA
2050        STA 85
2060        PLA
2070        STA 84
2080        LDA #2
2090        STA 82
2100        LDA #39
2110        STA 83
2120        LDA #30
2130        JSR PUTCH
2140        LDA #31
2150        JSR PUTCH
2160        LDY #1
2170        RTS
```

# DISKETA A DATOVÉ SÚBORY

Vladimír Tankovič, Peter Pipan, Atari klub Bratislava

## Použitie čiarky v príkaze PRINT #;

V súvislosti s príkazom PRINT #; budú, na oddelenie, často používané čiarky.

```
80 ? #4;"PROSIM","AKO SA VOLATE"
```

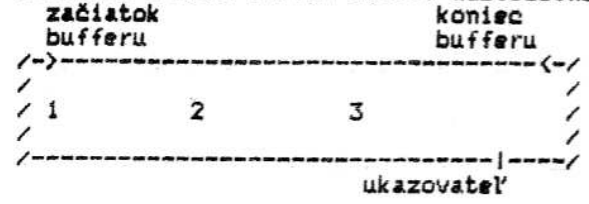
Ale môže byť práve tak dobre na konci príkazu PRINT #;:

```
3000 ? #2;"KONIEC",
```

Pre počítač nie je rozdiel, či bude zapisovať data na obrazovku alebo na disk. Na obrazovke bude pomocou čiarky posunutý kurzor na ďalšiu pozíciu tabulátora. Okrem toho bude zamedzený výstup znaku EOL. Pri diskovom súbore stojí ukazovateľ súboru na pozícii kurzoru.
Pozrime sa na nasledujúci príkaz bližšie.

```
PRINT #1;1,2,3,
```

Ak je kanál č. 1 otvorený, potom bude uložené jedno pole do otvoreného súboru. Keďže sú data oddelené čiarkou, bude nasledovať za každým číslom deväť prázdnych znakov. Príkaz je ukončený tiež čiarkou a preto nebude vydaný EOL znak. Po vykonaní vyzerá buffer súboru nasledovne:




Účinok čiarky v príkaze PRINT #;

Pre uloženie prázdnych znakov nebolo treba použiť čiarky. Keď chceme zadať data bez EOL znaku, môžeme použiť bodkočiarku.
Ak chceme zamedziť, aby boli data uložené bezprostredne za sebou, použijeme pre každé pole samostatný príkaz PRINT #; alebo zabezpečíme (viď príklad) samostatný vstup EOL znaku.

```
40 ? #5;"MEDZI TYMTO";CHR$ (155);
"A TYMTO STOJI EOL ZNAK"
```

CHR$ (155) zodpovedá EOL znaku. Ak chceme túto metódu použiť viackrát, je výhodnejšie ak EOL znak uložíme do

string-premennej.

```
10 DIM R$(1)
20 R$=CHR$(155):REM EOL
:
:
300 ? #2;MENO$;R$;POVOLANIE$;R$;
VEK$
```

Zapisovanie do datového súboru pomocou príkazu PUT.
**********

Pomocou príkazu PUT možno do diskového súboru zapísať jedno číslo s hodnotou medzi 0 a 255. Normálne toto číslo zodpovedá ATASCII-znaku. Každé číslo zaberá v súbore toľko miesta ako jeden znak uložený pomocou PRINT #;. Pri použití príkazu PUT nebude zapísaný EOL znak. Nasledujúci program zapíše, v uvodzovkách stojaci, text ("MENO") a EOL znak do uvedeného súboru.

```
10 OPEN #5,8,0,"D:MENO.TXT"
20 PUT #5,34:REM "
30 PUT #5,77:REM M
40 PUT #5,69:REM E
50 PUT #5,78:REM N
60 PUT #5,79:REM O
70 PUT #5,34:REM "
80 PUT #5,155:REM (EOL)
90 CLOSE #5
100 END
```

Čítanie zo sekvenčného datového súboru.
**********
Data uložené v diskovom súbore, možno aj vyvolať. Budú čítané pomocou INPUT #; a GET a priradené premennej. Pre názornú ukážku si napíšeme tento program a odštartujeme ho! Tým bude vytvorený súbor s menom DATSUBOR. BAS

```
30 OPEN #1,8,0,"D:DATSUBOR.BAS"
40 ? #1;"MOZNO ZAPISAT"
50 ? #1;"A ZNOVU VYVOLAT"
60 CLOSE #1
70 END
```

Pomocou nasledujúceho programu budú data uložené v súboru DATSUBOR.BAS prečítané a zobrazené na obrazovke.

```
10 DIM A$(100)
20 REM súbor otvoriť pre čítanie
30 OPEN #1,4,0,"D:DATSUBOR.BAS"
40 INPUT #1;A$
50 ? A$
60 GOTO 40
70 CLOSE #1
80 END
```

Výstup tohoto programu vyzerá nasledovne:

```
MOZNO ZAPISOVAT
```

```
A ZNOVA VYVOLAT

ERROR- 136 AT LINE 40
```

Chyba vznikla z pokusu prečítať viac dat ako obsahuje uvedený súbor. Proram sa na riadku 40 zastaví. Príkaz CLOSE na uzavretie súboru (riadok 70) už nemôže byť vykonaný. V tomto prípade sa nič zlé nestalo, lebo súbor bol prečítaný a v bufferi súboru neostali žiadne data, ktoré by boli zapísané na disku. Nie je to však elegantné čítanie, keď je program ukončený chybou. Túto chybu môžeme zachytiť príkazom TRAP. Vsunutím jednoho riadku.

```
35 TRAP 70
```

Pre istotu preskúšame, či sa skutočne vyskytla chyba v čítaní zo súboru (koniec súboru), lebo každá chyba uzavrie súbor. Pre toto preskúšanie upravíme program:

```
35 TRAP 62

61 REM VOLANIE CISLA CHYBY
62 CH=PEEK (195)
63 REM KONIEC SUBORU?
64 IF CH=136 THEN 70
65 REM AK BOLO VYDANE INE CISLO CHYBY
66 ? "V PROGRAME SA VYSKYTLA CHYBA";CH
```

Pomocou príkazu INPUT bude čítané pole datového súboru, ktorý bol zavolaný, a to všetky znaky, až po výskyt EOL znaku. Data uložené v tomto poli budú priradené príslúchajúcej premennej. Data a premenné musia byť samozrejme jedného druhu. Napríklad pri priradení stringu do číselnej hodnoty vznikne chyba.

Pomocou príkazu INPUT bude čítané pole datového súboru, ktorý bol zavolaný, a to všetky znaky, až po výskyt EOL znaku. Data uložené v tomto poli budú priradené príslúchajúcej premennej. Data a premenné musia byť samozrejme jedného druhu. Napríklad pri priradení stringu do číselnej hodnoty vznikne chyba.

## Čítanie zo súboru pomocou príkazu GET.

Pomocou príkazu GET bude čítaný jednotlivý znak datového súboru vo forme čísla, ktoré má hodnotu medzi 0 a 255. Ako budú tieto čísla interpretované určuje program. Vo väčšine prípadou budú tieto čísla, s použitím CHR$, zmenené na ATASCII-znaky. Nasledujúci program číta ten istý súbor ako v predchádzajúcom, ale s použitím funkcie CHR$. V programe je potrebné upraviť nasledovné riadky:

```
40 GET #1,A
50 ? CHR$(A)
```

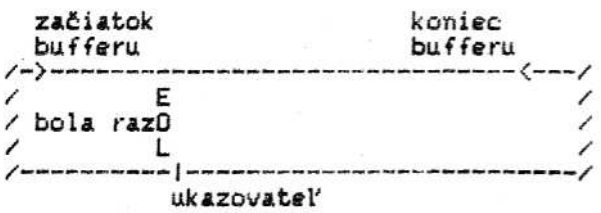
## Rozšírenie existujúceho súboru.

Ak chceme otvoriť existujúci datový súbor na disku tak, aby sa tento nevymazal, zadáme 9 ako druhé číslo po príkaze OPEN. Ukazovateľ súboru sa nastaví po tomto príkaze na koniec súboru. Ak však súbor neexistuje objaví sa (ERROR 170).

Predpokladajme napríklad, že sa na diskete nachádza súbor s menom PRIPOJIT.TXT. V tomto súbore sú uložené slová "BOLA RAZ" a EOL znak. Teraz zostavíme nasledovný program:

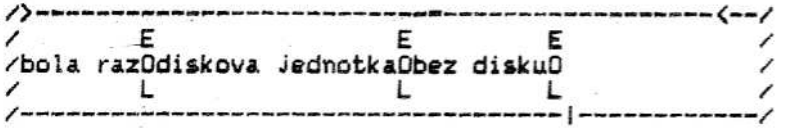
```
100 OPEN #2,9,0,"D:PRIPOJIT.TXT"
110 ? #2;"DISKOVA JEDNOTKA "
120 ? #2;"BEZ DISKU"
130 CLOSE #2
150 END
```

Príkaz OPEN na riadku 100 určuje, že súbor má byť rozšírený. Tým bude ukazovateľ súboru nastavený na koniec súboru.




Súbor otvorený na rozšírenie.

Data, ktoré budú dodatočne uložené do súboru, budú pripojené k predchádzajúcim




## Rozšírenie existujúceho súboru.

Vždy, keď bude zadaný príkaz OPEN pre rozšírenie súboru, bude pre tento súbor automaticky rezervovaný nový sektor (ukladacia kapacita 128 znakov), aj vtedy, ak v poslednom sektore, ktorý bol uložený do súboru, zostalo ešte miesto. Keď nie je program zostavený optimálne, môže byť na tomto základe obsadených veľa sektorov na disku.

Tento nedostatok pri nasledujúcej manipulácii so súborom nevzniká.

## Korekcia a zmena uložených dat v datovom súbore.

Ak zadáme za príkazom OPEN ako druhé číslo 8 alebo 9, môžme na disk len písať. Zadaním čísla 4 môžme len čítať. Ak chceme čítať i písať, musíme zadať číslo 12. V tomto prípade musí súbor, ktorý nasleduje po príkaze OPEN, už existovať.

Po otvorení súboru takýmto príkazom OPEN, nachádza sa ukazovateľ súboru na jeho začiatku. Teraz môžu byť data čítané aj zapisované. Je teda možnosť, už uložené data, prepísať a tým urobiť korektúry, alebo zmeny. S každým znakom, ktorý bude čítaný alebo zapísaný, posunie sa ukazovateľ súboru o jedno miesto ďalej v smere ku koncu súboru.

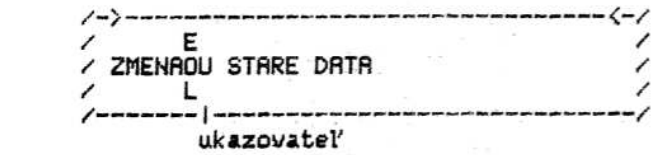
Keď budeme do súboru písať, prepíšu sa staré data novými. Na demonštráciu použijeme najprv program, ktorý uloží na disketu datový súbor s textom: "TOTO SU STARE DATA"

```
10 OPEN #4,8,0,"D:KOREKCIA"
20 ? #4;"TOTO SU STARE DATA"
30 CLOSE #4
40 END
```

Pomocou nasledujúceho programu bude uložený text zmenený.

```
100 OPEN #3,12,0,"D:KOREKCIA"
110 ? #3;"ZMENA"
120 CLOSE #3
130 END
```

Po vykonaní príkazu OPEN v riadku 100 sa ukazovateľ súboru nachádza na začiatku súboru. Text vydaný pomocou príkazu PRINT #; <v riadku 110> tam preto bude tiež uložený. Prvých šesť znakov starého súboru bude nahradených textom "zmena" a EOL-znakom. Teraz sa v zmenenom súbore nachádzajú dve polia oddelené EOL znakom.




### Prepísanie uložených dat.

Keď čítame v starom súbore, ukazovateľ sa pritom pohybuje smerom ku koncu súboru. Tak môžme v každom ľubovolnom bode súboru data zmeniť alebo korigovať. Čítame jednoducho tak dlho, pokiaľ sa ukazovateľ nenachádza na želanom mieste a potom napíšeme nové data do súboru. Čítať môžme pomocou príkazu INPUT #; ako aj GET. S INPUT #; bude čítané celé pole a s GET bude čítaný jednotlivý znak. V nasledujúcom príklade bude použitý príkaz GET.

```
10 TRAP 300:REM ak súbor neexistuje
20 REM otvoriťsúbor k zmene dat
```

```
30 OPEN #3,12,0,"D:SLOVA.TXT"
50 TRAP 200:REM Ak v súbore nie je znak $
60 REM Čítanie súboru po znak $
70 GET #3,A:IF CHR$(A)<>"$"THEN 70
80 REM $ Bol nájdený
90 ? #3;"1234,56";
100 REM Bodkočiarka na konci uvedeného
110 REM príkazu zabráni uloženiu
120 REM EOL-znaku
130 CLOSE #3
140 ? "DATA ZMENENE"
150 GOTO 350
200 ?"V SUBORE NEBOL NAJDENY ZNAK $"
210 GOTO 350
300 ? "SUBOR NEEXISTUJE"
350 END
```

V riadku 30 bude otvorený súbor pre zmenu, v ňom uložených dat. Ak súbor neexistuje, bude program ukončený (r. 300 a 350). Potom bude súbor čítaný znak po znaku, pokiaľ nebude nájdený znak $ (r. 70). Po nájdení znaku $ bude nasledujúcich sedem znakov zmenených číslom 1234,56 (r. 90).

## Ukladanie čísiel do datového súboru.

Na ukladanie čísiel do datového súboru je viacero možností. Možno ste už niektoré skúsili. V nasledujúcom programe sú všetky možnosti zhrnuté.

```
10 ? "OTVORENIE SUBORU"
20 OPEN #1,8,0,"D:DATSUBOR"
30 ? "SUBOR JE OTVORENY"
40 ? #1;"MAM 386 PROGRAMOV"
50 ? #1;1,2,3,4,5
60 A=20:B=15:C=40:D=25:E=50
70 ? #1;A,B+D,C,D*B,E
80 CLOSE #1
90 END
```

Ako vidno, možno číslo uložiť ako časť string-premennej (r. 40), v priamej forme ako konštantu (r. 50), ako číselnú premennú alebo ako aritmetický výraz (r. 70).

Ak ukladáme čísla priamo, môžme ich opäť čítať pomocou príkazu INPUT #;. Toto je ukázané v ďalšom programe:

```
10 REM Otvoriť súbor pre uloženie dat
20 OPEN #2,8,0,"D:CISLA.DAT"
30 REM Zadať 10 čísiel
40 FOR I=1 TO 10
50 ? "ZADAJTE CISLO, KTORE MA"
60 ? "BYT ULOZENE"
70 INPUT A
90 REM Zadané číslo uložiť do súboru
100 ? #2,A:NEXT I
110 CLOSE #2
```

```
120 REM Otvoriť súbor pre čítanie dat
130 OPEN #2,4,0,"D:CISLA.DAT"
140 REM Čítanie 10 čísiel
150 ? "ZADALI STE TIETO CISLA"
160 FOR I=1 TO 10
170 INPUT #2,A
180 REM Čítané čísla zobraziť
190 ? A:NEXT I
200 CLOSE #2
250 END
```

Najskôr bude na diskete otvorený nový súbor (r. 20). Potom musíme zadať 10 čísiel. Každé číslo bude po zadaní uložené (r. 40-100). Pri vykonaní príkazu CLOSE (r. 110) bude aktivovaná disketová jednotka a data uložené v bufferi súboru budú zapísané na disketu. Potom bude súbor otvorený znovu pre čítanie (r. 130). Čísla za sebou čítané budú zobrazené na obrazovke (r. 160-190).

Pre lepšie porozumenie krátky popis ako sú čísla z diskety čítané. Pre počítač končí čítané číslo až ukončením poľa, na konci ktorého je EOL znak. Ak však bude čítané číslo, ktoré bolo uložené s čiarkou, končí číslo tam, ale zo súboru budú čítané ďalšie data, pokiaľ nebude zistený EOL-znak. Znaky, ktoré budú čítané od čiarky po EOL znak, nemajú teda už žiadny vplyv na hodnotu čísla.

Pomocou príkazu PRINT #; možno tiež viac, čiarkou oddelených čísiel, uložiť naraz.

```
300 ? #3;1,2,3,
```

Po vykonaní tohto príkazu nie sú uložené čísla oddelené ani EOL znakom, ani čiarkou. Čiarky, vyskytujúce sa v príkaze PRINT #; sa postarajú o uloženie niekoľkých prázdnych znakou na disketu (viď obr. časť - použitie čiarky v príkaze PRINT #;)

Takto uložené čísla nie je možné čítať jedno po druhom pomocou príkazu INPUT #; a priradiť ich jednej premennej:

```
400 INPUT #3;X
410 INPUT #3;Y
420 INPUT #3;Z
```

Taktiež ich nemožno čítať všetky naraz:

```
500 INPUT #3;A,B,C
```

V oboch prípadoch nastane chyba, lebo medzi uloženými číslami nesmú byť žiadne prázdne znaky.

Keď použijeme za príkazom PRINT #;, k oddeleniu čísiel, miesto čiarky bodkočiarku, budú čísla uložené priamo za sebou, bez vloženia iných znakou. Pri čítaní preto nebudú rozoznané tri čísla ale len jedno.

Obvykle ale nevznikne problém, keď bude za každým číslom uložený EOL-znak.
Pozrite sa na nasledujúci program.

```
10 OPEN #4,8,0,"D:CISLA.DAT"
20 FOR I=1 TO 10
30 ? #4;I;
40 NEXT I
```

```
50 CLOSE #4
```

Desať čísiel bude vyzerať ako jedno veľké číslo, pretože boli uložené primo za sebou. Keď ale na konci príkazu PRINT #; odstránime bodkočiarku, bude po každom čísle uložený EOL-znak a tým sa dosiahne želaný efekt.

Keď oddelujeme ukladané čísla bodkočiarkou, musíme dodatočne vložiť za každé číslo čiarku. Aby sa čiarka uložila aj na disketu, musí byť zadaná v úvodzovkách.

Príklad:

```
150 ? #4;POCET;",";CENA;",";SUMA
```

Ak chceme čísla opäť čítať, musíme dbať na to, aby to nasledovalo v tej istej forme ako pri ukladaní. V tomto prípade to musí vyzerať takto:

```
300 INPUT #4;P,C,S
```

Nie je možné čítať čísla jednotlivo, napríklad takto:

```
300 INPUT #4;P
310 INPUT #4;C
320 INPUT #4;S
```

Potom, keď pri čítaní prvého čísla bude zistená čiarka, hľadá teraz BASIC ako už bolo opísané, po EOL znak. Pri tomto hladaní bude prečítané druhé i tretie číslo, lebo EOL znak je uložený až za tretím číslom. Potom nebudú bezchybne pracovať príkazy INPUT #; v riadkoch 310 a 320. Môžme prirodzene aj pomocou príkazu GET každý znak čítať jednotlivo a preskúšať, či sa jedná o čiarku, alebo EOL-znak. To je ale veľmi prácne.

Zhrnuím možno povedať, že najlepšie je ak sa staráme o to, aby bol po každom čísle uložený EOL-znak. Viac miesta tým nespotrebujeme, ale ušetríme si mnoho roztrpčenia.

## Datové súbory s voliteľným výberom.

Pri tomto spôsobe môžeme vyberať v ľubovolnom poradí a z ľubolného miesta súboru. Toto umožňujú BASIC príkazy NOTE a POINT.

## Príkaz NOTE.

Pomocou NOTE môže byť určená momentálna pozícia ukazovateľa vo vnútri súboru. Údaj pozície pozostáva z čísla naposledy vybraného sektoru a čísla posledného vybraného znaku vo vnútri tohoto sektoru. Nasledujúci riadok je príkladom pre použitie príkazu NOTE.

NOTE #2,SEKT,ZNAK

Bude určená pozícia ukazovateľa súboru otvoreného cez kanál č. 2. Číslo naposledy vybraného sektoru bude uložené do premennej SEKT. Do premennej ZNAK bude uložené poradové číslo znaku, vo vnútri sektoru, na ktorom zostal ukazovateľ stáť. Číslo sektoru sa nevzťahuje na počet sektorov osloveného súboru, ale na počet sektorov na diskete. Prvý sektor súboru nie je teda sektor č. 1. Okrem toho je možné, že sektory nižšieho čísla nebudú ako prvé, lebo súborom obsadené sektory nemusia byť bezpodmienečne uložené po sebe. Ako prvý môže byť napr. sektor č. 137 a druhý č. 142 alebo 118.

## Príkaz POINT.

POINT je opakom NOTE. Pomocou POINT bude ukazovateľ dosadený na určité miesto v súbore. Pre určenie tohoto miesta musíme zadať číslo sektoru a pozíciu znaku v sektore. Príkazy PRINT #;, PUT, INPUT #; alebo GET, budú data od tohoto miesta na diskete zapisovať, alebo z diskety čítať. Nasledujúci riadok je príkladom na použitie príkazu POINT.

```
100 S=125:Z=22:POINT #1,S,Z
```

Ukazovateľ súboru bude dosadený, cez otvorený kanál č. 1, na 22 znak sektoru 125.

Číslo sektoru a znaku musí byť zadané pomocou číselnej premennej. Nie je tu možné vložiť konštantu, hoci hodnota premennej sa pri vykananí príkazu POINT nemení.

Číslo znaku musí byť v rozpätí 0-125 (posledné tri byty každého sektoru sú použité pre vnútorné účely).

Kontrola, či je udaný sektor skutočne súčasťou udaného súboru nebude vykonaná skôr ako začne skutočný zápis. Skôr ako bude začne zápis na disketu, prekontroluje DOS, či pomocou kanálového čísla udaný sektor patrí k udanému súboru. Ak nie, bude vypísaná nasledovná chyba:

ERROR 170, pri čítaní pomocou príkazov INPUT #; alebo GET

ERROR 171, pri zápise pomocou príkazov PRINT #; alebo PUT

## Použitie príkazov NOTE a POINT.

Datové súbory založené pre voliteľný výber, použité pomocou príkazov NOTE a POINT. Keď tieto príkazy použijeme na správnom mieste, môžeme spracovať veľké množstvá dat.

V nasledujúcich odstavcoch bude popísané, ako možno vybudovať datové súbory pre voliteľný výber. Každá metóda má svoje výhody a nevýhody. Pre použitie takýchto súborov sú potrebné dlhé a komplikované programy.

## Indexované datové súbory.

Pri tejto metóde budú použité dva súbory miesto jedného. V jednom súbore budú v jednotlivých záznamoch uložené data, na ktoré musí vedieť program dosiahnuť. Druhý súbor prevezme funkciu indexovania. Pre každý záznam prvého súboru je tu uložené kľúčové slovo. Okrem toho tam bude zachytené, na ktorom mieste záznam začína. Údaj tejto pozície nasleduje pomocou čísla sektoru a znaku. Na začiatku nášho programu budú z diskety čítané tieto informácie, spolu s kľúčovými slovami a pridelené premenným (najčastejšie poliam).

Keď má byť teraz záznam vyhľadaný, program hľadá najskôr v indexovej listine kľúčové slovo pre tento záznam.Potom sa nastaví ukazovateľ prvého súboru na pozíciu začiatku želaného záznamu, pomocou kľúčového slova a príkazu POINT s číslom sektoru a znaku. Teraz možno záznzm pomocou INPUT #; alebo GET čítať, alebo pomocou PRINT #; alebo PUT zapísať.

Ak chceme vložiť záznam medzi dva už existujúce záznamy, je to pri indexovanom datovom súbore záležitosť náročná na čas. Najskôr musíme pre nový záznam urobiť miesto a všetky po ňom nasledujúce záznamy o jedno miesto v smere ku koncu súboru posunúť. Potom zapísať kľúčové slovo nového sektoru do indexového súboru a korigovať čísla sektorov a znakov posunutých záznamom, lebo tieto pozície sa predsa pri posúvaní zmenili. Ak chceme jeden záznam vymazať, je to rovnako časovo náročné. Všetky, po vymazanom, nasledujúce záznamy, musia byť posunuté o jedno miesto k začiatku súboru. Potom musí byť indexovaný súbor upravený ako v predošlom prípade. Pre šetrenie ukladacieho miesta môžeme číslo sektorv a znaku uložiť do jednej string-premennej

K premene znaku string-premennej na číslo a obrátene použijeme funkcie CHR$ a ASC. Potom potrebujeme len dva znaky pre číslo sektoru a jeden znak pre číslo znaku.

Výhodou indexovaného datového súboru je, že určitý záznam bude nájdený veľmi rýchlo. To preto, že pomocou premennej pevne uloženého kľúčového slova môžeme záznam nájsť oveľa rýchlejšie ako keď má byť na diskete prehľadávaný záznam po zázname

## Zreťazené datové súbory.

Druhá metóda spočíva v tom, že uložené záznamy sú navzájom zreťazené.

V tomto prípade obsahuje každý záznam ukazovateľ, ktorý udáva, na ktorom mieste je uložený nasledujúci záznam. Tento ukazovateľ sa skladá z čísiel sektoru a znaku.

Ak hladáme v takomto súbore určitý sektor, musíme začať od prvého záznamu a po ňom všetky nasledujúce čítať, do najdenia želaného. V tomto princípe tu ide o sekvenčný súbor.

Prednosť tohoto súboru je v tom, že môžeme veľmi ľahko

pridať nový záznam, alebo starý vymazať. Ak chceme v zreťazenom datovom súbore vložiť nový záznam, vložíme ho na disketu a príkazom NOTE zistíme na ktorom mieste v súbore je uložený.

Potom musíme určiť po ktorom zázname má novouložený nasledovať. V tomto zázname zmeníme ukazovateľ tak, aby ukazoval na nový uložený záznam. V novouloženom zázname bude uložený ukazovateľ, ktorý ukazuje na záznam, ktorý bol predtým nasledujúci.

Ak chceme záznam vymazať, potom jednoducho zmeníme ukazovateľ v predchádzajúcom zázname tak, aby ukazoval za vymazaný záznam.

Zreťazené datové súbory s indexom.
**********

V zreťazenom datovom súbore s indexom môžeme určitý záznam nájsť veľmi rýchlo.

Takisto aj postup vymazania a vloženia záznamu bude veľmi rýchly. Pre rýchle vyhľadanie určitého záznamu bude použitý, ako pri prvej metóde, indexový súbor. Okrem toho sú záznamy navzájom zreťazené. Tým pri vymazaní alebo vložení záznamu, musí byť zmenený len jeden alebo dva ukazovatele.

Tento postup je veľmi komplikovaný a programovo silný. Použiť túto metódu by sa mali odvážiť len tí, ktorí už majú väčšie skúsenosti v programovaní.

## TIPY A TRIKY

**********
* PRÍDAVNÉ FUNKCIE KLÁVES *
**********

Ľubomír Zvolenský, Atari klub Bratislava

V "Hardwarovej kuchárke" Atari klubu Ostrava bol opísaný spôsob, pomocou ktorého je možné prirobiť si na dosku počítača prídavné klávesy s rozličnými funkciami (stlačením príslušnej klávesy pohyb hore, dole, vpravo a vľavo, vypnutie/zapnutie zvuku klávesnice, vypnutie/zapnutie obrazovky, skok do ľavého horného, a do ľavého dolného rohu, premiestnenie kurzora stlačením jednej klávesy na začiatok alebo na koniec riadku). Avšak získanie takéhoto efektu nepotrebuje žiadne hardwarové zmeny (autor tiež vo svojom článku píše o tom, že k prídavným klávesám netreba žiadny software - pri navrhovaní počítača sa totiž s týmito prídavnými klávesami počítalo). To znamená, že niekde v pamäti ROM s operačným systémom musí byť aj program pre obsluhu prídavných kláves. Stačí iba premiestniť túto rutinu, nadefinovať nové kombinácie kláves (pretože pri návrhu popísaného programu sa rátalo s tým, že ho bude používať iba užívateľ, ktorý nemá zabudované prídavné klávesy), a máme poskytované tie isté funkcie ako s hardwarovou úpravou.

Stlačenie niektorej klávesy spôsobuje odskok na tabuľku

kláves (Key Definitions Table), ktorej adresa je daná vo vektore KEDEFP (pamäťové bunky 121 a 122). Každému čitateľovi už bude jasné, že stačí iba prekopírovať príslušnú rutinu z pamäte ROM, nasmerovať vektor KEDEFP na novú obslužnú rutinu, nadefinovať nové kombinácie kláves a sú poskytované tie isté funkcie, ako pri hardwarovej úprave.

V programe je vynechaný pohyb kurzora stlačením jednej klávesy. Bola realizovaná iba funkcia CONTROL+F3 a funkcie F1 - F4 spolu so stlačenou klávesou SHIFT. Po spustení programu máme k dispozícii tieto nové kombinácie kláves:

SHIFT+CLEAR - kurzor do ľavého horného rohu
SHIFT+INVERSE - kurzor do ľavého dolného rohu
SHIFT+ESC - kurzor na ľavý kraj riadku
SHIFT+RETURN - kurzor na pravý okraj riadku
CONTROL+ESC - vypnutie / zapnutie zvuku klávesnice.

Program aj po stlačení klávesy RESET zachová novú tabuľku na šiestej strane pamäti (oblasť od 1536 do 1791). Treba iba zopakovať sekvenciu príkazov na riadku **20** (POKE 121,0:POKE 122,6).

```
10 TK=PEEK(121)+256*PEEK(122):FOR I=0 TO 191:POKE 1536+I,PEEK
(TK+I):NEXT I:FOR I=0 TO 4:READ A,B:POKE 1536+A,B:NEXT I
20 POKE 121,0:POKE 122,6:END
30 DATA 118,138,103,139,92,140,76,141,156,137
```

```
****************************************
* OCHRANA PROGRAMOV PROTI KOPIROVANIU *
****************************************
```

Počítače ATARI majú tri médiá, na ktorých môžu byť zaznamenané programy - diskety, kazety a cartridge. Každé z týchto médií má svoje špecifické vlastnosti a vyžaduje odlišný spôsob ochrany.

CARTRIGDE

Ako prvým sa budeme zaoberať modulmi ROM pamäte - cartridgami. Cartridge obsahuje systém pamäti EPROM (mazateľná pamäť ROM), v ktorom je zapísaný program. Najjednoduchšou metódou kopírovania programov na cartridgi je ich prepísanie z pamäti do súboru na diskete, alebo kazete. Tento súbor sa však bude musieť načítavať do rovnakej adresovej oblasti, ako súbor na cartridgi, to znamená na adresy od A000-AFFF, a v prípade 8 kB cartridgi aj do oblasti od B000-BFFF. Ochrana pred kopírovaním programov z cartridge je jednoduchá - stačí ak program z cartridge niečo zapíše do svojej adresovej oblasti. Do pamäti EPROM sa normálne nedá nič vpísať a tak si program zistí, že sa jedná o originál. Ten istý program pri čítaní z kazety (alebo diskety) do pamäti RAM prepíše sám seba a stane sa nefunkčným.

Program zapísaný na cartridgi je tiež možné skopírovať aj do druhej pamäte EPROM. Takéto kopírovanie však vyžaduje vlastniť programátor pamätí EPROM. Vzhľadom na náklady a cenu pamätí EPROM je to však dosť drahé. Po prekopírovaní do druhej pamäti EPROM je však program aj naďalej chránený pred ďalším kopírovaním takou istou mierou, ako originál. Ďalšou výhodou je minimálny čas nahrávania programov z cartridge a prakticky nemožnosť ich

poškodenia.

## DISKETOVÉ OCHRANY

Pôvodné ochrany programov na disketách spočívali v znemožnení ich kopírovania cez DOS 2.5. Pre kopírovanie programov DOS 2.5 obsahuje funkcie C - COPY FILES, a D - DUPLICATE FILES. Obsahuje tiež aj funkciu kopírovania diskov - J - DUPLICATE DISK. Všetky tieto kopírovacie súbory využívajú pre kopírovanie zoznam súborov zapísaný v DIRECTORY sektoroch - v sektoroch 361-368. Používajú tiež mapu sektorov - takzvaný VTOC operátor. Pre ochranu programu pred kopírovaním stačí preniesť DIRECTORY sektory a VTOC operátor na iné miesto na diskete, a štandartný DOS už s takouto disketou nedokáže nič spraviť. Vykonanie tejto operácie je jednoduché a je nenáročné na prevedenie. Týmto spôsobom chránený program sa môže skladať z mnohých častí a navyše tieto časti môžu na vlastnej diskete, alebo na diskete týmto programom formátovaným nejaké kontrolné údaje zapísať.

Keď sa však objavili programy kopírujúce diskety sektor po sektore (sektorové kopírovacie programy), prestala táto ochrana byť skutočnou ochranou. Preto boli vymyslené takzvané vadné sektory (bad sectors). Takýto sektor je zapísaný pri inej rýchlosti, alebo na ATARI nekompatibilnej disketovej jednotke. S normálnou disketovou jednotkou sa však tieto sektory nedajú vytvoriť ani prečítať. Ochránený program po spustení skúša prečítať sektor, ktorý bol autorom programu poškodený a program o ňom "vie", a počíta s ním pri používaní. Ak tento sektor neprečíta, je istota, že ide o originál a program sa spustí. Avšak ak sa ten sektor dá prečítať, znamená to, že ide o kópiu programu a tento program sa nespustí. Dokonca sa môže užívateľovi aj nejako "pomstiť".

Avšak aj na takúto ochranu existuje pomoc. Na príslušné miesto na diskete treba zapísať (alebo poškodiť) sektor, a aj kópia programu bude pracovať. Samozrejme, dalo by sa aj z programu vymazať časť, ktorá sa pokúša prečítať patričný sektor, a program by "fungoval" aj na neochránenej diskete. Avšak to mnoho krát býva problematické, pretože chránený program je úplne "rozkúskovaný" po celej operačnej pamäti...

Najnovší spôsob ochrany spočíva vo vytvorení takzvaných neoznačených sektorov (missassigned sectors). Aby sme pochopili, na čom spočíva táto ochrana, musíme sa bližšie oboznámiť so systémom zápisu údajov na diskety.

Disketa je rozdelená na pravidelné kruhy, ktoré sa nazývajú stopami. Na diskete formátovanej cez DOS 2.5 je týchto stôp 40. Stopy obsahujú pravidelné úseky, ktoré sa nazývajú sektormi. Jedna stopa diskety formátovanej Single density a Double density obsahuje 18 sektorov, formátovaná v Medium density obsahuje 26 sektorov. Každý sektor obsahuje 128 bytov informácií (teda programov, dát...), ktoré možno aj čítať, aj zapisovať a posledný sektor na stope obsahuje 44 identifikačných bytov. Práve týchto 44 bytov obsahuje "kľúč" k tajomstvu neoznačených sektorov. V týchto 44-roch bytoch sú zahrnuté tieto informácie:

1) Číslo sektora
2) Číslo stopy
3) Kontrolnú sumu sektora (tzv. CRC kód)
4) Znak začiatku údajov (tzv. DATAMARK)
5) Znak zaplnenia sektora

Neurčené sektory sa tvoria zmenou identifikačných bytov spôsobom, ktorý znemožňuje ich využitie pomocou normálnej disketovej jednotky. Vzhľadom na uskutočnenú zmenu rozlišujeme tri druhy neurčených sektorov:

1) S chybnou kontrolnou sumou (CRC error)

2) S chybným znakom údajov (Bad datamark)
3) Dvojité sektory (Duplicate sectors)

Využitie prvých dvoch druhov ochrany neurčenými sektormi je podobné ako pri vadných sektoroch, hoci obsah sektora môže byť normálnou disketovou jednotkou čítaný. Program však môže kontrolovať chybu pri čítaní sektora, ako aj správnosť jeho obsahu. Trochu iným druhom sú takzvané dvojité sektory.

Normálne sektory na stope majú na diskete čísla od 1 do 18 (v prípade Medium density do 26). So sektormi dvojitými máme do činenia vtedy, keď sú na diskete zapísané dva sektory s rovnakými číslami. V jednom z nich je umiestnená nejaká informácia, napríklad tam môže byť aj časť programu a v druhom sú alebo nuly, alebo tiež nejaká informácia. Program číta prvý sektor s číslom 10 a umiestni ho v bufere. Potom prečíta druhý sektor s číslom 10 a umiestni ho v inom bufere. Teraz príde k porovnaniu obidvoch buferov. Ak je ich obsah rovnaký, program sa nespustí, lebo to značí, že bol načítaný dvakrát ten istý sektor, a teda že disketa nie je originálna. Ak je však obsah týchto dvoch buferov rozličný, program alebo pokračuje v čítaní ďalších údajov, alebo svojich častí, alebo sa spustí.

Kopírovanie takýchto programov vyžaduje špeciálne upravenú disketovú jednotku (so zmeneným operačným systémom).

Úplne odlišná ochrana programov, ktorá nezáleží od použitého média, je ochrana pomocou takzvaných kľúčov. Je to niekoľko obyčajných odporov, prispôsobených pre zapojenie do konektoru a zaliatych živicou. Program potom z príslušného portu zisťuje hodnoty potenciometrov a zisťuje, či sa v porte nachádza príslušný kľúč. Ak je kľúč pravý, až potom sa spustí hlavná (chránená) časť programu. Tento spôsob ochrany je však dosť nevýhodný, pretože je nákladný. Okrem iného aj jednoduchým Basicovským programom je možné "zistiť", aké odpory sa nachádzajú v príslušnom kľúči, a tak vykonanať kópie kľúča. Je možné vytvoriť aj taký kľúč, ktorý by pracoval iba s autorovým programom, ale toto by bolo veľmi nákladné, čo by podstatne zvýšilo cenu programu na trhu.

Ďalší zo spôsobov ochrany je vpisovanie v určitom mieste programu (alebo aj na niekoľkých miestach) jeho sériového čísla. Táto metóda síce neochraňuje program pred kopírovaním, ale umožňuje hneď rozpoznať piráta. Aj tento spôsob ochrany je dosť problematický, pretože vyžaduje dôkladnú registráciu majiteľov. Aj z tohoto dôvodu je prakticky nepoužívaný.

Vzhľadom na absenciu právnej ochrany má odhalenie osoby kopírujúcej niektorý ochránený program skôr morálny charakter.

KAZETOVÉ PROGRAMY

Pôvodná ochrana programov, zaznamenaných na kazetách spočívala v tom, že program sa po nahratí do počítača hneď spustil. Ďalej bolo nutné ochrániť program pred možným zastavením stlačením klávesy BREAK alebo RESET (jednoduché zablokovanie BREAK sa prevedie cez POKE 16,64: POKE 53774,64 - túto operáciu však treba vykonať po vstupe/výstupe pomocou operačného systému ; na RESET je potrebný iný program, uvedený v tomto Spravodaji).

Takto ochránené programy sa nedali skopírovať. Avšak akonáhle sa objavili prvé kopírovacie programy, prestala táto ochrana byť ochranou a vymyslel sa iný spôsob ochrany.

Tento spôsob spočíva v zápise programov na kazety neštandartným formátom - iste sa pamätáte na hry RED MAX 2, FIRE CHIEF a mnohé ďalšie. V takomto prípade ochrany prvý blok programu je nahratý štandartne a po jeho prečítaní sa spustí. V tomto bloku musí byť uložený program pre modifikáciu pôvodného

systému vstup/výstup pre jeho zmenenie na čítanie neštandartným formátom. Tieto modifikácie sú však rôzne a kopírovacie programy nie vždy stačia sledovať ich rozvoj. Avšak program uložený na kazete možno kedykoľvek skopírovať pomocou dvoch kvalitných magnetofónov, alebo pomocou double-decku. Nakoľko však pôvodný operačný systém umožňuje len veľmi pomalý a nespoľahlivý záznam na kazetu a navyše kazeta nie je "inteligentné" médium, nemôžu programy, uložené na kazetách poskytovať toľko možností práce, ako programy uložené na disketách.

Strojové programy to majú ľahké. Ale čo Basic programy?? Na rozdiel od ZX Spektra ATARI neposkytuje možnosť automatického štartu Basic programu (na Spektre sa dá uskutočniť pomocou SAVE "názov" GOTO). Preto bola snaha priniesť program, ktorý umožňuje automatický štart Basic programu. Taký jednoduchý program by mohol byť napríklad tento:

```
HB 10 GRAPHICS 0: SETCOLOR 2,0,0: SETCOLOR 1,0,0: POKE
752,1: POSITION 2,5: PRINT "CLOAD"
NX 20 POSITION 2,20: PRINT "POKE 842,12:GRAPHICS 0:RUN":
POSITION 2,0: POKE 842,13: STOP
```

Tento program na obrazovku vypíše inštrukciu CLOAD pre načítanie Basic programu a GRAPHICS 0: POKE 842,12: RUN pre spustenie tohoto načítaného programu, čo však nevidíme, pretože obrazovka aj písmená budú mať čiernu farbu. Avšak skúsený programátor si z tohoto programu zistí, že automaticky spúšťaný program je v Basicu, a môže ho cez zadanie inštrukcie CLOAD načítať do pamäte bez automatického štartu a následne ho zmeniť. Preto bola snaha priniesť také riešenie, ktoré by tento postup neumožňovalo. Tak vznikol nasledujúci program:

```
FH 1 REM *AUTOSTART PROGRAMOV*
JO 2 REM *(C) 1987  by BAJTEK*
ZK 3 REM **********************
AF 10 FOR G=0 TO 214: READ A: S=S+A
JH 20 POKE 1536+G,A: NEXT G
CC 30 IF S<>13114 THEN PRINT "CHYBA V DATACH!":END
CE 40 OPEN #1,8,128,"C:"
UI 50 POKE 850,11: POKE 852,0
AI 60 POKE 853,6: POKE 856,215
KE 70 POKE 857,0: POKE 858,8
PD 80 A=USR(ADR("hhh[illegible]"),16)
YC 90 CLOSE #1: END
BB 100 DATA 0,2,,0,6,16,6,169,60
PC 110 DATA 141,2,211,169,0,133,65
TI 120 DATA 96,169,148,141,197,2
MB 130 DATA 173,48,2,133,128,173
YR 140 DATA 49,2,133,129,160,4,177
CK 150 DATA 128,133,130,200,177
VH 160 DATA 128,133,131,230,131
HL 170 DATA 230,131,160,169,162,0
ID 180 DATA 189,91,6,145,130,232
YT 190 DATA 136,208,247,169,13,141
XL 200 DATA 74,3,169,0,133,84,169
XM 210 DATA 254,141,179,2,169,127
XJ 220 DATA 141,180,2,76,0,160,0,0
KS 230 DATA 0,0,0,0,0,0,0,0,0,0,0
KU 240 DATA 0,0,0,0,0,0,0,0,0,0,0
KW 250 DATA 0,0,0,0,0,0,0,0,0,0,0
KY 260 DATA 0,0,0,0,0,0,0,0,0,0,0
MF 270 DATA 0,9,9,4,0,0,56,8,50,36
```

```
ZC 280 DATA 33,8,50,51,53,29,56,26
LE 290 DATA 16,14,50,39,26,18,17
KT 300 DATA 12,20,22,23,0,37,43,47
CV 310 DATA 48,26,18,17,12,18,20
CR 320 DATA 24,0,37,43,0,0,47,48
EM 330 DATA 26,2,155,68,44,255,137
RL 340 DATA 155,150,0,68,137,40,52
XP 350 DATA 137,40,151,137,218,253
OQ 360 DATA 130,2,29,4,56,26,9,16
HB 370 DATA 20,8,4,56,0,45,41,36,0
EZ 380 DATA 0,0,0,0,0,0,0,0
```

Pozor!!! V riadku 80 je [*] a [d] v strojovom reťazci v inverzii!!!

Tento program zapíše na kazetu strojový program, ktorý po nahratí do počítača (nahráva sa pri stlačenej klávese START) začne čítať Basic program a po dočítaní ho spustí.

Existuje ešte jeden spôsob automatického štartu programov. Stačí zadať:

```
POKE 65,0:POKE 764,32:X=USR(ADR("[graphics]"))
```

Tento spôsob štartu programov sa dá využiť aj programovo:

```
FG 0 REM *AUTOSTART PROGRAMOV*
KC 1 REM *(C) 1988  by BAJTEK*
ZJ 2 REM **********************
YD 3 POKE 566,158:POKE 580,1
FX 4 TRAP 9
NW 5 DIM X$(40):POKE 764,33
DM 6 GOSUB 8
DJ 7 X=USR(ADR("[graphics]"))
IC 8 GRAPHICS 0:PRINT "                LOADING...":RETURN
SS 9 GRAPHICS 0:PRINT "                 Error ";PEEK(195);" in loading."
JL 10 POKE 566,146:POKE 580,0:NEW
```

Tento program po spustení načíta a spustí náš Basic program.

Existujú ešte dva spôsoby ochrany. Jedným z nich je znemožnenie výpisu premenných programu, čo prakticky znemožňuje akýkoľvek zásah do nášho programu. Táto operácia sa po doladení programu prevedie týmito dvomi riadkami:

```
KJ      32766     FOR    ZN1=PEEK(130)+256*PEEK(131)    TO PEEK(132)+256*PEEK(133):POKE ZN1,155:NEXT ZN1
SN 32767 POKE PEEK(138)+256*PEEK(139)+2,0:CSAVE
```

Program spustíte príkazom GOTO 32766. Ešte pred spustením si pripravte kazetu.

Poslednou ochranou programov zaznamenaných na kazetách je ochrana pomocou zavádzacích tónov (áno, zavádzacích!). Táto úprava sa dá jednoducho previesť pomocou kopírovacieho programu FCOPY 1.60, kde pri výstupe programu na kazetu v hociktorom mieste stlačte klávesu START a podržte ju asi 10 sekúnd. V prestávke medzi blokmi sa nahraje zavádzací tón, ktorý znemožní kopírovanie programu. Takýmto spôsobom ochránený program sa však dá bez akéhokoľvek zavádzača zaviesť do počítača.

Na záver sa nám vynorí otázka : Po spoznaní všetkých popísaných ochrán je vôbec možné spraviť program úplne ochránený proti kopírovaniu? Alebo je možné program tak ochrániť, že by

nebolo vôbec možné ho upraviť? Odpoveď znie: rozhodne nie! Pretože ochrany programov ich kopírovanie iba odďaľujú, avšak ho neznemožňujú. Časom sa určite objaví aj "odchránená" verzia Vášeho programu.

Aby sa sťažila zmena programu, sú data, alebo aj niektoré časti programu nejako ukryté. Napríklad do pamäte sa môže najprv načítavať iba každý druhý byte a prvý sa doplní neskôr. Alebo samotný program je tak ukrytý, že jeho výsledný programový efekt sa získa až po vykonaní operácie EOR, OR alebo AND s príslušnou časťou pamäte. Avšak takéto operácie iba oddialujú kopírovanie ochráneného programu. Na ako dlho, to záleží iba na "fantázii" a šikovnosti autora.

Spracované podľa časopisu Bajtek, číslo 6/1988

## PRERÁBANIE HIER NA NESMRTEĽNOSŤ

Tento článok vznikol zhromaždením údajov, potrebných pre prerábanie hier do nesmrtelnosti. Väčšina údajov o hrách bola zozbieraná z poľského časopisu Bajtek (ročník 1988).

Pre prevedenie hier do nesmrtelnosti je ideálne ak máme disketovú jednotku a disketový monitor, napríklad DISK WIZARD II, ktorého popis je zverejnený v Spravodaji AK Bratislava. Pochopiteľne, že hry možno na nesmrteľnosť prerábať aj monitormi spolupracujúcimi s magnetofónom, no takýto spôsob už je oveľa pracnejší.

### Postup prerábania hier

Do počítača sa nahrá disketový monitor DISK WIZARD II. V hlavnom menu zvolíme funkciu DISK EDIT pomocou klávesy SELECT a stlačením klávesy START vyvoláme toto menu na obrazovku. Následne zvolíme funkciu 3 - Directory, aby sme si zistili, kde je uložená hra, ktorú chceme prerobiť. Do stĺpca FIRST sa zobrazí počiatočný sektor a do stĺpca LEN sa zobrazí dĺžka hry. Tieto údaje si musíme zapísať!!!

Do menu sa vrátime z Directory funkcie stlačením klávesy RETURN. V menu zvolíme funkciu 2 - SCAN SECTORS. Na obrazovku sa zobrazí:

1) BYTE SCAN
2) STRING SCAN

Zvolte 1 - BYTE SCAN (vyhľadávanie údajov podľa zadaných čísel, zodpovedajúcim strojovým inštrukciám). Počítač zobrazí:

ENTER BYTE (0-FF)

Tu musíte zadať hľadané byty. Ich prehľad je uvedený pri jednotlivých hrách. Po odoslaní posledného hľadaného bytu stlačte klávesu RETURN a dostabete ďalšie otázky:

ENTER START SECTOR (1-720) - zadaj sektor, od ktorého sa bude hľadať

ENTER END SECTOR (od-720) - zadaj sektor, po ktorý sa bude hľadať. Tento sektor sa vypočíta, keď začiatočný sektor hry spočítate s jej dĺžkou.

ENTER NUMBER OF BYTES/SECTOR (125/128) - koľko bytov v zadaných sektoroch sa má prezerať? Pri súbore v DOSe je to 125.

Po zadaní tohoto parametru sa zobrazí nápis:

SCANNING SECTOR XXX - hľadám v sektore XXX,

a počítač začne vyhľadávať nami zadané byty. Ak ich nájde,

zobrazí sektor, v ktorom sa tieto byty nachádzajú. Kurzor bude umiestnený na prvom byte zo skupiny bytov hľadaných. Dvakrát stlačte klávesu RETURN pre návrat do menu funkcie DISK-EDIT. Zvolte funkciu 1 - MODIFY SECTORS. Zobrazí sa otázka!

ENTER SECTOR NUMBER: a zadáme číslo sektora, v ktorom bola nájdená nami zadaná sekvencia bytov. Po načítaní tohoto sektora sa v dolnej časti obrazovky zobrazia možné funkcie:

Atascii Enter Mod Nxt + - Prt Wrt

Nás bude zaujímať funkcia MOD - MODIFY (zmena bytu v sektore). Po zvolení tejto funkcie sa vypíše výzva:

ENTER HEX ADRESS (0-7F) - zadaj adresu hľadaného bytu v sektore. Po zadaní sa kurzor nastaví na zadaný byte v sektore. Nasleduje výzva:

ENTER NEW HEX DATA (0-FF) - zadaj novú inštrukciu (ktorá sa zapíše miesto zadaného bytu v sektore). Kurzor sa posunie na nasledujúci byte v sektore.

Zasa je zobrazená výzva : ENTER NEW HEX DATA (0-FF):. Dvakrát stlačte RETURN a dostanete sa do podmenu Atascii Enter Mod...

Zvolte W - WRITE SECTOR. Počítač sa opýta:

WRITE TO WHICH SECTOR (1-720): - zadaj číslo sektoru, pod ktorým sa uloží upravený sektor. Ako pomôcka vám poslúži nápis SECTOR XXX, ktorý je zobrazený nad bytmi sektoru a označuje číslo sektoru, v ktorom sa nachádzali zobrazené byty. Zadaním iného čísla sektoru prichádza ku kolíziam programov, uložených na disketách!

Prehľad hier a inštrukcií, ktoré treba zmeniť:

## ELECTRICIAN

Treba vyhľadať sekvenciu príkazov LDA $08C5 SBC #$01 STA $08C5 (operačné kódy $AD, $C5, $08, $E9, $01, $8D, $C5, $08) a zmeniť inštrukciu SBC #$01 (operačné kódy $E9, $01) na dvakrát NOP (operačné kódy $EA, $EA).

## ARAX

Hľadať SBC #$05 STA $9B8D (op. kódy $E9, $05, $8D, $8D, $9B). Zmeniť SBC #$05 za dve inštrukcie NOP (op. kódy $EA).

## PACMAN

Hľadať DEC $54 a DEC $53 (op. kódy $C6, $54) a $C6, $53. Zmeniť na LDA $54 a LDA $53.

## TWILIGHT WORLD

Hľadať DEC $BA5C (op. kódy $CE, $5C, $BA). Zmeniť na LDA $BA5C (op. kódy $AD, $5C, $BA).

## STEALTH

Hľadať DEC $ABCC (op. kódy $CE, $CC, $AB). Zmeniť na LDA $ABCC (op. kódy $AD, $CC, $AB).

## DESMONDS DUNGEON

Hľadať DEC $1042 (op. kódy $CE, $42, $10). Zmeniť na LDA $1042 (op. kódy $AD, $42, $10).

## SHAMUS

Hľadať DEC $0202 (op. kódy $CE, $02, $02). Zmeniť na LDA $0202 (op. kódy $AD, $02, $02).

## RIVER RAID

Hľadať DEC $5B (op. kódy $C6, $5B). Zmeniť na LDA $5B (op. kódy $A5, $5B) pre nemínanie paliva.

Hľadať CMP $A5 (op. kódy $C5, $A5). Zmeniť na dve inštrukcie NOP (op. kódy $EA, $EA), alebo na CMP $A4 (op. kódy $C5, $A4) pre zľahčenie trasy letu.

## PASTFINDER

Hľadať DEC $F4 (op. kódy $C6,$F4). Zmeniť na LDA $F4 (op. kódy $A5, $F4).

## MEGAMANIA

Hľadať DEC $B3 (op. kódy $C6, $B3). Zmeniť na LDA $B3 (op. kódy $A5, $B3).

## JET BOOT JACK

Hľadať LDA #$05 STA $2464 (alebo pri staršej verzii STA $2468) (op. kódy $A9, $05, $8D, $64, $24), alebo pri staršej verzii je predposledný kód $68. Pre získanie 255-tich životov zmeniť na LDA #$FF STA $2464 (op. kódy $A9, $A9, $FF, $8D, $64, $24). Pre získanie nesmrtelnosti zmeniť DEC $2464 na LDA $2464 (op. kódy $CE, $64, $24 na $AD, $64, $24).

## GREEN BERET

Hľadať DEC $06CC (op. kódy $CE, $CC, $06). Zmeniť na LDA $06CC (op. kódy $AD, $CC, $06).

## DROPZONE

Hľadať DEC $05AD (op. kódy $CE, $AD, $05). Zmeniť na LDA $05AD (op. kódy $AD, $AD, $05).

## STARQUAKE

Hľadať DEC $D2 (op. kódy $C6, $D2). Zmeniť na LDA $D2 (op. kódy $A5, $D2).

## CRYSTAL CASTLE

Hľadať DEC $1BB8,X (op. kódy $DE, $B8, $1B). Zmeniť na LDA $1BB8,X (op. kódy $BD, $B8, $1B) pre získanie nesmrtelnosti pre prvého hráča. Pre získanie nesmrtelnosti aj pre druhého hráča treba zopakovať postup, ale s adresou $1BBB.

## ROBIN HOOD

Hľadať DEC $2DA5 (op. kódy $CE, $A5, $2D). Zmeniť na LDA $2DA5 (op. kódy $AD, $A5, $2D).

## GYRUSS

Hľadať DEC $7F,X (op. kódy $D6, $7F). Zmeniť na LDA $7F,X (op. kódy $B5, $7F). Hľadať aj DEC $AA,X (op. kódy $D6, $AA). Zmeniť na LDA $AA,X (op. kódy $B5, $AA).

## PANTHER

Hľadať DEC $A0 (op. kódy $C6, $A0). Zmeniť na LDA $A0 (op. kódy $A5, $A0).

## MOUSE TRAP

Hľadať DEC $50 (op. kódy $C6, $50). Zmeniť na LDA $50 (op. kódy $A5, $50).

## MONTEZUMAS REVENGE

Hľadať DEC $8D (op. kódy $C6, $8D). Zmeniť na LDA $8D (op. kódy $A5, $8D).

## DONKEY KONG JR.

Hľadať DEC $89,X (op. kódy $D6, $89). Zmeniť na LDA $89,X (op. kódy $B5, $89).

## PINHEAD

Hľadať DEC $82 (op. kódy $C6, $82). Zmeniť na LDA $82 (op. kódy $A5, $82).

## BRUCE LEE

Hľadať DEC $26 (op. kódy $C6, $26). Zmeniť na LDA $26 (op. kódy $A5, $26).

## ZAXXON

Hľadať DEC $9A,X (op. kódy $D6, $9A). Zmeniť na LDA $9A,X (op. kódy $B5, $9A).

## MOLECULE MAN

Hľadať DEC $0605 (op. kódy $CE, $05, $06). Zmeniť na LDA $0605 (op. kódy $AD, $05, $06) pre získanie nekonečného počtu bômb. Hľadať DEC $0606 (op. kódy $CE, $06, $06). Zmeniť na LDA $0606 (op. kódy $AD, $06, $06) pre získanie nekonečného počtu tabletiek. Hľadať DEC $0607 (op. kódy $CE, $07, $06). Zmeniť na LDA $0607 (op. kódy $AD, $07, $06) pre získanie nekonečného počtu funtov.

## URIDIUM

Hľadať LDA #$03 STA $D20F STA $646D (op. kódy $A9, $03, $8D, $0F, $D2, $8D, $6D, $64). Pre získanie 255-tich životov treba v tejto sekvencii zmeniť LDA #$03 za LDA #$FF (op. kódy $A9, $FF). Ostatné príkazy (STA $D20F a STA $646D) zostávajú nezmenené. Pre nekonečný počet životov treba nájsť DEC $9E7F (op. kódy $CE, $7F, $9E) a zmeniť na LDA $9E7F (op. kódy $AD, $7F, $9E). Pre získanie nekonečného počtu bômb treba nájsť inštrukciu DEC $9E93 (op. kódy $CE, $93, $9E) a zmeniť na LDA $9E93 (op. kódy $AD, $93, $9E).

## JAMES BOND 007

Hľadať DEC $89,X (op. kódy $D6, $89). Zmeniť na LDA $89,X (op. kódy $B5, $89).

## LIVEWIRE

Hľadať DEC $C8 (op. kódy $C6, $C8). Zmeniť na LDA $C8 (op. kódy $A5, $C8) pre získanie nekonečného počtu životov. Pre získanie nekonečného počtu superstriel treba nájsť DEC $B1 (op. kódy $C6, $B1) a zmeniť na LDA $B1 (op. kódy $A5, $B1).

## DROL

Hľadať skupinu príkazov LDA $74 SBC #$01 STA $74 (op. kódy $A5, $74, $E9, $01, $85, $74). Zmeniť na LDA $74 NOP NOP STA $74 (op. kódy $A5, $74, $EA, $EA, $85, $74).

## PACMAN 2

Hľadať DEC $23D2 (op. kódy $CE, $D2, $23). Zmeniť na LDA $23D2 (op. kódy $AD, $D2, $23).

## MISS PACMAN

Hľadať DEC $93 (op. kódy $C6, $93). Zmeniť na LDA $93 (op. kódy $A5, $93). Hľadať DEC $94 (op. kódy $C6, $94). Zmeniť na LDA $94 (op. kódy $A5, $94).

## QUIX

Hľadať DEC $52 (op. kódy $C6, $52). Zmeniť na LDA $52 (op. kódy $A5, $52).

## SPECIAL DELIVERY

Hľadať DEC $E5 (op. kódy $C6, $E5). Zmeniť na LDA $E5 (op. kódy $A5, $E5).

KANGAROO

Hľadať DEC $97 (op. kódy $C6, $97). Zmeniť na LDA $97 (op. kódy $A5, $97). Hľadať DEC $98 (op. kódy $C6, $98). Zmeniť na LDA $98 (op. kódy $A5, $98).

## SYNTRON

Hľadať DEC $9D (op. kódy $C6, $9D). Zmeniť na LDA $9D (op. kódy $A5, $9D).

## ATTACK OF MUTANT CAMELS

Hľadať DEC $13BE (op. kódy $CE, $BE, $13). Zmeniť na LDA $13BE (op. kódy $AD, $BE, $13).

## CAVERNS OF MARS

Hľadať DEC $1AAC (op. kódy $CE, $AC, $1A). Zmeniť na LDA $1AAC (op. kódy $AD, $AC, $1A).

## JUMP

Hľadať DEC $801F (op. kódy $CE, $1F, $80). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA).

## ARKANOID

Hľadať DEC $31F4 (op. kódy $CE, $F4, $31). Zmeniť na LDA $31F4 (op. kódy $AD, $F4, $31).

## JET SET WILLY

Hľadať DEC $46F3 (op. kódy $CE, $F3, $46). Zmeniť na LDA $46F3 (op. kódy $AD, $F3, $45).

## FIRE CHIEF

Hľadať DEC $2F80 (op. kódy $CE, $80, $2F). Zmeniť na LDA $2F80 (op. kódy $AD, $80, $2F) pre získanie nekonečného počtu automobilov. Hľadať DEC $5047 (op. kódy $CE, $47, $50). Zmeniť na LDA $5047 (op. kódy $AD, $47, $50) pre získanie nekonečného počtu hasičských kombinéz. Hľadať DEC $2F81 (op. kódy $CE, $81, $2F). Zmeniť na LDA $2F81 (op. kódy $AD, $81, $2F) pre získanie nekonečného množstva paliva. Hľadať DEC $5DB9 (op. kódy $CE, $B9, (5D). Zmeniť na LDA $5DB9 (op. kódy $AD, $B9, $5D) pre získanie nekonečného množstva vody na hasenie ohňa.

DIAMONDS

Hľadať DEC $0605 (op. kódy $CE, $05, $06). Zmeniť na LDA $0605 (op. kódy $AD, $05, $06).

## AIR STRIKE II

Hľadať DEC $0664 (op. kódy $CE, $64, $06). Zmeniť na LDA $0664 (op. kódy $AD, $64, $06). Hľadať DEC $0665 (op. kódy $CE, $65, $06). Zmeniť na LDA $0665 (op. kódy $AD, $65, $06). Hľadať DEC $0666 (op. kódy $CE, $66, $06). Zmeniť na LDA $0666 (op. kódy

$AD, $66, $06).

## SUPER PAC-MAN

Hľadať DEC $05F6 (op. kódy $CE, $F6, $05). Zmeniť na LDA $05F6 (op. kódy $AD, $F6, $05).

## MISTER DO

Hľadať DEC 0618,X (op. kódy $DE, $18, $06). Zmeniť na LDA $0618,X (op. kódy $BD, $18, $06).

## PIEMAN

Hľadať DEC $80 (op. kódy $C6, $80). Zmeniť na LDA $80 (op. kódy $A5, $80).

## PHOBOS

Hľadať DEC $1285 (op. kódy $CE, $85, $12). Zmeniť na LDA $1285 (op. kódy $AD, $85, $12).

## CAVERNS OF MARS II

Hľadať DEC $3B39 (op. kódy $Ce, $39, $3B). Zmeniť na LDA $3B39 (op. kódy $AD, $39, $3B).

## ASTEROIDS

Hľadať DEC $B1,X (op. kódy $D6, $B1). Zmeniť na LDA $B1,X (op. kódy $B5, $B1).

## DIG-DUG

Hľadať DEC $1805,X (op. kódy $DE, $05, $18). Zmeniť na LDA $1805,X (op. kódy $BD, $05, $18).

## SWAT

Hľadať DEC $066B (op. kódy $CE, $6B, $06). Zmeniť na LDA $066B (op. kódy $AD, $6B, $06).

## DAN STRIKES BACK

Hľadať DEC $0608 (op. kódy $CE, $08, $06). Zmeniť na LDA $0608 (op. kódy $AD, $08, $06) pre získanie nekonečného počtu životov. Hľadať LDA $F2 ORA #1 (op. kódy $A5, $F2, $09, $01). Zmeniť na LDA ##$02 NOP NOP (op. kódy $A9, $02, $EA, $EA). Táto úprava spôsobí, že náš najväčší nepriateľ nás nebude naháňať.

## CAPTAIN STYCKYS GOLD

Hľadať DEC $064B (op. kódy $CE, $4B, $06). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA).

## THE GOONIES

Hľadať DEC $112C (op. kódy $CE, $2C, $11). Zmeniť na LDA $112C (op. kódy $AD, $2C, $11).

## MARIO BROS

Hľadať DEC $29 (op. kódy $C6, $29). Zmeniť na LDA $29 (op. kódy $A5, $29).

## CRYSTAL RIDERS

Hľadať DEC $3B9F (op. kódy $CE, $9F, $3B). Zmeniť na LDA $3B9F (op. kódy $AD, $9F, $3B).

## THE LIVING DAYLIGHTS

Hľadať DEC $80AF (op. kódy $CE, $AF, $80). Zmeniť na LDA $80AF (op. kódy $AD, $AF, $80).

## FROGGIE

Hľadať DEC $B8 (op. kódy $C6, $B8). Zmeniť na LDA $B8 (op. kódy $A5, $A8).

## BLAST

Hľadať DEC $2C1A (op. kódy $CE, $1A, $2C). Zmeniť na LDA $2C1A (op. kódy $AD, $1A, $2C).

## BREW BIZ

Hľadať DEC $32E1 (op. kódy $CE, $E1, $32). Zmeniť na LDA $32E1 (op. kódy $AD, $E1, $32). Hľadať DEC $32E2 (op. kódy $CE, $E2, $32). Zmeniť na LDA $32E2 (op. kódy $AD, $E2, $32).

## OLLIES FOLLIES

Hľadať DEV $3680 (op. kódy $CE, $80, $36). Zmeniť na LDA $3680 (op. kódy $AD, $80, $36). Hľadať DEC $3681 (op. kódy $CE, $81, $36). Zmeniť na LDA $3681 (op. kódy $AD, $81, $36).

## GHOST CHASER

Hľadať DEC $27D3 (op. kódy $CE, $D3, $27). Zmeniť na LDA $27D3 (op. kódy $AD, $D3, $27). Hľadať DEC $27D4 (op. kódy $CE, $D4, $27). Zmeniť na LDA $27D4 (op. kódy $AD, $D4, $27).

## LEAPER

Hľadať DEC $1B9F (op. kódy $CE, $9F, $1B). Zmeniť na LDA $1B9F (op. kódy $AD, $9F, $1B).

## ZORRO

Hľadať DEC $0625 (op. kódy $CE, $25, $06). Zmeniť na LDA $0625 (op. kódy $AD, $25, $06).

## ZORRO II

Hľadať DEC $0498 (op. kódy $CE, $98, $04). Zmeniť na LDA $0498 (op. kódy $AD, $98, $04).

## TRAILBLAZER

Hľadať DEC $1D27 (op. kódy $CE, $27, $1D). Zmeniť na LDA $1D27 (op. kódy $AD, $27, $1D). Hľadať DEC $1D28 (op. kódy $CE, $28, $1D). Zmeniť na LDA $1D28 (op. kódy $AD, $28, $1D).

## PRILIMINARY MONTY

Hľadať DEC $88 (op. kódy $C6, $88). Zmeniť na LDA $88 (op. kódy $A5, $88).

## CLOWNS AND BALLONS

Hľadať DEC $EA (op. kódy $C6, $EA). Zmeniť na LDA $EA (op. kódy $A5, $EA). Hľadať DEC $89 (op. kódy $C6, $89). Zmeniť na LDA $89 (op. kódy $A5, $89).

## GREMLINS

Hľadať DEC $53 (op. kódy $C6, $53). Zmeniť na LDA $53 (op. kódy $A5, $53). Hľadať DEC $0B9B,X (op. kódy $DE, $9B, $0B). Zmeniť na LDA $0B9B,X (op. kódy $BD, $9B, $0B).

## MR. ROBOT

Hľadať JMP $8CF4 (op. kódy $4C, $F4, $8C). Zmeniť na NOP $8CF4 (op. kódy $5C, $F4, $8C).

## HERBERT

Hľadať DEC $5086,X (op. kódy $DE, $86, $50). Zmeniť na LDA $5086,X (op. kódy $BD, $86, $50).

## AMAUROTE

Hľadať ADC #$01 STA $63AC (op. kódy $69, $01, $8D, $AC, $63). Zmeniť na ADC #$00 STA $63AC (op. kódy $69, $00, $8D, $AC, $63).

## CAVE LORD

Hľadať LDA $0609 SEC SBC #$0A STA $0609 (op. kódy $AD, $09, $06, $38, $E9, $0A, $8D, $09, $06). Zmeniť na LDA $0609 SEC SBC #$00 STA $0609 (op. kódy ($AD, $09, $06, $38, $E9, $00, $8D, $09, $06). Hľadať LDA $0609 SEC SBC #$0F STA $0609 (op. kódy $AD, $09, $06, $38, $39, $0F, $8D, $09, $06). Zmeniť na LDA $0609 SEC SBC #$00 STA $0609 (op. kódy $AD, $09, $06, $38, $E9, $00, $8D, $09, $06) pre úplné zneškodnenie vtákov.

## H.E.R.O.

Hľadať DEC $39 (op. kódy $C6, $39). Zmeniť na LDA $39 (op. kódy $A5, $39) pre získanie nekonečného počtu životov. Hľadať STA $38C6 (op. kódy $8D, $C6, $38). Zmeniť na STA $38A5 (op. kódy $8D, $A5, $38) pre získanie nekonečného počtu bômb.

## ZEPPELIN

Hľadať DEC $2584 (op. kódy $CE, $84, $25). Zmeniť na LDA $2584 (op. kódy $AD, $84, $25).

## MORKY

Hľadať DEC $0615 (op. kódy $CE, $15, $06). Zmeniť na LDA $0615 (op. kódy $AD, $15, $06).

## LOCO

Hľadať LDA $4FB0 SBC #$01 STA $4FB0 (op. kódy $AD, $B0, $4F, $E9, $01, $8D, $B0, $4F). Zmeniť na LDA $4FB0 NOP NOP STA $4FB0 (op. kódy $AD, $B0, $4F, $EA, $EA, $8D, $B0, $4F).

## PANIC EXPRESS

Hľadať DEC $062E (op. kódy $CE, $2E, $06). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA) pre získanie nekonečného počtu životov. Hľadať DEC $9FA6 (op. kódy $CE, $A6, $9F). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA) pre získanie nekonečného počtu životov.

## ZYBEX

Hľadať DEC $3C7F (op. kódy $CE, $7F, $3C). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA) pre získanie nekonečného počtu životov pre hráča číslo 1. Hľadať DEC $3C88 (op. kódy $CE, $88, $3C). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA) pre získanie nekonečného počtu životov pre hráča číslo 2.

## BASIL

Hľadať DEC $13FD (op. kódy $CE, $FD, $13). Zmeniť na LDA $D01F (op. kódy $AD, $1F, $D0). Táto zmena nám umožní využívať nekonečné množstvo energie, a súčastne nám poskytuje možnosť stlačenia kláves START+SELECT+OPTION naraz, čím sa dosiahne

prerušenie hry a návrat do hlavičky.

### CENTIPEDE

Hľadať DEC $BE (op. kódy $C6, $BE). Zmeniť na LDA #$05 (op. kódy ($A9, $05).

### JUNGLE HUNT

Hľadať DEC $2C (op. kódy $C6, $2C). Zmeniť na NOP NOP (op. kódy $EA, $EA).

### DONKEY KONG

Hľadať DEC $D2 (op. kódy (C6, $D2). Zmeniť na LDA #$05 (op. kody $A9, $05).

### SUBMISSION

Hľadať DEC $7C48 ($CE, $48, $7C). Zmeniť na NOP NOP NOP (op. kody $EA, $EA, $EA).

### KISSIN KOUSINS

Hľadať DEC $6825 ($CE, $25, $68). Zmeniť na LDA $6825 ($AD, $25, $68).

### WHISTLER'S BROTHER

Hľadať LDA #$04 STA $BF (op. kódy $A9, $05, $85, $BF). Zmenením druhého bytu tejto sekvencie získame príslušný počet životov. Pretože chceme z toho "vyťažiť" čo najviac, zameníme druhý byte za číslo 255. Táto úprava nám dá 255 životov, čo bohato stačí. Musíme teda získať cieľové inštrukcie LDA #$FF STA $BF (op. kódy $A9, $FF, $85, $BF).

### BOULDER DASH

Hľadať DEC $07B6 (op. kódy $CE, $B6, $07). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA).

### ROSEN'S BRIGADE

Hľadať DEC $0600 (op. kódy $CE, $00, $06). Zmeniť na LDA $0600 (op. kódy $AD, $00, $06).

### KABOOM

Hľadať DEC $90 (op. kódy $C6, $90). Zmeniť na NOP NOP (op. kódy $EA, $EA). Táto úprava sa týka iba voľby s dvomi hráčmi.

### SPIKY HAROLD

Hľadať DEC $410A (op. kódy $CE, $0A, $41). Zmeniť na LDA $410A (op. kódy $AD, $0A, $41). Táto úprava spôsobuje neubúdanie životov.

Hľadať DEC $4106 (op. kódy $CE, $06, $41). Zmeniť na LDA $4106 (op. kódy $AD, $06, $41). Táto úprava spôsobuje nepribúdanie času.

### DROP ZONE

Hľadať DEC $05AC (op. kódy $CE, $AC, $05). Zmeniť na NOP NOP NOP (op. kódy $EA, $EA, $EA).

# LISTINGY PROGRAMOV

**********
* LISTING PROGRAMU ASA.COM *
**********

Program je zostavený v jazyku C. Listing zdrojového programu sa skladá z troch častí a z linkera, ktorý spája tieto časti.

Listing ASA.C
**********

```
/*Program dos XL to dos 2.5*/
  /*pridava startovaciu adresu*/

  main()${
    int m;
    for ( ; ; )${
      poke(82,0);
      printf("\f  Append start address");
      printf("    verzia 2.2    ");
      printf("  Drahomir Volny 1989 ");
      printf("    AK Bratislava ");
      poke(82,2);
      cclose(3);
      cclose(4);
      printf("\n\n");
      printf("\n\n\t    Directory");
      printf("\n\n\t    Vypis adries");
      printf("\n\n\t    Pridanie adresy");
      printf("\n\n\t    Zmazanie adresy");
      printf("\n\n\t    Navrat do DOSu");
      printf("\n\n\n\t    Vyber si ");
      switch(m=get())${
        case 'D': dir();break;
        case 'V': adresy();break;
        case 'P': appe();break;
        case 'Z': depe();break;
        default : break;$}
      if (m == 'N') ${
        printf("\n\n\n\t    DOS (A/N) ");
        if (get()=='A') break;$};$}
  $}

  /*Vypisuje adresy suboru*/

  adresy()${
    char n[15],s[40];
    int io,i,f,t,p,pio;
    int a,x,y;
    printf("\f\nTlaciaren zap/vyp ");
    switch(get())${
      case 'Z': p=1;break;
      default : p=0;break;$}
```

```
printf("←←←←←←←←←");
if (p==1) $(
   printf("zapnuta");
   printf("\n\nZadaj hlavicku\n\n");
   gets(s);$)
else
   printf("vypnuta");
printf("\n\nZadaj (zar:meno) ");
gets(n);
io = copen(n,'r');
if (io < 0 ) $(
  printf("\n\nCyba cislo %d",-io);
  getend();
  return;$)
if (binf(io)==-1) return;
printf("\n┌──────────┬──────────┐");
printf("\n|Zaciatok| Koniec |");
printf("\n├──────────┼──────────┤");
if (p==1) $(
pio = copen("P:",'w');
if (pio < 0 ) $(
  printf("\n\nCyba cislo %d",-pio);
  getend();
  return;$)
fprintf(pio,"←%s← \n",s);
fprintf(pio,"\n----------------------");
fprintf(pio,"\n|Zaciatok| Koniec |");
fprintf(pio,"\n|--------+--------|");
$)
for ( ; ; ) $(
  if ((f=doubl(io)) == 65535)
    if ((f=doubl(io)) == -1) break;
  if ((t=doubl(io)) == -1) break;
  printf("\n|%6x  |%6x  |",f,t);
  if (f==738 && t==739)
    printf(" init $%x",a=doubl(io));
  else if (f==736 && t==737)
    printf(" run  $%x",a=doubl(io));
  else $(
    x = hi(t)-hi(f);
    y = lo(t)-lo(f);
    if (y<0) $(
      --x;
      y = 256+y;$)
    y = y+256*(x%4)+1;
    for (i=x/4;i>0;--i)
      ciov(io,7,8192,1024,-1,-1);
    if (y>0)
      ciov(io,7,8192,y,-1,-1);$)
  if (p==1) $(
  fprintf(pio,"\n|%6x  |%6x  |",f,t);
  if (f==738 && t==739)
  fprintf(pio," init $%x",a);
  else if (f==736 && t==737)
  fprintf(pio," run  $%x",a);$)$)
printf("\n└──────────┴──────────┘");
if (p==1) $(
fprintf(pio,"\n----------------------");
cclose(pio);$)
```

```
  cclose(io);
  getend();
  return;
$)

/*Pridava startovaciu adresu*/

appe()$(
  char n[15],s[5];
  int io,a,f,t,x,y,ad1,ad2;
  int pole[99],i,j,k,l;
  printf("\f\nInit or Start ?");
  printf("\n\nVyber si ");
  switch(get())$(
    case 'I':ad1=738;ad2=739;break;
    case 'S':ad1=736;ad2=737;break;
    default :return;$)
  printf("\f\nZadaj (zar:meno) ");
  gets(n);
  io = copen(n,'r');
  if (io < 0 ) $(
    printf("\n\nCyba cislo %d",-io);
    getend();
    return;$)
  if (binf(io)==-1) return;
  for (j=1;j<100;++j) $(
    if ((f=doubl(io)) == 65535)
      if ((f=doubl(io)) == -1) break;
    if ((t=doubl(io)) == -1) break;
    pole[j] = f;
    if (f==738 && t==739)
      printf("\n\ninit address $%x",doubl(io));
    else if (f==736 && t==737)
      printf("\n\nrun address  $%x",doubl(io));
    else $(
      x = hi(t)-hi(f);
      y = lo(t)-lo(f);
      if (y<0) $(
        --x;
        y = 256+y;$)
      y = y+256*(x%4)+1;
      for (i=x/4;i>0;--i)
        ciov(io,7,8192,1024,-1,-1);
      if (y>0)
        ciov(io,7,8192,y,-1,-1);$)$)
  cclose(io);
  k = --j/4+1;
  printf("\n");
  for (l=1;l<=k;++l)$(
    a=l;
    printf("\n%3d%5x",a,pole[a]);
    if ((a=l+k)<=j)
      printf("%4d%5x",a,pole[a]);
    if ((a=l+2*k)<=j)
      printf("%4d%5x",a,pole[a]);
    if ((a=l+3*k)<=j)
      printf("%4d%5x",a,pole[a]);$)
  printf("\n\nVyber si ");
  gets(s);
```

```
    a=val(s);
    if (a==0) ${
      printf("\n\nZadaj adresu (hex) ");
      gets(s);
      a=hval(s);
      pole[0]=a;
      a=0;$}
    if (pole[0]==0) return;
    printf("\n\nnew address %x",pole[a]);
    io = copen(n,'a');
    if (io < 0 ) ${
      printf("\n\nCyba cislo %d",-io);
      getend();
      return;$}
    cputc(ad1 % 256,io);
    cputc(ad1 / 256,io);
    cputc(ad2 % 256,io);
    cputc(ad2 / 256,io);
    i = lo(pole[a]);
    j = hi(pole[a]);
    cputc(i,io);
    cputc(j,io);
    cclose(io);
    printf("\n\nok");
    getend();
    return;
  $}

  binf(io)
  int io;${
  if (cgetc(io)!=255 || cgetc(io)!=255)${
    printf("\n\nNieje binarny subor");
    cclose(io);
    getend();
    return(-1);$}
  else return (1);
  $}
```

Listing ASA1.C

```
/*Zmazanie startovacej adresy*/

  depe()${
    char n[15],s[15];
    int i,j,a,b,c,st1,st2,x,y,z,f,t;
    printf("\f\nZadaj stare (zar:meno) ");
    gets(n);
    printf("\n\nZadaj nove  (zar:meno) ");
    gets(s);
    i = copen(n,'r');
    if (i < 0 ) ${
      printf("\n\nCyba %d pri citani",-i);
      getend();
      return;$}
    if (binf(i)==-1) return;
    j = copen(s,'w');
```

```
  if (j < 0 ) $(
    printf("\n\nCyba %d pri zapise",-j);
    getend();
    return;$)
  cputc(255,j);
  cputc(255,j);
  for ( ; ; ) $(
    if ((f=dbl(i,j)) == 65535)
      if ((f=dbl(i,j)) == -1) break;
    if ((t=dbl(i,j)) == -1) break;
    if (f==738 || f==736) $(
      printf("\n\nok");break;$)
    x = hi(t)-hi(f);
    y = lo(t)-lo(f);
    if (y<0) $(
      --x;
      y = 256+y;$)
    y = y+256*(x%4)+1;
    for (z=x/4;z>0;--z) $(
      ciov(i,7,8192,1024,-1,-1);
      if(ciov(j,11,8192,1024,-1,-1)<0)
        $(
        printf("\n\nCyba");
        getend();
        return; $)$)
    if (y>0) $(
      ciov(i,7,8192,y,-1,-1);
      if(ciov(j,11,8192,y,-1,-1)<0)
        $(
        printf("\n\nCyba");
        getend();
        return; $)$)$)
  cclose(i);
  cclose(j);
  getend();
  return;
$)

/*Kopiruje dva byte z kanala i na j*/

dbl(i,j)
int i,j;$(
  int d1,d2,d;
  if ((d1=cgetc(i)) < 0) return (-1);
  if ((d2=cgetc(i)) < 0) return (-1);
  d=d1+256*d2;
  if (d)=736 && d(=739) return (d);
  if (cputc(d1,j) < 0)
    printf("\n\nChyba pri zapise");
  if (cputc(d2,j) < 0)
    printf("\n\nChyba pri zapise");
  return (d);
$)

/*Cakanie na ukoncenie podrogramu*/

getend()$(
  printf("\n\nstlac klavesu ");
  get();
```

```
  return;
$)

/*Vstup znaku z klavesnice*/

get()$(
  int i;
  cclose(3);
  open(3,4,0,"K:");
  i = cgetc(3);
  cclose(3);
  return(i);
$)

/*Vybera dva byte z kanala io*/

doubl(io)
int io;$(
  int d1,d2;
  if ((d1=cgetc(io)) < 0) return (-1);
  if ((d2=cgetc(io)) < 0) return (-1);
  return (d1+256*d2);
$)

hi(a)
int a;$(
  int d;
  if (a<0)$(a=a&32767;d=128;$)
    else d=0;
  return(a/256+d);
$)

lo(a)
int a;$(
  a=a&32767;
  return(a%256);
$)


     Listing ASA2.C
     **************

dir()
  $(
    char str[20],fs[15],*sp;
    printf("\f\nDrive 1,2,3,4,8 ");
    switch (get()) $(
      case '1':sp="D1:*.*";break;
      case '2':sp="D2:*.*";break;
      case '3':sp="D3:*.*";break;
      case '4':sp="D4:*.*";break;
      case '8':sp="D8:*.*";break;
      default :
        printf("\n\nZadaj (zar:meno) ");
        gets(fs);sp=fs;break;$)
    open(3,6,0,sp);
    printf("\n");
    while (cgets(str,3)>0)
      printf("\n%s",str);
```

```
  getend();
  return;
$)

cgets(str,iocb)
char *str;
int iocb;
$(
  int r;
  if((r=ciov(iocb,5,str,120,-1,-1))<0)
    return r;
  str[r=dpeek(840+16*iocb)-1]=0;
  return r;
$)
```

Listing ASA.LNK
***************

```
asa
asa1
asa2
aio
printf
dbc.obj
```

**********
* LISTING OKNA NA ATARI XE XL *
**********

```
1 REM
2 REM * OKNA PRE ATARI XE/XL *
3 REM
5 REM
10 AD=40448:RESTORE 100:POKE 106,158:GRAPHICS 0
20 FOR I=0 TO 37:SUMA=0:FOR J=0 TO 9
30     READ A:POKE AD+10*I+J,A:SUMA=SUMA+A:NEXT J:READ S
40      IF  S<>SUMA  THEN  ?  "CHYBA  DAT  NA RIADKU ";100+I:LIST
100+I:END
50 NEXT I:POKE 812,79:POKE 813,0:POKE 814,158
60    ?    CHR$(125):POKE  205,14:POKE  206,4:OPEN  #1,13,9,"D":?
#1;CHR$(29);"   HOTOVO"
90 POSITION 2,20:END
100 DATA 14,158,248,158,248,158,251,158,248,158,1799
101 DATA 248,158,76,249,158,189,74,3,133,203,1491
102 DATA 189,75,3,133,204,165,87,240,3,160,1259
103 DATA 255,96,56,165,205,201,3,176,3,160,1320
104 DATA 200,96,165,206,201,3,176,3,160,200,1410
105 DATA 96,24,165,203,101,205,133,205,56,201,1389
106 DATA 40,144,3,160,200,96,165,204,101,206,1319
107 DATA 133,206,56,201,24,144,3,160,200,96,1223
108 DATA 165,88,133,207,165,89,133,208,162,0,1350
109 DATA 232,24,169,40,101,207,133,207,169,0,1282
110 DATA 101,208,133,208,228,204,208,238,169,0,1697
111 DATA 164,203,145,207,200,196,205,208,249,24,1801
112 DATA 169,40,101,207,133,207,169,0,101,208,1335
```

```
113 DATA 133,208,232,228,206,208,227,198,6,198,1844
114 DATA 205,165,88,133,207,165,89,133,208,162,1555
115 DATA 0,232,24,169,40,101,207,133,207,169,1282
116 DATA 0,101,208,133,208,228,204,208,238,169,1697
117 DATA 81,164,203,145,207,200,169,82,145,207,1603
118 DATA 200,196,205,208,249,169,69,145,207,24,1672
119 DATA 169,40,101,207,133,207,169,0,101,208,1335
120 DATA 133,208,169,124,164,203,145,207,164,205,1722
121 DATA 145,207,232,228,206,208,228,164,203,169,1990
122 DATA 90,145,207,200,169,82,145,207,200,196,1641
123 DATA 205,208,249,169,67,145,207,165,203,133,1751
124 DATA 4,165,204,133,5,230,4,230,5,160,1140
125 DATA 1,96,170,165,203,133,82,165,205,133,1353
126 DATA 83,165,84,72,165,85,72,165,4,133,1028
127 DATA 85,165,5,133,84,138,201,125,208,11,1155
128 DATA 104,133,85,104,133,84,230,205,76,80,1234
129 DATA 158,32,176,242,165,85,133,4,165,84,1244
130 DATA 133,5,165,4,197,203,208,8,165,205,1293
131 DATA 133,4,198,4,198,5,197,205,208,8,1160
132 DATA 165,203,133,4,230,4,230,5,165,5,1144
133 DATA 197,204,208,6,165,206,133,5,198,5,1327
134 DATA 197,206,208,6,165,204,133,5,230,5,1359
135 DATA 104,133,85,104,133,84,169,2,133,82,1029
136 DATA 169,39,133,83,169,30,32,176,242,169,1242
137 DATA 31,32,176,242,160,1,96,0,0,0,738
```

```
**************************************
* LISTING DEMO OKNA NA ATARI XE XL *
**************************************
```

```
10 ? CHR$(125)
20 OPEN #1,4,0,"K":DIM A$(120):POKE 752,1
30  FOR  I=1  TO 9:READ A1,A2,A3,A4:POKE 205,A3:POKE 206,A4:CLOSE
#2:OPEN #2,A1,A2,"O"
40 READ A$:? #2;CHR$(29);" ";A$:GET #1,A
50 NEXT I
60 GRAPHICS 0:END
100  DATA  3,4,24,7,NIE  LEN  VELKE  POCITA-CE MOZU VYUZIT SYSTEM
OKIEN.
110  DATA 5,8,33,6,AJ NA TVOJOM OSEMBITOVOM POCI-TACI ATARI MOZNO
ZISKAT PODOBNEEFEKTY.
120  DATA  2,15,30,6,SAMOZREJME  ZE  NIE  SU TAKEJ KVALITY AKO NA
VELKYCH POCI-TACOCH.
130  DATA  18,3,21,8,NEMUSIS  NARIEKAT    TENTO PROGRA USPO- KOJI
TVOJE ZAKLADNEPOZIADAVKY.
140  DATA  10,13,21,7,POVEDZ SAM CI TO  CO MAS NA OBRAZOVKENIE JE
PEKNE?
150  DATA  8,9,27,7,AK  UZNAS  ZE JE MYSLIM ZEBUDES TENTO PROGRAM
VYU- ZIVAT.
160 DATA 4,1,23,18,OBSLUHA OKIEN JE    JEDNODUCHA - POUZIVA-JU SA
VBUDOVANE KANA-LY IOCB.
170  DATA 4,1,23,18,SURADNICE LAVEHO    HORNEHO ROHU SA ZADA-VAJU
AKO PARAMETRE   INSTRUKCIE OPEN.
180  DATA  4,1,23,18,ROZMERY  OKNA  TREBA    ESTE PRED VYVOLANIM
INSTRUKCIE OPEN VPI- SAT DO ADRIES 205    (WX) A 206 (WY).
```

```
**********************
* OZDOBNE PISMO *
**********************
```

Program OZDOBNE PISMO umožňuje vytvoriť ozdobné písmo, a používať ho spolu s normálnou znakovou sadou. Znaková sada s novým ozdobným písmom sa aktivuje príkazom POKE 756,156, a vypne POKE 756,224. Nie je umožnené používať obidva druhy písma (ozdobný a štandartný) naraz. Užívateľ však môže použiť ozdobné písmo pre vypísanie hlavičky, alebo potrebných údajov a potom znovu používať štandartné znaky. Po spustení BASICovského programu je vygenerované nové písmo a BASICovský program sa môže vymazať. Ozdobné písmo zostane zachované až do jeho prepísania, alebo do uskutočnenia studeného štartu, pri ktorom je vymazaná celá pamäť.

```
32000 RESTORE 32100:CHBAS=PEEK(740)-4:POKE 106,CHBAS:POKE 559,0
32020 POKE 756,CHBAS:CHBAS=CHBAS*256:FOR I=0 TO 1023:READ X:POKE
CHBAS+I,X:NEXT I:POKE 559,34:END
32100 DATA
0,0,0,0,0,0,0,0,56,124,124,124,56,0,56,0,238,238,238,0,0,0,0
32101 DATA
0,108,254,254,108,254,254,108,48,124,176,124,54,254,252,48,0,198,
06,28,56,112,230,198
32102 DATA
0,124,238,124,237,198,238,125,0,24,24,24,48,0,0,0,0,60,120,112,11
,112,120,60
32103 DATA
0,120,60,28,28,28,60,120,0,102,126,60,60,126,102,0,0,56,56,254,25
,56,56,0
32104 DATA
0,0,0,0,56,56,56,112,0,0,0,254,254,0,0,0,0,0,0,0,56,56,56,0
32105 DATA
0,6,14,28,56,112,224,192,0,124,230,238,246,230,254,124,0,56,120,2
8,56,56,254,254
32106 DATA
0,124,206,30,60,120,254,254,0,124,30,62,28,30,254,124,0,28,60,108
254,254,28,28
32107 DATA
0,254,224,252,14,230,254,124,0,124,224,252,246,254,254,124,0,254,
0,30,60,120,240,224
32108 DATA
0,124,238,124,238,254,254,124,0,124,238,254,126,14,254,252,0,0,56
56,0,56,56,0
32109 DATA
0,0,56,56,0,56,56,112,0,14,30,60,120,60,30,14,0,0,254,254,0,254,2
4,0
32110 DATA
0,224,240,120,60,120,240,224,0,124,206,14,60,56,0,56,0,124,230,23
,238,224,254,124
32111 DATA
0,124,246,251,246,246,246,246,0,252,238,252,238,238,254,252,0,124
230,224,230,254,254,124
32112 DATA
0,248,236,230,238,254,254,252,0,254,240,252,240,254,254,254,0,254
240,252,240,240,240,240
32113 DATA
0,126,224,238,230,254,254,126,0,246,246,254,246,246,246,246,0,254
56,56,56,56,254,254
32114 DATA
0,30,30,30,30,222,254,124,0,230,238,252,248,252,238,238,0,240,240
```

```
240,240,240,254,254
32115 DATA
0,238,254,254,214,198,198,198,0,230,246,254,254,238,230,230,0,124
230,230,254,254,254,124
32116 DATA
0,252,230,230,254,252,224,224,0,124,230,230,238,252,254,118,0,252
230,230,254,252,238,238
32117 DATA
0,124,224,124,6,126,254,124,0,254,120,120,120,120,120,120,0,246,2
6,246,246,254,254,124
32118 DATA
0,246,246,246,254,254,124,56,0,198,198,214,254,254,238,198,0,246,
46,124,56,124,246,246
32119 DATA
0,246,246,254,124,56,56,56,0,254,30,60,120,240,254,254,0,124,112,
12,112,112,112,124
32120 DATA
0,192,224,112,56,28,14,6,0,124,28,28,28,28,28,124,0,16,56,124,238
198,0,0
32121 DATA
0,0,0,0,0,0,255,255,0,108,254,254,254,124,56,16,24,24,24,31,31,24
24,24
32122 DATA
3,3,3,3,3,3,3,3,24,24,24,248,248,0,0,0,24,24,24,248,248,24,24,24
32123 DATA
0,0,0,248,248,24,24,24,3,7,14,28,56,112,224,192,192,224,112,56,28
14,7,3
32124 DATA
0,238,0,124,6,126,238,126,238,0,246,246,246,254,254,124,0,238,0,1
4,246,254,254,124
32125 DATA
0,238,0,246,246,246,254,124,0,248,236,248,236,252,248,224,255,255
0,0,0,0,0,0
32126 DATA
0,0,0,0,0,0,255,255,0,0,0,0,56,56,56,112,0,28,28,119,119,8,28,0
32127 DATA
0,0,0,31,31,24,24,24,0,0,0,255,255,0,0,0,24,24,24,255,255,24,24,2

32128 DATA
0,0,60,126,126,126,60,0,0,198,124,230,230,254,254,124,192,192,192
192,192,192,192,192
32129 DATA
0,0,0,255,255,24,24,24,24,24,24,255,255,0,0,0,0,198,124,246,254,2
6,246,246
32130 DATA
24,24,24,31,31,0,0,0,120,96,120,96,126,24,30,0,0,16,56,124,254,56
56,56
32131 DATA
0,56,56,56,254,124,56,16,0,16,48,126,254,126,48,16,0,16,24,252,25
,252,24,16
32132 DATA
0,24,60,126,126,60,24,0,0,0,124,14,126,238,254,126,0,224,224,252,
46,246,254,252
32133 DATA
0,0,124,240,240,252,252,124,0,14,14,126,238,238,254,126,0,0,124,2
8,254,224,254,124
32134 DATA
0,62,120,254,120,120,120,120,0,0,126,238,126,14,254,252,0,240,240
252,254,246,246,246
```

```
32135 DATA
0,120,0,248,120,120,120,252,0,30,0,30,30,30,254,124,0,224,224,238
252,252,238,230
32136 DATA
0,120,56,56,56,56,124,124,0,0,108,254,254,214,214,198,0,0,252,254
246,246,246,246
32137 DATA
0,0,124,246,254,254,254,124,0,0,252,246,254,252,224,224,0,0,126,2
8,254,126,14,14
32138 DATA
0,0,220,254,240,240,240,240,0,0,126,240,124,14,254,252,0,120,254,
20,120,120,126,62
32139 DATA
0,0,246,246,246,246,254,124,0,0,246,246,246,254,124,56,0,0,198,21
,254,254,254,108
32140 DATA
0,0,246,124,56,124,246,246,0,0,246,246,126,60,252,248,0,0,254,60,
20,240,254,254
32141 DATA
0,24,60,126,126,24,60,0,24,24,24,24,24,24,24,24,0,126,120,124,110
102,6,0
32142 DATA 0,8,24,56,120,56,24,8,0,32,48,56,60,56,48,32

10 GRAPHICS 18:SETCOLOR 2,0,0:SETCOLOR 4,0,0:POKE
756,156:POSITION 0,0
50 PRINT #6;"       starsoft":? #6;"       starsoft":? #6;"
STARSOFT"
60 DL=PEEK(560)+256*PEEK(561):POKE DL+9,6:? #6:? #6;"
PRESENTS:":? #6:? #6:? #6;"       LETTERS"
70 POKE DL+15,2:POSITION 0,10:? #6;"2.3V COPYRIGHT (C) Lubos
ZVOLENSKY, 1989";:SETCOLOR 2,0,0:SETCOLOR 4,0,0
80 POKE 756,156:FOR A=1 TO 250:POKE 709,A:POKE
710,ABS(A-150):POKE 711,INT(A/16):NEXT A:POKE 756,224
110 FOR A=1 TO 250:POKE 709,A:POKE 710,ABS(A-150):POKE
711,INT(A/16):NEXT A:GOTO 80
```

**********
* BACKGROUND PRINTER *
**********

V Spravodaji AK Olomouc 1-2/1989 bol uverejnený program BACKGROUND PRINTER, ktorý (ako sa podľa popisu zdalo) môže byť výborným pomocníkom pri práci s tlačiarňou. Avšak v programe sa vyskytla chyba, preto na stranách tohoto Spravodaja uverejňujeme jeho správny listing.

```
10 REM *BACGROUND PRINTER*
50 REM *Printer active : LIST "B:NAME"
60 REM *File (NAME) is saved on disk
70 REM *Read sector to sector and
80 REM *Print this sectors on printer
90 GRAPHICS 0:DATA
0,1,2,3,4,5,6,7,8,9,0,0,0,0,0,0,0,10,11,12,13,14,15
110 DIM DAT$(96),HEX(22):FOR X=0 TO 22:READ N:HEX(X)=N:NEXT
X:LINE=490:RESTORE 500:TRAP 190:? "CHECKING DATA"
120 TOTAL=0:LINE=LINE+10:POSITION 2,2:? "LINE:";LINE:READ DAT$:IF
LEN(DAT$)<>96 THEN 220
130 DATLIN=PEEK(183)+256*PEEK(184):IF DATLIN<>LINE THEN ? "LINE
";LINE;" MISSING!":END
140 FOR X=1 TO LEN(DAT$)-1 STEP
2:D1=ASC(DAT$(X,X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=HEX(D1)+16*H
```

```
X(D2)
150 IF PASS=2 THEN PUT #1,BYTE:NEXT X:READ CHKSUM:GOTO 120
160 TOTAL=TOTAL+HEX(D1)+HEX(D2):NEXT X:READ CHKSUM:IF
TOTAL=CHKSUM THEN 120:GOTO 220
190 IF PEEK(195)<>6 AND PEEK(195)<>5 THEN 220
200 IF PASS=0 THEN OPEN
#1,8,0,"D:AUTORUN.SYS":PASS=2:LINE=490:RESTORE 500:TRAP 210:?
"CREATING FILE":GOTO 120
210 CLOSE #1:END
220 IF LEN(DAT$)=30 AND LINE=660 THEN TRAP 200:GOTO 130
230 ? "BAD DATA IN LINE ";LINE:END
500 DATA
FFFF00220E221522822279227A22792279224C7A2216221123A5CCF0298A48A20
A90B9D4203A9FA9D4403A9239D4503,545
510 DATA
A9119D4803A9009D49032056E468AAA081989D430360A5CDD0D3A9E69D4403A92
9D4503A90185CBA000B91A03C944F0,597
520 DATA
05C8C8C810F4C8B91A0385CEC8B91A0385CFA4CBB1CE4888B1CE48A5D06085D0A
0785CBD0D2A5CBC907D0F1A90385CB,728
530 DATA
205522A210A9039D4203A9E69D4403A9239D4503A9049D4A032056E4A9269D420
2056E4BD4C0385CCBD4D0385CDA90C,587
540 DATA
9D42032056E4A90085CE85D085CF85D1A974850AA9E4850BA907A0E2A222205CE
604C62E4A5D1D0F9A5D0D02D85CB8D,659
550 DATA
1522A9018D0103A9528D0203A90B8D0403A9248D0503A5CC8D0A03A5CD8D0B032
53E4983012230A24CCE6D0D0C8EE15,563
560 DATA
22AD1522C902D0BEA9008D1522A5CFD07BE6D1A4CEB90B2448A057A228A94E8D0
038C0203861EA61D689DC003E8E41E,649
570 DATA
F015861DC99BF005A0014C8F23A9209DC003E8E41ED0F8A900851DA2C0A0038E0
038C0503A9408D0003A9018D0103A9,565
580 DATA
808D0303A51E8D0803A9008D0903A51C8D06032059E4C6D1981004E6CFD00AA4C
C8CC8A24B02584CE4CDF22C6CFE6D1,656
590 DATA
4C6023200000AD13228D2402AD14228D2502A98B850AA924850BD0DDA90085CE8
D0AD8824290385CDF002E6CBAD8924,587
600 DATA
85CCF002E6CBA5CBF0C94CDF22443A5052494E542E53504C443A4D454D2E53415
D0D2C9CED4AED3D0CCA0C9CEA0D5D3,711
610 DATA
C59B8B247D25A210A90D9D4203A9F19D4403A9239D45032056E4C0AAD034A000B
1A03C942F005C8C8C810F4A900991A,593
620 DATA
03A9008DE702A9228DE802AD0F22850CAD1022850DAD1122850AAD1222850B6C0
00AD11228500AD12228501600000A9,512
630 DATA
0085CC85CDAD0F228DF724AD10228DF8242000004C0E25A50C8D0F22A50D8D102
A9E4850CA924850DA50A8D1122A50B,595
640 DATA
8D1222A98B850AA924850BAD24028D1322AD25028D1422A97E8DE702A9258DE80
A000B91A03F00CC950D003205B25C8,573
650 DATA
C8C810EFA942991A03C8A900991A03C8A922991A03609848C8B91A0385CEC8B91
0385CFA00218B1CE69018DAD23C8B1,651
660 DATA CE69008DAE2368A860E002E102FC24,195
```

SPRAVODAJ ATARI KLUBU BRATISLAVA č.4/1989
Vydal ATARI KLUB Bratislava, 301. ZO Zväzarmu
Vedúci redaktor: Ing. Vladimír Zvolenský
Neprešlo jazykovou úpravou.

Dané do tlače: september 1989
Tlač povolil: Národný výbor hl. mesta Bratislavy
Číslo povolenia: 57/1988
NEPREDAJNÉ - len pre vnútornú potrebu ČK

Tlač: SLUŽBA VD, Rustaveliho 7


Naskenovanie a prekonvertovanie
podkladov do pdf
Igi 2019

zdroj: archív Igi

http://blog.3b2.sk/igi