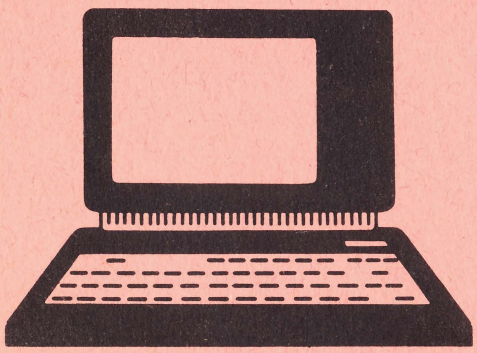
Karel Haupt

BASIC-EXP V5.0/G

pro počítač ONDRA SPO 186

/příručka pro začátečníky/



Komenium, n.p., Praha
1988

Ing. Karel Haupt

B A S I C - E X P V5.Ø/G pro počítač ONDRA SPO 186

/příručka pro začátečníky/

Komenium, n.p., Praha

1988

Autor:Ing.Karel Haupt
Recenzenti:RNDr.Miloslav Feil,CSc.,
Karla Javorková

O b s a h

Programovat se naučíme pouze p r o g r a m o v á n í m .

BASIC proto studujme postupně. Jednotlivé příklady a programy si ověřujme nejen na počítači, ale podle vysvětlujícího textu si je zkoušejme sami sestavit.
Brzy zjistíme, že mnohé z nich budou dokonalejší
- kratší, s vyšším komfortem obsluhy apod.,
než jejich základní verze uvedené v této příručce.

Zkoušejme si sestavovat i další programy podle svých vlastních potřeb - tedy podle svého vlastního zadání.

Úvodem

Osobní počítače se postupně stávají všední záležitostí. Možná, že pro generaci dnešních žáků základních škol bude časem využívání osobních počítačů stejně běžné, jako je dnes používání ostatních přístrojů spotřební elektroniky.

Příručky č.1 "Návod k obsluze" a č. 4 "BASIC", dodávané výrobcem počítače ONDRA, mají dohromady více než 360 stránek a ke každému počítači jsou dodávány jen v jednom výtisku. Zkušenosti ukazují, že při současném pracovním i mimopracovním zatížení učitelů i žáků se s nimi seznámí jen úzký okruh možných uživatelů počítače ONDRA.

Ostatním zájemcům o práci s počítačem ONDRA je určena tato, podstatně kratší příručka. Vychází se v ní z poznatků a zkušeností z kroužku výpočetní techniky na základní škole a z obvodního střediska elektroniky pro děti a mládež. Těmto poznatkům je podřízen výběr základních instrukcí operačního systému, klíčových slov, obsah jednotlivých kapitol i způsob výkladu. Těžiště je v konkrétních řešených příkladech, které zahrnují nejen výpočty, operace se řetězci a základní grafické příkazy, ale i jednoduché počítačové hry.

Je samozřejmé, že úspěšnost a rychlost postupu žáka podle této příručky je podmíněna i jeho konkrétními znalostmi z matematiky. Většinu kapitol však mohou studovat i žáci s minimálními znalostmi z matematiky. Pomoc učitele by měla být spíše jen v rozšiřování jejich matematických znalostí.

> Počítače se nebojme. Jeho obsluha je vlastně docela jednoduchá. Stačí si jen představit, že místo počítače máme před sebou "chytrý psací stroj s obrazovkou".

1. Základní informace

Osobní počítač ONDRA je výkonný elektronický přístroj umožňující provádět různé operace s čísly, s písmeny a s dalšími grafickými znaky. Je napájen ze samostatného napájecího zdroje. Zapnutím nebo vypnutím napájecího zdroje současně zapneme nebo vypneme i počítač.

Jako trvalé vnější paměti počítače se používá magnetofon. Nejvhodnější je typ s dálkovým elektrickým spouštěním a zastavováním motorku magnetofonu.

Videosignál z počítače se přivádí do upraveného černobílého televizoru /upravené čs. televizory PLUTO nebo MERKUR/.

1.1 Propojení systému počítače ONDRA

Počítač ONDRA je elektrický spotřebič zařazený do první izolační třídy. Jeho napájecí zdroj proto smí být připojen pouze do zásuvky s kolíkem zapojeným na ochrannou soustavu elektrické rozvodné sítě!

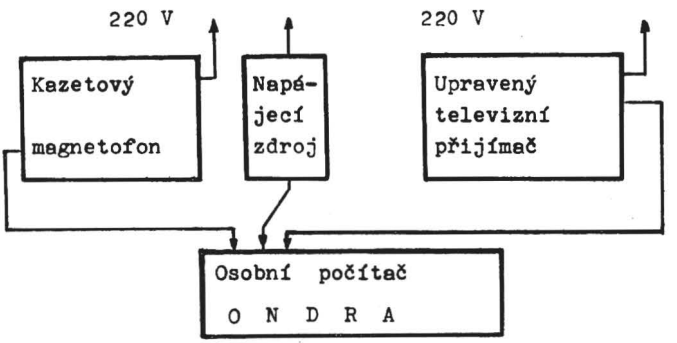
Ještě dříve, než televizor, magnetofon a napájecí zdroj počítače připojíme do elektrické rozvodné sítě, zkontrolujeme, zda jsou všechny tyto přístroje vypnuty.

Propojení jednotlivých přístrojů provedeme kabely, které jsou dodávány spolu s počítačem. Uspořádání pracoviště s počítačem ONDRA a způsob propojení jednotlivých přístrojů je uveden na obrázku na str. 7.

Postup při propojování jednotlivých přístrojů:

a/ Napájecí zdroj - počítač ONDRA: konektorovou zástrčku od napájecího zdroje vsuneme do prostřední konektorové zásuvky počítače. Zástrčku zajistíme našroubováním převlečné matice.

Uspořádání pracoviště s počítačem ONDRA:



b/ Počítač - televizor: použijeme kabel, který má na obou koncích tříkolíkové zástrčky:

- jednu zástrčku vsuneme do pravé konektorové zásuvky počítače;
- druhou zástrčku vsuneme do h o r n í konektorové zásuvky na zadní straně televizoru /pozor - vsunutím zástrčky do spodní zásuvky televizoru počítač p o š k o d í m e/;
- páčku přepínače na zadní straně televizoru přepneme do polohy "nahoru" /z televizního signálu přepneme na videosignál/;
- zapneme televizor.

c/ Počítač - magnetofon:

- sedmikolíkovou zástrčku zbývajícího kabelu vsuneme do levé konektorové zásuvky počítače;
- pětikolíkovou zástrčku tohoto kabelu vsuneme do konektorové zásuvky magnetofonu, která je určena pro nahrávání a přehrávání;
- magnetofon zapneme.

Počítač zapneme přepnutím páčky napájecího zdroje do polohy "nahoru". Na obrazovce se objeví nápis:

Ondra V3,86 nebo **Ondra V.5**

/u první výrobní serie/ /u typu SPO 186/

Poznámka: pro vyšší názornost odlišujeme znaky nebo text, který se objeví na obrazovce, obtažením jednotlivých znaků.

Dále se na obrazovce objeví hlášení vnitřního systému počítače:

•_

Znak "tečka" signalizuje připravenost vnitřního systému počítače přijímat některé naše povely.

Znak "_" se nazývá kurzor a ukazuje na místo, na němž by se zobrazilo písmeno, číslice nebo jiný znak, který bychom vložili do počítače.

Ovládacími prvky televizoru nastavíme příjemný jas i kontrast obrazu a snížíme hlasitost, aby nás nerušil šum z reproduktoru.

1.2 Zavedení překladače BASIC do počítače

Paměťové obvody v počítači ONDRA zatím nedovolují, aby v nich byl trvale uložen překladač /interpret/ jazyka BASIC. Proto musíme BASIC do počítače zavést po každém zapnutí počítače.

Činnost počítače určujeme stisknutím příslušných tlačítek na klávesnici. Počítač však "sám od sebe nepozná", že vkládání určitého povelu, příkazu, početního výrazu apod. je již ukončeno a že počítač má "začít pracovat". Proto po ukončení vkládání musíme stisknout tlačítko označené symbolem ←┘. Tímto tlačítkem počítači signalizujeme, že vkládání je ukončeno a že má začít provádět naše příkazy a povely. V dalším textu budeme tlačítko se symbolem ←┘ označovat písmeny CR.

Písmena CR tedy znamenají: stiskni tlačítko se symbolem ←┘.

Překladač jazyka BASIC je nahrán na kazetě dodávané s počítačem. BASIC lze do počítače zavést dvěma způsoby: u novějšího typu počítače ONDRA /SPO 186/ i zkráceným způsobem, u počítače z předchozí výrobní serie jen tak, jak do počítače zavádíme všechny systémové programy.

Při zkráceném způsobu zavádění překladače BASIC se počítač nedotazuje na název zaváděného programu /souboru/. Do počítače se zavede ten program, který se přečte jako první a zavedený program se spustí. Kazetu proto vložíme do magnetofonu a přetočíme ji na začátek před BASIC.

O b r a z o v k a

znaky:	význam znaků:	naše činnost:
._	počítač je připraven:	stiskneme tlačítko s písmenem L /tedy "vložíme L"/
.L cteni - hotove?	je kazeta nastavena před BASIC?	kazeta je nastavena, proto vložíme CR /tedy ↵/
obrazovka ztemní	zavádění BASIC	na magnetofonu spustíme funkci "přehrávání".

Zavedení BASIC z kazety do počítače trvá asi 3 minuty. Správný průběh zavádění signalizuje pravidelné blikání obou světelných diod vlevo od klávesnice počítače.

Při normálním /nezkráceném/ způsobu zavádění BASIC do počítače je postup složitější:

Obrazovka:	význam znaků:	naše činnost:
._	připravenost počítače	vložíme písmena KL
K_L	povel byl správně převzat	
nazev	jaký je název zaváděného programu?	vložíme písmena BASIC vložíme CR

čtení - hotovo?	máme kazetu nastavenu před nahrávku BASIC?	ano, máme: vložíme CR spustíme magnetofon;
obrazovka ztemní -	zavádění BASIC ukončení zavádění; připravenost počítače	čekáme; vložíme písmeno B /přechod do BASIC/.

Po zavedení interpetu BASIC do počítače se na obrazovce objeví hlášení:

```
BASIC - EXP V5.Ø/G
(C) 1986 TESLA DIZ
READY
:
```

Zavedený program se hlásí svým jménem /BASIC/ a označením své verze /EXP V5.Ø/G/.
Hlášení **READY** /čti "redy"/ znamená připravenost BASIC.
Dvojtečka oznamuje, že BASIC je připraven přijímat povely a příkazy vkládané klávesnicí.

Poznámka: u počítače ONDRA se jednotlivé programy od sebe rozlišují svými názvy. Název programu může obsahovat až 11 znaků. Název lze při zavádění programu do počítače zadat i zkráceně, tedy třeba jen jeho prvním písmenem.

1.3 Chyby při zavádění programů z kazety do počítače

Jestliže po vložení povelu L /nebo po stisknutí CR/ a po spuštění magnetofonu obě světelné diody shodně neblikají, BASIC /nebo jiný program/ se do počítače nezavádí. Proto

a/ špičatým nástrojem stiskneme červené tlačítko na levém boku počítače; tím přerušíme činnost počítače a opět vyvoláme základní hlášení ._

b/ kazetu přetočíme zpět před nahrávku BASIC /nebo jiného programu/ a pokud to náš magnetofon umožňuje, nastavíme na něm vyšší

úroveň /"hlasitost"/ výstupního signálu, nebo alespoň jinou úroveň regulátoru výšek a hloubek tónů;

c/ opakovaně se pokusíme BASIC /nebo jiný program/ do počítače zavést způsobem popsaným v předchozí kapitole 1.2.

Pokud tato opatření nepomohou, použijeme jiný magnetofon s lépe nastavenou snímací hlavou, nebo přeneseme náš magnetofon k jinému uživateli počítače ONDRA a požádáme ho, aby nám BASIC přehrál přímo ze svého počítače na kazetu v našem magnetofonu.

Někdy se stane, že světelné diody sice začnou zprvu shodně blikat, ale po chvíli toto blikání přestane a na obrazovce se objeví chybové hlášení, například

15

Chyba cteni - Znova /N

Počítač tak signalizuje, že správně nepřečetl určitý blok záznamu BASIC nebo jiného programu. V našem případě správně nepřečetl blok s pořadovým číslem 15. Chybu odstraníme tak, že

- do počítače vložíme CR

  zobrazí se: **vrat zpet cteni - hotovo?**

- kazetu v magnetofonu přetočíme o kousek zpět;
- do počítače vložíme CR a na magnetofonu opět spustíme funkci "přehrávání"; pokoušíme se tedy znova do počítače zavést stejný blok programu.

Jestliže se opět zobrazí stejné chybové hlášení, zastavíme chod magnetofonu, vyjmeme z něho kazetu s programem a páskovou dráhu se pokusíme vyčistit specielní kazetou /například EMGETON HC-1/.

Po průchodu čistící kazety do magnetofonu opět vložíme kazetu s programem. Červeným tlačítkem na boku počítače přerušíme jeho činnost a opakovaně zkusíme zavést celý program do počítače.

Pokud se nám ani nyní nepodaří zavést celý program do počítače, je vadná část pásky v kazetě. Musíme proto použít jinou nahrávku tohoto programu.

Každý program /i BASIC/ mějme na kazetě nahraný dvakrát nebo i třikrát. Vadnou kazetu pro další práci nepoužívejme!

Upozornění: pokud bychom po náhodném stisknutí tlačítka B v průběhu zavádění BASIC do počítače obdrželi hlášení, že "BASIC je READY", není to pravda. Do počítače se zavedla jen část BASIC. Počítač proto později nebude "rozumět" některým našim povelům a příkazům, nebo bude hlásit, že ve správném programu "jsou chyby". Zavedení BASIC do počítače je proto třeba od začátku opakovat!

Při oblečení do výrobků ze syntetických materiálů může naše tělo získat i velmi vysoký elektrostatický náboj. Po dotyku s kovovou skříňkou počítače nebo s obrazovkou televizoru může tento náboj zčásti nebo i úplně vymazat programy v počítači. Pak nezbývá, než se převléci, nebo používat gumové rukavice.

1.4 Operační systém počítače ONDRA

Každý počítač je již od výrobce vybaven základním operačním systémem. Bez něho bychom do počítače nedokázali ani zavést BASIC. Písmena KL , která jsme do počítače vložili při zavádění BASIC /nebo L při zkráceném zavádění/ patří mezi základní instrukce operačního systému počítače ONDRA. Operační systém je tvořen dvěma základními systémovými programy:

a/ systémový program MONITOR přijímá znaky vkládané klávesnicí, vysílá znaky na obrazovku, umožňuje spouštění programů, apod.,

b/ systémový program MIKOS umožňuje zavádění programů a dat /čísel/ z magnetofonu do počítače a nahrávat je z počítače na kazetu.

Pro využití všech vlastností MIKOSu je nejvhodnější magnetofon s dálkovým ovládáním chodu motoru. Při použití jiného typu magneto-

fonu musíme sami ovládat jeho chod příslušnými tlačítky. Je třeba si uvědomit, že n e j d ř í v e musíme spustit ten přístroj, do něhož údaje vstupují a teprve pak přístroj, z něhož údaje vystupují. V opačném případě by se část údajů nezavedla nebo nezaznamenala na kazetu.

1.5 Klíčová slova - povely a příkazy

Činnost počítače je po zavedení BASIC řízena klíčovými slovy - povely a příkazy. Povely řídí překladač jazyka BASIC, ovládají a sledují chod programu, atd. Nemají však žádný vztah k řešenému úkolu, neboť povelem RUN /čti:"ran"/ spustíme libovolný program, povelem LIST /"list"/ vypíšeme libovolný program na obrazovku.

Příkazy zajišťují konkrétní činnosti související s určitým úkolem, například provedení výpočtů, zobrazení textů nebo výsledků na obrazovce, vkládání vstupních údajů, apod.

Vlastní činnost počítače neprobíhá přímo v BASIC, ale ve vnitřním kódu počítače, do něhož se z BASIC překládají jednotlivé povely a příkazy. Počítač proto "rozumí" našim povelů a příkazům jen tehdy, jestliže je do počítače vložíme přesně tak, jak určují pravidla programovacího jazyka BASIC-EXP V5.Ø/G. Jak by si počítač asi poradil s naším příkazem

"nakresli digitální holinky" ?

Počítač nepozná, že jsme omylem zaměnili jediné písmeno v příkazu

"nakresli digitální hodinky"

a bude hlásit chybu.

1.6 Základní režimy práce počítače

Jednotlivá klíčová slova /povely a příkazy/ do počítače vkládáme klávesnicí. Na obrazovce se tyto povely a příkazy objevují v řádcích. Vkládání klíčových slov musíme ukončit vložením CR /tedy

stisknutím tlačítka ⏎ /. Jinak by počítač "nepoznal", že vkládání je ukončeno a že má začít pracovat:

- buď okamžitě vykonat obsah d i a l o g o v é h o řádku,
- nebo p r o g r a m o v ý řádek uložit do paměti počítače.

V dialogovém režimu jednotlivé řádky nečíslujeme. Po vložení CR se ihned splní všechny povely a příkazy vložené do dialogového řádku. Činnost počítače v dialogovém režimu proto můžeme přirovnat k činnosti kalkulátoru /"kalkulačky"/: po vložení určitých čísel a stanovení způsobu výpočtu se ihned zobrazí výsledek.

V programovém režimu do počítače vkládáme jako první v pořadí číslo programového řádku. U počítače ONDRA můžeme pro číslování programových řádků používat libovolná celá čísla od nuly do 64999. Programový řádek odešleme do paměti počítače pouhým vložením CR .

1.7 Ovládání počítače

Obsluha klávesnice u počítače ONDRA je poněkud složitější než u některých jiných typů osobních počítačů. Okamžitou funkci jednotlivých tlačítek nejsnáze určíme podle svitu kontrolních diod po levé straně klávesnice:

s v í t í :	obě diody	horní dioda	dolní dioda	žádná dioda
velká PÍSMENA:	ano			
malá písmena:		ano		
číslice 1 až Ø:				nesvítí
znaky ve druhé a třetí řadě:				nesvítí
všechny znaky:			ano	

Funkci jednotlivých tlačítek určuje předchozí stisknutí specielních přepínacích tlačítek.

U počítače ONDRA SPO 186

- trvalé přepnutí na číslice a znaky ve druhé a třetí řadě: stisknutím přepínacího tlačítka se symbolem "Ø-9";
- trvalé přepnutí na písmena a znaky: tlačítkem s trojúhelníčky nebo tlačítkem se čtverečky; funkci těchto tlačítek můžeme ovlivnit i tlačítkem "Ø-9";
- tlačítko ČS je určeno pro přepnutí na znaky české abecedy, pokud jsme soubor těchto znaků předem zavedli z kazety do počítače.

U původní verze počítače ONDRA provede tlačítko ČS trvalé přepnutí na číslice a na znaky ve druhé a ve třetí řadě. Tlačítko Ø-9 provede toto přepnutí jen po dobu jeho stisknutí. Na rozdíl od novějšího provedení počítače ONDRA se u původní verze používají jen velká písmena.

Tlačítko CTRL /čti: "kontrol"/ mění význam písmen na řídící znaky, například stisknutím tlačítek

CTRL C se přeruší chod počítače,

CTRL D se při opravě programového řádku vymaže 1 znak, atd.

Čtyři tlačítka se šipkami v pravém spodním rohu klávesnice slouží k ovládání abecedně-číselného nebo grafického kurzoru.

Převzetí vloženého znaku počítačem je signalizováno krátkým zvukovým signálem /pípnutím/. Klávesnice je vybavena i programovým opakováním stejného znaku při delším stisknutí klávesy.

1.8 Zobrazování v abecedně-číselném režimu

Při běžných výpočtech se využívá základní, tedy abecedně číselný režim zobrazování. Obrazovka je v tomto režimu členěna do 24 řádků. Do každého řádku lze vložit až 40 znaků /sloupců/, takže na obrazovku lze vložit až 960 znaků. Musíme však rozlišovat mezi řádkem na obrazovce a programovým řádkem: zatím co do obrazového

řádku se vejde jen 40 znaků, do programového řádku můžeme vložit až 132 znaků /vč. mezer/. To znamená, že plný programový řádek zabírá až 3,3 obrazových řádků. Přechod mezi jednotlivými obrazovými řádky je automatický - kurzor se po naplnění obrazového řádku automaticky přesune na začátek dalšího obrazového řádku.

Místo, na němž se zobrazí další znak, ukazuje abecedně číselný kurzor "_". Při vložení chybného znaku můžeme tuto chybu opravit pouhým stisknutím tlačítka ←—, jímž chybný znak vymažeme. Takto jednoduše lze opravy provádět jen tehdy, když chybu zjistíme ještě před vložením CR .

U p o z o r n ě n í: počítač přísně rozlišuje písmeno "O" a číslici "nula". Na obrazovce je i odlišným způsobem zobrazuje.

Číslici "nula" proto při práci s počítačem zapisujeme takto: Ø , například 1Ø3, 2ØØ5. Do počítače vkládáme číslici "nula" tlačítkem se znakem "Ø", které je úplně vpravo nahoře.

Kontrolní otázka: bylo číslo 705 do počítače vloženo takto správně?

705 CR

Číslo nebylo do počítače vloženo správně. Počítač vložené znaky vyhodnotí jako

číslici 7 písmeno O číslici 5 .

Správný způsob vložení tohoto čísla do počítače: 7Ø5 CR

1.9 Vkládání klíčových slov zkráceným způsobem

U počítače ONDRA je možno některé nejčastěji používané povely a příkazy vkládat zkráceným způsobem. Stačí jen stisknout tlačítko se šipkou vpravo a pak do počítače jen vložit příslušné písmeno, například —→ I CR vloží se klíčové slovo INPUT

—→ D CR vloží se klíčové slovo DATA , atd.

U některých klíčových slov však není vkládané písmeno shodné s prvním písmenem klíčového slova, například J znamená RUN.

Klíčové slovo PRINT můžeme do počítače zkráceně vložit tlačítkem se znakem "?". Tedy místo PRINT A CR stačí vložit pouze ? A CR do paměti se neuloží znak "?", ale klíčové slovo PRINT.

Vkládání klíčových slov zkráceným způsobem nejen zrychluje práci s počítačem, ale snižuje i možnost vzniku chyb v klíčových slovech. Přehled klíčových slov, která lze do počítače vkládat zkráceným způsobem, je uveden v příloze č. 1.

Pro úspěšnou práci s počítačem stačí jen:
- počítače se nebát, neboť zničit počítač není snadné,
- získat zručnost při ovládání klávesnice,
- umět přesně číst a dokázat se "prokousat" touto příručkou,
- všechny uváděné příklady si prakticky ověřovat.

2. Jednoduché výpočty v dialogovém režimu

V dialogovém režimu řádky vkládané do počítače nečíslujeme. Vložené povely a příkazy se vykonají ihned po vložení CR , tedy po stisknutí tlačítka ←┘ . Z kontrolních důvodů jsou výsledky jednotlivých příkladů uváděny v závorce.

Příklad 1: na počítači je třeba sečíst čísla 20 a 18.
Řešení: musíme postupovat jinak než při výpočtu na kalkulátoru. Do počítače je třeba postupně vložit:

PRINT 2Ø + 18 CR

co se má provést s výsledkem: vlastní výpočet: ukončení řádku.

Počítač nejdříve provede vše, co následuje za příkazem PRINT /čti "print"/ a pak vykoná povel PRINT.
Povel PRINT znamená: "zobraz na dalším řádku na obrazovce".
Do počítače vložíme přesně to, co je zapsáno v dalším řádku.

PRINT 2Ø + 18 CR /CR znamená: stiskni tlačítko ⏎ /.

Po vložení CR se ihned zobrazí výsledek, tedy číslo 28. Rovnítko "=" v těchto typech příkladů n e p o u ž í v á m e.

U některých druhů výpočtů používáme na počítači jiná znaménka /operátory/, než na jaká jsme zvyklí z matematiky.

Příklad 2: k součinu čísel 2 x 3 přičtěme číslo 5.

PRINT 2*3 + 5 CR /W=11/

Pro násobení používáme operátor "*", tedy znak, který je nad písmenem Z .

Příklad 3: vypočítejme hodnotu výrazu 2 + 3 + 2x7 .

PRINT 2 + 3 + 2*7 CR /W=19/

Výpočet se provede tak, jako kdyby byl uzavřen v těchto závorkách:

2 + 3 + (2 x 7).

Příklad 4: vypočítejme součet čísel 10,5 + 0,3 + 31,723 .

Řešešní: při výpočtech s desetinnými čísly do počítače vkládáme místo desetinné čárky desetinnou t e č k u. Samotnou nulu u čísel menších než 1 není třeba do počítače vkládat.

PRINT 1Ø.5 + .3 + 31.723 CR /W=42.523/

Vkládání klíčového slova PRINT lze nahradit vložením znaku ?

? 1Ø.5 + .3 + 31.723 CR

Příklad 5: vypočítejme podíl čísel 129,48 : 15,6 .

Pro dělení používáme operátor "šikmé lomítko":

? 129.48 / 15.6 CR /W=8.3/

Příklad 6: vypočítejme druhou mocninu čísla 3, tedy 3^2.

Pro výpočet můžeme použít buď násobení, tedy

? 3*3 CR /W=9/

nebo operátor pro výpočet mocniny, který je u písmene F :

PRINT 3↑2 CR /W=9/

Příklad 7: vypočítejme třetí mocninu čísla 5, tedy 5^3.

? 5↑3 CR /W=125/

Cvičení: byly do počítače správně vloženy tyto výrazy?

PRIVT 28,5 + 14.7 CR ? 2 x 3 CR

158.1 x 3 CR PRINT 8 : 4 CR

PRIN 1Ø3 + 10.7 CR ? 2 + 3) x 4 CR

/Žádný výraz nebyl do počítače vložen správně. V některých výrazech je i několik chyb./

2.1 Identifikátory, konstanty a proměnné

Při některých výpočtech uvítáme, jestliže si počítač určité číslo "zapamatuje"; například často používané Ludolfovo číslo 3,14 /číslo "pí"/. Vložme proto číslo "pí" do paměti počítače.

Na rozdíl od obyčejného kalkulátoru s jednou pamětí však musíme u počítače určit, do které z mnoha pamětí počítače má být číslo "pí" uloženo.

Jednotlivé paměti v počítači označujeme jejich názvy, které si dokonce můžeme sami vymýšlet. Těmto názvům říkáme "identifikátory". Jednotlivé identifikátory /názvy pamětí/ mohou mít až 8 znaků. Prvním znakem vždy musí být p í s m e n o, dalšími znaky mohou být písmena nebo číslice. Avšak pozor:

- identifikátory nesmějí obsahovat prvky klíčových slov; identifikátorem proto nemohou být např IF, DO, GO, TO apod.;
- počítač u každého identifikátoru posuzuje jen dva první znaky; proto identifikátory KRUH a KRUPICE bude považovat za shodné, neboť je vždy vyhodnotí jako identifikátor KR .

Identifikátory budeme vytvářet z velkých písmen a číslic tak, aby nám připomínaly svůj obsah; pro číslo "pí" použijeme označení PI.

Konkrétní hodnoty se do jednotlivých pamětí počítače vkládají p ř i ř a z o v a c í m příkazem. Přiřazovací příkaz používá symbol /tlačítko/ "=".

PI	=	3.14
identifikátor;	přiřazovací příkaz;	co se má přiřadit /vložit/ do paměti označené identifikátorem PI.

Příklad 8: hodnotu čísla "pí" = 3,14 vložme do paměti označené identifikátorem PI.

PI = 3.14 CR přiřazovací příkaz - vložení čísla 3.14 do PI.

Poznámka: pro zvýšení přehlednosti a srozumitelnosti bude u většiny dalších příkladů uváděn na pravé polovině stránky stručný význam jednotlivých dialogových nebo programových řádků.

Příklad 9: zobrazme číslo /hodnotu/ uložené v paměti PI.

PRINT PI CR zobrazení hodnoty uložené v paměti označené identifikátorem PI .

Obsah určité paměti může být během celého výpočtu stále stejný. Může v ní být například stále uloženo jen číslo 3.14 - pak hovoříme o paměti k o n s t a n t . Nebo do jiných pamětí postupně vkládáme různá čísla, která mezi sebou násobíme. Protože se obsah těchto pamětí stále mění, hovoříme o pamětech p r o m ě n n ý c h.

Příklad 10: vypočítejme obvod kruhu o poloměru r=5.
Úvaha: pro výpočet použijeme vzorec $O = 2\pi r$. Číslo 3.14 jsme však do paměti počítače vložili již v příkladu č. 8. Proto nyní stačí číslo 3.14 vyvolat z paměti označené identifikátorem PI.
Řešení: PRINT 2*PI*5 výpočet obvodu kruhu o poloměru 5.

Číslo 3.14 zůstane v paměti označené identifikátorem PI tak dlouho, dokud do této paměti nevložíme jiné číslo, dokud obsah všech pamětí nevymažeme povelem RUN nebo dokud nevypneme počítač.

Příklad 11: vypočítejme obvody a plochy kruhů s poloměry r_1=5, r_2=7 a r_3= 10. Výsledky vkládejme do proměnných

O1, O2 a O3 při výpočtu obvodů jednotlivých kruhů,

S1, S2 a S3 při výpočtu plochy jednotlivých kruhů.

Řešení rozdělíme do čtyř částí:

a/ Číslo 3.14 je již uloženo v paměti PI - stačí je jen vyvolat.

b/ Jednotlivé poloměry vložíme do pamětí označených identifikátory R1, R2 a R3:

R1 = 5 CR vložení poloměru r_1 do paměti R1,

R2 = 7 CR vložení poloměru r_2 do paměti R2,

R3 = 1Ø CR vložení poloměru r_3 do paměti R3.

c/ Jednotlivé výpočty provedeme s čísly uloženými v příslušných pamětech počítače. Místo konkrétních čísel použijeme příslušné identifikátory, tzn. že místo čísla 3.14 použijeme identifikátor PI, místo čísla 5 použijeme identifikátor R1 atd.

O1 = 2*PI*R1 CR výpočet obvodu kruhu s poloměrem 5,

O2 = 2*PI*R2 CR výpočet obvodu kruhu s poloměrem 7,

O3 = 2*PI*R3 CR výpočet obvodu kruhu s poloměrem 1Ø,

S1 = PI*R1↑2 CR výpočet plochy kruhu s poloměrem 5,

S2 = PI*R2↑2 CR výpočet plochy kruhu s poloměrem 7,

S3 = PI*R3↑2 CR výpočet plochy kruhu s poloměrem 1Ø.

d/ Jednotlivé výsledky si z příslušných proměnných vyvoláme příkazem PRINT, nebo zkráceně "?":

PRINT O1 CR ? O2 CR ? O3 CR obvody kruhů,

PRINT S1 CR ? S2 CR ? S3 CR plochy kruhů.

Poznámka: s údaji uloženými v jednotlivých pamětech počítače můžeme provádět další různé výpočty. Například zjišťovat, o kolik % je plocha kruhu o poloměru r_3 větší než plocha kruhu o poloměru r_1, ap.

2.2 Standardní číselné funkce

Standardní číselné funkce umožňují zjišťovat vlastnosti čísel nebo s nimi jednoduše provádět některé výpočty. Jsou výrobcem počítače ONDRA předem naprogramovány, takže je stačí jen správným způsobem z počítače vyvolat.

Upozornění: argument funkce, tedy číslo, s nímž pracujeme, musí být uzavřen v kulaté závorce. Mezi jménem funkce a závorkou, v níž je argument, n e s m í být žádná mezera.

Jméno funkce:		význam a příklad funkce:
ABS(N)	/čti "abs"/	funkcí se zjistí absolutní hodnota čísla N:
PRINT ABS(-5.3) CR		zobrazí se: 5.3
SQR(N)	/"esqér"/	přímý výpočet druhé odmocniny z čísla N:
PRINT SQR(144) CR		zobrazí se: 12
INT(N)	/"intýžr"/	zobrazí pouze celočíselnou část z čísla N:
PRINT INT(123.45) CR		zobrazí se: 123 /tedy bez des. čísel/
LOG(N)		výsledkem je p ř i r o z e n ý logaritmus kladného čísla N /v matematice: ln N/.
SGN(N)	/"sajn"/	zjišťuje druh čísla N: jestliže číslo N je kladné, je výsledkem 1, jestliže číslo N je záporné, je výsledkem -1, jestliže N je rovno Ø, je výsledkem číslo Ø
PRINT SGN(-5.5) CR		zobrazí se: -1 /číslo N je záporné/.
EXP(N)	/"exp"/	výsledkem exponenciální funkce je přirozené číslo e^N /číslo $e^{na\ Ntou}$/:
PRINT EXP(1Ø) CR		zobrazí se: 22Ø26.5
RND(N)	/"erende"/	generátor pseudonáhodných čísel od Ø do Ø.99..

PRINT RND(N) CR zobrazí se například Ø.Ø183 .

FRE(Ø) /"frí"/ zjištění délky volné paměti v bytech /čti
PRINT FRE(Ø) CR "bajtech"/.

2.3 Řetězcové proměnné

Do paměti počítače můžeme vkládat nejen čísla, ale i písmena a různé grafické znaky - řetězce. Musíme však předem počítači "oznámit", že do počítače nevložíme číslo, ale řetězec znaků. Příslušný identifikátor proto doplníme znakem $ /čti "string"/ a řetězec znaků vložíme do uvozovek.

Příklad 12: do řetězcové proměnné /například do P$/ je třeba vložit řetězec znaků, který vyjadřuje název počítače ONDRA.
Řešení: do počítače vložíme:

P$ = "ONDRA" CR vložení řetězce ONDRA do P$

Příklad 13: dále je třeba vložit do proměnné R$ řetězec "počítač" a do proměnné S$ řetězec "školní":

R$ = "POCITAC" CR S$ = "SKOLNI" CR

Příklad 14: na obrazovce je třeba zobrazit text "SKOLNI POCITAC ONDRA".
Úvaha: všechny potřebné řetězce jsme již v minulém příkladu vložili do řetězcových proměnných P$, R$ a S$. Stačí proto jen opakovaně příkazem PRINT vyvolat příslušný řetězec z paměti počítače:

PRINT S$ CR	zobrazí se:	SKOLNI
PRINT R$ CR		POCITAC
PRINT P$ CR		ONDRA

Příklad 14: do číselné proměnné C vložme číslo 27. Do řetězcové proměnné C$ vložme řetězec složený ze znaků 27:

C = 27 CR C$ = "27" CR

S řetězcem "27" nelze pracovat stejně jako s číslem 27. Je mezi

nimi asi takový rozdíl jako mezi výsledkem násobení 3 x 9 a číslem "27", které udává číslo domu v nějaké ulici.

Řetězce lze slučovat pomocí operátoru "+". Musíme však do jednotlivých řetězců vložit na správné místo i znak "mezera".

Příklad 15: zobrazme předchozí řetězce P$, R$ a S$ v jednom obrazovém řádku.

Řešení: jednotlivé řetězce sloučíme operátorem "+"

```
PRINT S$ + R$ + P$  CR     zobrazí se: SKOLNIPOCITACONDRA
```

Příklad 15: navrhněme způsob, jímž správně rozdělíme předchozí řetězec.

Řešení: nechceme-li dodatečně do jednotlivých řetězců vkládat mezery, tedy "SKOLNI " "POCITAC " ,
vytvoříme nový pomocný řetězec M$, jehož obsahem bude znak "mezera" a všechny řetězce sloučíme operátorem "+":

```
M$ = " "  CR                          vytvoření pomocného řetězce,
PRINT S$ + M$ + R$ + M$ + P$  CR      zobrazí se:
                                      SKOLNI POCITAC ONDRA
```

Příklad 16: vytvořme nový řetězec Q$, který bude obsahovat text SKOLNI POCITAC ONDRA .

```
Q$ = S$ + M$ + R$ + M$ + P$  CR       vytvoření nového řetězce.
```

2.4 Standardní řetězcové funkce

Standardní řetězcové funkce umožňují snadnou manipulaci s nejrůznějšími řetězci. Například podle koncovky v příjmení určit, zda se jedná o příjmení hocha či dívky; vytvářet abecední seznamy atd. Příslušný řetězec musí být vložen do kulatých závorek a mezi jménem funkce a levou závorkou n e s m í být mezera.

Význam jednotlivých řetězcových funkcí si opět objasníme na příkladech. Vložme si proto do proměnné A$ řetězec "ONDRA":

```
A$ = "ONDRA"  CR
```

Jméno funkce, příklad		význam funkce
LEN(A$)	/"len"/	výsledkem je počet znaků v proměnné A$:
PRINT LEN(A$)	CR	zobrazí se: 5 /v textu ONDRA je 5 znaků/.
LEFT$(A$,n)	/"left"/	výsledkem je nový řetězec vytvořený z řetězce A$ odříznutím n znaků zleva:
B$ = LEFT$(A$,2)	CR	odříznou se 2 znaky zleva a vloží se do B$
PRINT B$	CR	zobrazí se: ON /2 znaky z ONDRA zleva/.
RIGHT$(A$,n)	/"rajt"/	nový řetězec vznikne odříznutím n znaků
B$ = RIGHT$(A$,2)	CR	zprava: 2 znaky zprava se vloží do B$,
PRINT B$	CR	zobrazí se RA /2 znaky z ONDRA zprava/.
MID$(A$,n)	/"mid"/	do nového řetězce se přenese zbytek řetězce A$ od n-tého znaku:
B$ = MID$(A$,2)	CR	do B$ se přenesou znaky od 2 znaku z A$,
PRINT B$	CR	zobrazí se: NDRA /znaky od 2 pozice/.
MID$(A$, n_1, n_2)		do nového řetězce se přenese od pozice n_1 celkem n_2 znaků směrem doprava:
B$ = MID$(A$,2,3)	CR	přenesou se 3 znaky od 2 pozice zleva,
PRINT B$	CR	zobrazí se: NDR .

Všechny znaky /i písmena a číslice/, které jsou na klávesnici, jsou zakódovány do čísel soustavy ASCII /čti "aski"/. Jednotlivé kódy jsou uvedeny v příloze k publikaci č. 4 "BASIC" dodávané výrobcem počítače ONDRA. Pro převod mezi tvarem znaku na tlačítku klávesnice a kódem soustavy ASCII se používají funkce:

ASC(A$)	/"ask"/	převede znak ze stisknutého tlačítka do kódu soustavy ASCII,
CHR$(N)	/"chr"/	převede číselný kód N na znak, který odpovídá příslušnému tlačítku na klávesnici.

Někdy bývá třeba převést znaky uložené v řetězcové proměnné na číslo v číselné proměnné a naopak. Do počítače vložme A$="25" CR

VAL(A$)	/"val"/	číslo se převede z řetězcové do číselné proměnné:
B = VAL(A$)	CR	převod mezi proměnnými různých typů,
PRINT B	CR	zobrazí se číslo 25 ;
STR$(N)		číslo uložené v číselné proměnné N se pře-
A$ = STR$(N)	CR	vede na znaky v řetězcové proměnné A$.

2.5 Trigonometrické funkce

Trigonometrické výpočty lze provádět s údaji vkládanými ve stupních nebo vkládanými v radiánech. Druh výpočtu se nastavuje příkazem /přepínačem/:

DEG ON	CR	zadávání úhlů ve stupních,
DEG OFF	CR	zadávání úhlů v radiánech.
SIN(N)		trigonometrická funkce "sinus", například:
DEG ON	CR	přepnutí na údaje ve stupních,
PRINT SIN(3Ø)	CR	zobrazí se: .5 /sin 30 = 0,5/;
DEG OFF	CR	přepnutí na radiány;
PRINT SIN(3Ø)	CR	zobrazí se: -.988Ø32 ,
COS(N)		funkce "cosinus",
TAN(N)		funkce "tangens",
ATN(N)		funkce "arcus-tangens".

2.6 "Kreslení" přímo na obrazovku

Přímé "kreslení" na obrazovku předpokládá, že v počítači je zaveden překladač BASIC-EXP V5.G. Kreslíme příkazem DRAW INPUT.

CLS	CR	vymazání abecedně-číselných údajů z obrazovky.

DRAW INPUT A$ CR vložení kombinovaného grafického příkazu.

Počátek grafického zobrazování je v levém s p o d n í m rohu obrazovky. Nalezneme zde světlý bod - g r a f i c k ý kurzor. Tlačítky se šipkami ← ↑ ↓ → pro ovládání kurzoru přemístíme tento bod ke středu obrazovky a stiskneme tlačítko "V". Vytvoří se čára, která spojí předchozí polohu grafického kurzoru s jeho současnou polohou. Klávesnice m u s í být přepnuta na velká písmena!

Jestliže čáru nechceme vytvořit, stiskneme tlačítko "S". Čára se nevytvoří, ale poloha tohoto bodu se uloží v paměti.

Příklad 17: s využitím kombinovaného grafického příkazu nakresleme "domeček". Počátek zobrazování určíme příkazem MOVE /čti "mův"/.
Řešení: jestliže máme obrazovku zaplněnu jinými znaky, vymažeme ji.

CLS: MOVE Ø,Ø CR vymazání obrazovky, počátek zobrazování;
DRAW INPUT CR do počítače vložíme uvedená písmena, stiskneme ↵ a nalezneme grafický kurzor.

Domeček nakreslíme tak, že tlačítky se šipkami pro ovládání grafického kurzoru přesuneme tento kurzor na požadované místo a pak stiskneme tlačítko "V" nebo tlašítko "S".

Poznámka: souřadnice jednotlivých bodů, v nichž jsme stiskli tlačítko "V" nebo tlačítko "S", se ukládají do řetězcové proměnné A$. Pokud bude mít náš domeček příliš mnoho rohů, může se stát, že kreslení se přeruší. Kapacita řetězcové proměnné A$ nebude postačovat na jeho dokreslení. Pak je třeba kombinovaný grafický příkaz doplnit o další řetězcové proměnné, např. o B$ a C$:

DRAW INPUT A$, B$, C$ CR

Pro případné vymazání obrazovky používáme příkaz CLS. Jestliže nám vadí vypsání příkazu DRAW INPUT na obrazovce, vložíme oba příkazy do jednoho řádku; musíme je však od sebe oddělit dvojtečkou:

CLS : DRAW INPUT A$, B$, C$ CR

3. Programový režim počítače

Těžiště využívání počítačů je v činnostech, které probíhají podle předem stanoveného programu. Program je přesný popis všech činností, které má počítač vykonat.

3.1 Struktura programu

Každý program lze rozdělit do těchto základních částí:

.1 Určení hodnot -proměnných a konstant a jejich typu /číselné, řetězcové/, s nimiž bude program pracovat.

.2 Určení způsobu, jakým počítač bude s jednotlivými proměnnými a konstantami pracovat, aby bylo dosaženo požadovaného výsledku.

.3 Určení způsobu, jak se má naložit s výsledkem /zobrazit jej, vytisknout na tiskárně apod./.

.4 Určení, co má počítač dělat po zobrazení nebo vytisknutí výsledku /přejít na začátek programu a výpočet opakovat s jinými čísly, přechod k jinému programu, atd./.

Pro sestavení správného programu nestačí jen znát programovací jazyk tak, jak je uváděn u učebnicích programování. Je třeba znát i jeho verzi pro konkrétní počítač, například BASIC-EXP V5.Ø/G pro počítač ONDRA, neboť

- u různých počítačů může mít určité klíčové slovo jiné významy,
- pro vykonání určité činnosti se u různých počítačů mohou používat různá klíčová slova, popř. tato slova vůbec neexistují.

Někdy však nestačí ani dobrá znalost správné verze programovacího jazyka. Například když je třeba vypočítat průměr lana, které ponese lanovku, nebo navrhnout motor pro nový sportovní automobil.

> Jestliže víme, jak určitý program vypočítáme "ručně na papíru", určitě u každého počítače najdeme klíčová slova, z nichž sestavíme správný program.

Pokud však neznáme ani postup /algoritmus/ řešení, pak ani nevíme, jaké příkazy je třeba použít a jak program vůbec sestavit.

Znalost programovacího jazyka i znalost postupu /algoritmu/ řešení se tedy doplňují.

3.2 Začínáme programovat

Se základy programování se budeme seznamovat stavebnicově. Tedy tak, jak budeme jednotlivá klíčová slova potřebovat k řešení určitých úkolů.

Příklad 18: je třeba sestavit program pro sečtení čísel 28, 30, 42. V dialogovém režimu bychom do počítače vložili:

PRINT 28 + 3Ø + 42 CR

Po vložení CR by se ihned zobrazil výsledek 1ØØ.

V programovém režimu musí být každý programový řádek očíslován. Program se pak provádí od řádku s nejnižším pořadovým číslem k řádku s vyšším pořadovým číslem.

Vložme proto do počítače číslo /například 1Ø/ programového řádku a za ně to, co bychom do počítače vložili v dialogovém režimu:

1Ø PRINT 28 + 3Ø + 42 CR

Po vložení CR se výsledek nezobrazil a dvojtečka s kurzorem se přesunula na další řádek na obrazovce. To znamená, že počítač celý programový řádek 1Ø uložil ve své paměti.

Náš první program si spustíme povelem RUN /čti "ran"/. Do počítače vložíme písmena R U N a vložíme CR /tedy stiskneme ←┘/. Na obrazovce se objeví výsledek výpočtu podle programu v řádku 1Ø, tedy číslo 1ØØ.

Protože výsledek výpočtu se již zobrazil, počítač čeká, co od něho budeme dále požadovat. Spusťme si opět program povelem RUN. Znova se zobrazí výsledek 1ØØ, neboť se sčítají stále stejná čísla - programové konstanty 28, 3Ø a 42.

Příklad 19: sestavme program pro násobení čísel 5 x 25.

Řešení: budeme postupovet obdobně jako u programu pro sečítání.

1Ø PRINT 5*25 CR

Vložením řádku 1Ø s jiným obsahem jsme původní programový řádek 1Ø pro sečítání přepsali programem pro násobení. I tento nový program spustíme vložením povelu RUN CR .

3.3 Výpis programu z paměti počítače

Povelem LIST na obrazovce postupně zobrazíme všechny programové řádky, které jsou uloženy v paměti počítače.

Povel LIST používejme při každém zasednutí k zapnutému počítači. Zjistíme tak, zda v počítači již není uložen nějaký program. Jaké by asi byly výsledky našich výpočtů, kdyby se v počítači dva různé programy "smíchaly dohromady"?

LIST M CR	/"list"/	výpis programu na obrazovce začíná až od řádku s pořadovým číslem M:
LIST 25	CR	program se vypíše až od řádku 25.
LIST M,N	CR	zobrazí se celkem N programových řádků od řádku s pořadovým číslem M:
LIST 2Ø,2	CR	zobrazí se řádek 2Ø a jeden další programový řádek /tedy celkem 2 řádky/.

3.4 Spuštění vloženého programu

Nejčastěji program spouštíme povelem RUN CR . Povel RUN však způsobí, že se vynulují /vymaží/ všechny číselné i řetězcové proměnné, do nichž byla v předchozím průběhu výpočtu vložena nějaká hodnota! Program se spustí od programového řádku s nejnižším pořadovým číslem.

RUN M CR	program se spustí až od řádku s pořadovým číslem M. Všechny proměnné se rovněž vynulují /vymaží/.

Jestliže je třeba program spustit, ale proměnné se nesmějí vynulovat /vymazat/, použijeme povel GOTO /čti "goutů"/:

GOTO M CR — program se spustí až od řádku s číslem M:

GOTO 1ØØ CR — spuštění programu od řádku 1ØØ; proměnné se nevynulují.

3.5 Přerušení chodu programu nebo výpisu programu na obrazovku

Chod programu nebo výpis programu na obrazovku přerušíme současným stisknutím tlačítek "CTRL" a "C". Jestliže tato tlačítka stiskneme během chodu programu, program se přeruší, ale proměnné se nevynulují.

Jestliže je třeba zastavit chod programu v určitém místě programu, vložíme na toto místo programu příkaz STOP. Počítač pak v tomto místě zobrazí hlášení

BREAK IN LINE M /přerušení v řádku s pořadovým číslem M/.

3.6 Pokračování v chodu programu

Jestliže byl program přerušen /buď vložením "CTRL" "C" nebo programově příkazem STOP/ tak, že se na obrazovce zobrazilo

BREAK IN LINE M lze v programu pokračovat vložením povelu

CONT CR /"kont"/.

Jestliže se po přerušení programu zobrazí hlášení **READY** nebo jen dvojtečka, je třeba program spustit povelem **RUN** .

3.7 Vymazání programu

Program můžeme z počítače vymazat jen tehdy, jestliže jsme si dotazem ověřili, že tak s m í m è učinit. Ani nám by nebylo milé, kdyby někdo, třeba jen omylem, vymazal náš program, který jsme do počítače právě vložili a který má například 120 řádků.

NEW CR /"ňů"/ z počítače se vymaží všechny programy.

3.8 Vložení nového programu do počítače

Nejdříve si povelem LIST ověříme, jestli v počítači již nějaký program není uložen.

Pokud se na obrazovce neobjeví výpis nějakého programu, začneme do počítače vkládat náš program. Vkládáme jej přesně tak, jak je zapsán a každý programový řádek ukončíme vložením CR . Tím programový řádek odešleme do paměti počítače.

Jestliže se po vložení povelu LIST zobrazí nějaký program, a my si nejsme jisti, jestli jej s m í m e vymazat, vložíme náš program jako další za program, který je již v počítači uložen. Stačí jen změnit číslování řádků našeho programu tak, aby od posledního řádku předchozího programu vznikla mezera alespoň 2ØØ řádků.

Příklad 20: povelem LIST jsme zjistili, že v počítači je uložen cizí program, který začíná řádkem 8 a končí řádkem 17Ø. Tento program nesmíme vymazat. Musíme proto přečíslovat náš program, který začíná programovým řádkem s pořadovým číslem 1Ø.

Řešení: ke každému pořadovému číslu v našem programu přičteme číslo 390. Původní číslování našeho programu: 1Ø nové číslování: 4ØØ

2Ø 41Ø

3Ø atd. 42Ø

Náš program spustíme povelem RUN 4ØØ CR .

Při vkládání programu do počítače využívejme možnosti automatického číslování jednotlivých programových řádků i možnosti zkráceného vkládání některých klíčových slov /viz příloha č. 1/.

3.9 Automatické číslování programových řádků

Automatické číslování jednotlivých programových řádků zrychlí naši práci. Zabrání však i tomu, abychom omylem programový řádek do počítače nevložili bez pořadového čísla řádku, nebo abychom použili chybné číslo programového řádku.

AUTO CR automatické číslování začíná od programového řádku 1Ø s krokem /rozestupem/ 1Ø mezi řádky;

AUTO M CR automatické číslování s krokem 1Ø mezi řádky začíná až od programového řádku číslo M;

AUTO M,N CR automatické číslování začíná od řádku M, krok mezi jednotlivými řádky je určen číslem N.

Automatické řádkování zrušíme opakovaným vložením CR .

Příklad 21: je třeba nastavit automatické číslování programových řádků od řádku č. 25 s krokem mezi řádky 5.

AUTO 25, 5 CR

3.1Ø Opravy v programu

Opravy v programu můžeme provádět dvěma základními způsoby: buď opravou chyby v příslušném řádku, nebo opakovaným vložením téhož řádku do počítače. U krátkých programových řádků je obvykle rychlejší druhý způsob.

Jestliže chybu zjistíme ještě před odesláním programového řádku do paměti /ještě před vložením CR/, stiskneme tlačítko se šipkou ← . Tímto tlačítkem vymažeme 1 nebo i více znaků a na jejich místa vložíme znaky správné.

Při výskytu chyby ve spuštěném programu provede počítač chybové hlášení a zastaví chod programu. Chybové hlášení se skládá ze dvoumístného kódu zjištěné chyby, slova "ERROR IN" /chyba v řádku/ a z čísla řádku, v němž byla zjištěna chyba. Chybu opravíme takto:

- podle kódu chyby /seznam chybových hlášení je uveden v příloze č. 1/ zjistíme, o jakou chybu se asi jedná;
- chybný programový řádek vyvoláme z paměti vložením " ⓐ " CR ;
- stiskneme tlačítko se šipkou pro ovládání kurzoru a chybný řádek

přesuneme do horní části obrazovky a tlačítky se šipkami umístíme kurzor z a chybný znak;

- vložením "CTRL" "D" chybný znak vymažeme a na jeho místo vložíme správný znak;
- po skončení všech oprav řádek odešleme zpět do paměti vložením CR .

Program spustíme povelem RUN .

Pokud chybu zjistíme sami /například po zobrazení programu povelem LIST/, musíme za znak "@" vložit i číslo řádku, který požadujeme vyvolat z paměti:

@ 1Ø CR

Vlastní opravu chyby provedeme způsobem popsaným v předchozím odstavci.

Příklad 22: vložme do počítače následující program, spusťme jej povelem RUN a odstraňme všechny chyby.

```
1Ø  PRIMT  5 + 3      CR
2Ø  PRINT 10,5 x 3    CR
3Ø  PRINT 36 : 6      CR        RUN   CR
```

Příklad 23: v programu podle příkl. 22 je třeba dodatečně změnit v řádku 1Ø znaménko "+" na znaménko "-". Proveďte opravu.
Řádek 1Ø vyvoláme vložením @ 1Ø CR a provedeme opravu.

Příklad 24: pořadové číslo řádku 1Ø je třeba změnit na číslo 15. Postupovat budeme podle příkladu 23. Musíme si však uvědomit, že přepsáním čísla na obrazovce z 1Ø na 15 jsme nezrušili řádek 1Ø v paměti počítače. Z paměti počítače řádek 1Ø vymažeme tak, že do počítače vložíme číslo tohoto řádku a CR :

```
1Ø  CR          vymazání řádku 1Ø z paměti počítače.
```

3.11 Zrušení určitých programových řádků

Opravy a úpravy programu mohou vést k potřebě vypuštění /zrušení/ určitých programových řádků. Jedná-li se jen o malý počet řádků, stačí postupně do počítače vkládat číslo řádku a CR .

Při potřebě zrušení více řádků programu využíváme příkaz

DELETE M,N CR /čti "dilít"/ — kde číslo M je číslo počátečního řádku a číslo N je číslo posledního řádku, který má být zrušen. Po vložení CR se zruší všechny řádky od řádku číslo M až do řádku číslo N.

DELETE M CR — zruší všechny řádky od řádku s pořadovým číslem M až do konce programu.

S povelem DELETE tedy musíme zacházet velmi promyšleně[1/].

3.12 Přečíslování programových řádků

Při volbě malého kroku mezi řádky může dojít k tomu, že do programu je třeba dodatečně vložit ještě další řádek, ale místo pro něj chybí. Povelem RENUM zajistíme nejen přečíslování všech programových řádků, ale automaticky i čísel řádků v příkazech skoků /GOTO, GOSUB, RUN/.

RENUM CR /čti "rinem"/ — přečíslování programu od programového řádku 1Ø s krokem 10 mezi řádky, tedy např. 1Ø, 2Ø, 3Ø ..

RENUM M CR — přečíslování s krokem 10 začíná až od řádku M .

RENUM M,N CR — přečíslování začíná až od řádku N původního /starého/ programu; původní řádek N získá nové číslo M; krok mezi řádky činí 10.

RENUM M,N,K CR — význam je stejný jako u předchozího typu povelu, ale krok mezi řádky činí K .

1/ O zrušení nebo o přečíslování řádků se přesvědčíme povelem LIST.

3.13 Vyhledání určitého textu v programu

Někdy je třeba v programu najít a změnit určitý text. Například změnit obsah řetězcové proměnné z "PETR" na "PAVEL", proměnnou B$ změnit na Q$, změnit příkaz skoku z GOTO 100 na GOTO 80, atd. Pro tyto účely používáme povel SEEK /čti "sík"/.

SEEK text CR — po chvíli se zobrazí řádek, v němž se nachází
SEEK PETR CR — požadovaný text; konec textu je na obrazovce
SEEK Q$ CR — označen kurzorem.

3.14 Trasování programem

Při zkoušení a hledání chyb v programu /při ladění programu/ je výhodné kontrolovat, kterými řádky program skutečně prochází. Do počítače proto vložíme:

TRON CR — povel pro trasování programem,
RUN CR — spuštění programu.

Na obrazovce se začnou objevovat hranaté závorky s čísly řádků, jimiž program právě prochází.

TROFF CR — konec trasování - ukončení výpisů čísel řádků na obrazovce; povel TROFF vkládáme při zastaveném /přerušeném/ chodu programu.

TRSTEP CR — trasování po jednotlivých programových řádcích. V závorce se zobrazí číslo řádku a počítač "čeká" na stisknutí mezerníku - teprve pak se obsah řádku vykoná.

Trasování po jednotlivých programových řádcích ukončíme vložením CR.

4 Univerzální programy

Programy sestavené podle kapitoly 3.2 fungovaly sice správně, ale měly jednu nevýhodu - byly určeny jen pro výpočty s těmi čísly, /konstantami/, která byla předem vložena do programu. Pro stejný druh výpočtu s jinými čísly bylo třeba sestavit nový program.

Výhodnější jsou proto takové programy, které jsou univerzální a lze do nich vkládat různá čísla.

Příklad 25: sestavme univerzální program pro sečítání čísel A, B, C. Řešení: pro vkládání čísel použijeme příkaz INPUT , pro zobrazení výsledku příkaz PRINT /čti "input", "print"/. Pro zvýšení přehlednosti příručky budeme v dalším textu jednotlivé programy označovat jejich pořadovými čísly. Tato značení do počítače nevkládáme!

PROG-1

```
1Ø INPUT A          příkaz pro vložení čísla A;
2Ø INPUT B          příkaz pro vložení čísla B;
3Ø INPUT C          příkaz pro vložení čísla C;
4Ø W = A + B + C    vlastní výpočet: součet se vloží do pro-
                    měnné označené identifikátorem W;
5Ø PRINT W          zobrazení výsledku /je uložen v prom. W/;
6Ø END              konec programu.
```

Program začíná řádkem 1Ø a končí řádkem 6Ø. Do počítače jej vložíme přesně tak, jak je zapsán. Každý programový řádek ukončíme vložením CR - odešleme jej do paměti počítače. Tedy

```
1Ø INPUT A   CR
2Ø INPUT B   CR   atd.
```

Program spustíme povelem RUN CR /nebo RUN 1Ø CR/. Na obrazovce se objeví otazník - počítač čeká na vložení prvního čísla. Do počítače vložíme např. číslo 5 CR ; tím číslo 5 odešleme do paměti počítače. Zobrazí se další otazník - vložíme další

číslo, například 8 CR . Zobrazí se další otazník - vložíme třetí číslo, například 3 CR . Po vložení posledního čísla se ihned zobrazí výsledek /v našem případě číslo 16/ a program se příkazem END ukončí.

Opakování programu pro stejný druh výpočtu, avšak s jinými čísly A, B, C spustíme vložením RUN CR .

Příklad 26: sestavme program pro násobení 2 čísel. Na obrazovce je třeba zobrazit i název tohoto programu a název od vlastních výpočtů oddělit jedním prázdným řádkem. Po zobrazení výsledku je třeba program opět spustit, ale bez ručního vkládání povelu RUN .

Analýza /rozbor/ příkladu:

- pro zobrazení názvu programu použijeme příkaz PRINT a název programu vložíme do uvozovek;
- prázdný řádek vytvoříme pouhým příkazem PRINT ; protože za tímto příkazem nebude nic uvedeno, nic se nezobrazí a počítač přejde na další řádek;
- povel RUN vložíme do posledního programového řádku; protože ale nepožadujeme opakované zobrazení názvu programu, doplníme povel RUN číslem programového řádku, který následuje jako první za názvem programu.

Řešení: PROG-2

```
1Ø PRINT "SOUCIN 2 CISEL"      název programu;
2Ø PRINT                       vytvoření prázdného řádku;
3Ø INPUT A                     příkaz pro vložení čísla A;
4Ø INPUT B                     příkaz pro vložení čísla B;
5Ø W = A*B                     výpočet výsledku;
6Ø PRINT W                     zobrazení výsledku;
7Ø RUN 2Ø                      opakované spuštění programu;
8Ø END                         označení konce programu.
```

Po zobrazení výsledku se automaticky přejde do řádku 2Ø a počítač "čeká" na vložení další dvojice čísel. Jestliže je třeba tento druh výpočtu již ukončit, stiskneme tlačítka CTRL C .

4.1 Kombinovaný příkaz INPUT

V programech podle příkladů 25 a 26 nám počítač signalizoval zobrazením otazníku, že čeká na vložení čísla. Jistě by bylo vhodné, kdyby počítač zobrazil nejen otazník, ale i proměnnou, kterou máme do počítače vložit - zda číslo A nebo číslo B. Nejsnazší nápovědu vytvoříme tak, že před příkaz INPUT A zařadíme příkaz PRINT "A"; před příkaz INPUT B zařadíme příkaz PRINT "B" .

Příklad 27: upravme program PROG-2 tak, aby signalizoval, které číslo /zda A nebo B/ je třeba do počítače vložit.

PROG-3

```
1Ø PRINT "SOUCIN 2 CISEL"        název programu;
2Ø PRINT                         vytvoření prázdného řádku;
25 PRINT "A"                     zobrazení znaku A;
3Ø INPUT A                       příkaz pro vložení čísla A;
35 PRINT "B"                     zobrazení znaku B;
4Ø INPUT B                       příkaz pro vložení čísla B;
5Ø W = A*B                       výpočet výsledku;
6Ø PRINT W                       zobrazení výsledku;
7Ø RUN 2Ø                        opakované spuštění programu;
8Ø END                           označení konce programu.
```

Program PROG-3 sice funguje, ale je zbytečně dlouhý a jeho vkládání je pracné. Raději proto použijeme kombinovaný příkaz INPUT.

Příkazy v řádcích 25, 3Ø a 35, 4Ø nahradíme příkazy INPUT s výpisem řetězce, tedy:

```
3Ø INPUT "A"; A      zobrazí se:  ?A      vložíme číslo A;
4Ø INPUT "B"; B      zobrazí se:  ?B      vložíme číslo B.
```

Při vysokých nárocích na úspornost programu vystačíme jen s jedním programovým řádkem a jedním kombinovaným příkazem INPUT:

```
3Ø INPUT "A,B"; A, B     zobrazí se: A,B?   po vložení čísla A
                                    se zobrazí další otazník - vložíme číslo B.
```

Řetězcové konstanty /A,B/ v kombinovaném příkaze INPUT musejí být v uvozovkách a ukončeny oddělovačem - středníkem.

Příklad 28: sestavme program pro výpočet druhé mocniny čísla x.

Upozornění: do počítače jsme již vložili program, který je třeba před vložením jiného programu z paměti počítače vymazat povelem NEW CR . Tento povel budeme v dalším textu uvádět za označením programu.

Řešení příkladu 28:

```
PROG-4   NEW CR
  1Ø INPUT "VLOZ CISLO X"; X      nápověda, vložení čísla X;
  2Ø X = X↑2                      výpočet druhé mocniny;
  3Ø PRINT X                      zobrazení výsledku;
  4Ø END                          konec programu.
```

Příkaz v řádku 2Ø je přiřazovací příkaz. Nejdříve se vypočítá druhá mocnina čísla X /tedy X^2/ a výsledkem se "přepíše" původní hodnota uložená v číselné proměnné X. Příkazem v řádku 3Ø se zobrazí tento nový obsah proměnné X. Proměnnou X zde tedy používáme pro dva účely: pro uložení čísla, jehož druhou mocninu je třeba vypočítat,
pro uložení výsledku, tedy druhé mocniny čísla X.

Příklad 29: z programu PROG-4 vypusťme řádek 2Ø.
Řešení: budeme postupovat obdobně jako při výpočtech v dialogovém režimu. To znamená, že druhou mocninu nebudeme vkládat do proměnné X, ale rovnou ji zobrazíme. Číslo uložené v proměnné X pak můžeme použít k jiným /dalším/ výpočtům.

```
PROG-5   NEW  CR
  1Ø INPUT "VLOZ X"; X          nápověda, vložení čísla X;
  2Ø PRINT X ↑ 2                výpočet a zobrazení výsledku;
  3Ø RUN                        opakování výpočtu pro další X;
  4Ø END                        označení konce programu.
```

Poznámka: příkaz END používáme zpravidla v těchto případech:

a/ V paměti počítače je uloženo několik programů a je třeba, aby počítač po provedení prvního programu nepřešel na další programový řádek, jímž začíná další program. Příkazem END se činnost počítače ukončí.

b/ Program je ukončen příkazem RUN /opakované spuštění programu/, ale i ve výpisu programů je účelné zobrazit jejich ukončení. V tomto případě se příkaz END neprovede, ale slouží jen pro informaci o ukončení programu - viz PROG-3, PROG-5.

Příklad 30: do proměnné PR$, která označuje příjmení žáka, je třeba vložit příjmení Novák.

```
PROG-6   NEW  CR
  1Ø INPUT "VLOZ PRIJMENI"; PR$
```

Po spuštění programu se zobrazí: VLOZ PRIJMENI?

Do počítače vložíme: NOVAK CR

Příklad 31: v řádku 2Ø je třeba zobrazit příjmení uložené v PR$.

```
  2Ø PRINT PR$                  zobrazí se:  NOVAK
```

Příklad 32: v řádku 1Ø je třeba vložit příjmení žáka, v řádku 2Ø jeho známku z metematiky. V řádku 5Ø a 6Ø je třeba tyto vložené údaje opět zobrazit.

Analýza úkolu: příjmení vložíme do proměnné P1$. Známku z matematiky vložíme do proměnné Z1. Pro "nápovědu" použijeme kombinovaný příkaz INPUT. Pro zvýšení názornosti vložíme do ř. 3Ø příkaz CLS.

```
PROG-7   NEW   CR
  1Ø INPUT "PRIJMENI"; P1$          nápověda, vložení příjmení;
  2Ø INPUT "ZNAMKA"; Z1             nápověda, vložení známky;
  3Ø CLS                            vymazání obrazovky;
  5Ø PRINT P1$                      zobrazení příjmení;
  6Ø PRINT Z1                       zobrazení známky;
  7Ø END                            konec programu.
```

4.2 Více klíčových slov v jednom programovém řádku

Do každého programového řádku můžeme vložit až 132 znaků /vč. pořadového čísla řádku a mezer mezi znaky/. Jestliže do každého programového řádku vložíme více klíčových slov, program je kratší /má méně programových řádků/ a probíhá rychleji. Jednotlivá klíčova slova však od sebe musíme oddělit oddělovačem "dvojtečka".

Příklad 33: program pro výpočet odmocniny sestavme do 1 řádku.

```
PROG-8   NEW   CR
  1Ø INPUT "VLOZ CISLO"; A  :  PRINT SQR(A)     :  RUN
     nápověda - vložení A;     výpočet a zob-      opětovné spuš-
                               razení výsledku;    tění programu.
```

Program přerušíme vložením CTRL C .

4.3 Vymazání obrazovky

Pro vymazání obsahu obrazovky používáme povel CLS . Zpravidla tento povel zařazujeme na začátek programu.

Příklad 34: sestavme program pro řešení lineární rovnice typu $ax + b = 0$. Před každým výpočtem je třeba obrazovku vymazat.

Analýza: rovnici $ax + b = 0$ upravíme do tvaru vhodného pro výpočet neznámé x :

$$ax = -b$$

$$x = -b/a$$

Příkaz CLS vložíme do řádku 1Ø.

```
PROG-9   NEW   CR
 1Ø CLS                                  vymazání obrazovky;
 2Ø INPUT "VLOZ KOEFICIENT A"; A         vložení koeficientu "a";
 3Ø INPUT "VLOZ KOEFICIENT B"; B         vložení koeficientu "b";
 4Ø X = -1*( B / A)                      vlastní výpočet neznámé x ;
 5Ø PRINT X                              zobrazení výsledku;
 6Ø INPUT Q$                             přerušení chodu počítače;
 7Ø RUN                                  opakované spuštění programu.
```

V řádku 6Ø je použita pomocná proměnná Q$, takže po zobrazení výsledku /ř. 5Ø/ se v ř. 6Ø zobrazí otazník. Přeruší se tak chod programu na dobu, než si poznamenáme výsledek. Po stisknutí libovolného tlačítka a následném CR /nebo jen po vložení CR/ se povelem RUN přejde na začátek programu do ř. 1Ø.

Pokud bychom v ř. 6Ø nepoužili pomocnou proměnnou Q$, výsledek by se zobrazil jen tak krátce, než by byl příkazem CLS v ř. 1Ø vymazán.

Povel CLS lze použít i v dialogovém režimu: CLS CR .

4.4 Vytvoření "okna" na obrazovce

Po zaplnění celé obrazovky je třeba vytvořit prostor pro další programové řádky nebo pro výpočty. Nejvyšší řádek se proto "ztratí" z obrazovky a všechny zbývající řádky se posunou /rolují/ o jeden řádek směrem nahoru.

Jestliže je třeba určité údaje /například název programu/ na obrazovce zachovat, nebo tyto údaje nesmějí být vymazány povelem CLS, vytvoříme povelem WND na obrazovce "okno".

Příklad 34: sestavme program pro řešení lineární rovnice typu ax + b = c . Typ rovnice je třeba zobrazit a zachovat i po zaplnění obrazovky.

Analýza úkolu: rovnici převedeme do tvaru vhodného pro výpočet:

ax + b = c; ax = c - b; x = (c - b) / a .

Název programu "ROVNICE AX + B = C" zabere 1 řádek na obrazovce. Proto za příkaz WND /čti "vindou"/ vložíme číslo 1 - je třeba na obrazovce trvale uchovat 1 řádek.

```
PROG-10 NEW CR
  1Ø CLS                          vymazání obrazovky;
  2Ø PRINT "ROVNICE AX + B = C"   název - označení typu rovnice;
  3Ø WND 1                        "zablokování" jednoho řádku
                                  na obrazovce;
  4Ø INPUT "KOEFICIENT A"; A      vložíme koeficient "a";
  5Ø INPUT "KOEFICIENT B"; B      vložíme koeficient "b";
  6Ø INPUT "KOEFICIENT C"; C      vložíme koeficient "c";
  7Ø X = (C - B) / A              výpočet neznámé "x";
  8Ø PRINT X                      zobrazení výsledku;
  9Ø INPUT QŠ: RUN 4Ø             přerušení a opakované spuštění;
  1ØØ END                         ukončení programu.
```

Příkaz WND N rozdělí obrazovku na dvě části:

- na část, která je určena parametrem /číslem/ N a zůstane "zablokována po celou dobu platnosti příkazu WND N;
- na část /od řádku N+1/, která je nadále přístupná uživateli.

Vytvořené okno zrušíme příkazem WND Ø .

WND N	vytvoří se nepřístupný /zablokovaný/ pás o N řádcích na obrazovce shora;
WND N,M	vytvoří se 2 nepřístupné pásy: první N řádků shora, druhý M sloupců zleva;
WND N,M,P	vytvoří se 3 nepřístupné pásy: N řádků shora, M sloupců zleva; P řádků zdola;
WND N,M,P,R	další nepřístupný pás o R sloupcích zprava.

WND Ø zrušení všech nepřístupných pásů .

4.5 Oddělovače za příkazem PRINT

Jestliže použijeme v příkazu PRINT jako oddělovače středník, zobrazí se znaky následně za sebou:

PRINT "VYSLEDEK ="; 5 CR zobrazí se: VYSLEDEK = 5

Příklad 35: do řetězcových proměnných A$ a B$ vložme řetězce ROZ a DIL; oba řetězce zobrazme v jednom řádku.

A$ = "ROZ" CR

B$ = "DIL" CR

PRINT A$; B$ CR zobrazí se: ROZDIL

Textové řetězce se zobrazují bez mezer. Jestliže má být mezi nimi mezera, musíme ji vložit do příslušného řetězce.

Čísla se zobrazují s jednou mezerou vlevo a jednou mezerou vpravo od čísla. Mezera vlevo od čísla je určena pro případné znaménko "-".

```
1Ø A = 2Ø: B = 15: C = -5        naplnění číselných proměnných;
2Ø PRINT A; B; C                 zobrazí se:  2Ø  15 -5
```

Příklad 36: upravme řádek 8Ø v programu PROG-10 tak, aby bylo zřejmé, že zobrazené číslo je výsledkem.

```
8Ø PRINT "VYSLEDEK="; X
```

Čárka ve funkci oddělovače v příkazu PRINT způsobí, že zobrazení se provede na předem určených tabelačních pozicích. Jednotlivé tabelační pozice jsou od sebe vzdáleny o 14 mezer. Protože na obrazovku se do jednoho řádku vejde 40 znaků, jsou zobrazené znaky posunuty. Při výpisu na tiskárně by byly znaky uspořádány do úhledných sloupců.

Příklad 37: 1Ø PRINT 1, 2, 3, 4, 5, 6, 7, 8

Po spuštění programu povelem RUN se zobrazí:

1 2 3

4 5 6

7 8

Běžně oddělovač "," používáme tehdy, jestliže požadujeme zobrazení uspořádané do dvou, nejvýše do tří sloupců na obrazovce:

PRINT A$, B, C CR

4.6 Zarovnání čísel podle desetinné tečky

BASIC zobrazuje čísla tak, že je zarovnává podle jejich levého okraje, například: 1Ø.8

3ØØ2.25

Pro zarovnání čísel podle desetinné tečky /nebo pro zobrazování čísel menších než 1 v normálním a ne v exponenciálním tvaru/ použijeme klíčové slovo USING /čti "jůsing"/ doplněné příslušnou maskou:

1Ø PRINT USING "####.##" ; 1Ø.8 zobrazí se: 1Ø.8

2Ø PRINT USING "####.##" ; 3ØØ2.25 3ØØ2.25

Počet znaků v masce /v uvozovkách/ musí být stejný, jako je nejvyšší počet míst před a za desetinnou tečkou. Při nesplnění této podmínky se zobrazí znak "%" signalizující, že počet míst v čísle je větší než počet znaků # v masce.

4.7 Počet mezer v textu za příkazem PRINT

Počet mezer mezi prvky za příkazem PRINT lze určit i příkazem SPC počet mezer /čti "spejs"/:

PRINT "CSSR"; SPC 8 ; "PRAHA" mezi slovy CSSR a PRAHA bude 8 mezer.

4.8 Zobrazování ve sloupcích

Pro zobrazování v předem určených sloupcích /pozicích/ používáme příkaz TAB . Za příkazem TAB následuje v závorce číslo sloupce a za závorkou proměnná nebo text, které se mají zobrazit.

V příkazu TAB není třeba používat oddělovač.

Příklad 38: od 5 sloupce je třeba zobrazit křestní jméno žáka, od sloupce 2Ø jeho příjmení. Hlavičku /nadpis tabulky/ je třeba podtrhnout. Jednotlivá jména a příjmení byla již dříve uložena do řetězcových proměnných J1$, P1$; J2$, P2$ atd.

PROG-11

```
5Ø PRINT TAB(5)"JMENO:" ; TAB(2Ø)"PRIJMENI":"  hlavička tabulky;
6Ø PRINT TAB(5)"_____" ; TAB(2Ø)"_________"  podtržení hlavičky;
7Ø PRINT TAB(5)J1$; TAB(2Ø)P1$      první dvojice jméno, příjmení,
8Ø PRINT TAB(5)J2$; TAB(2Ø)P2$      druhá dvojice jméno, příjmení.
```

4.9 Nastavení abecedně-číselného kurzoru do určeného místa

Často je třeba, aby zobrazování výsledků nezačínalo v levém horním rohu obrazovky, ale až v určitém řádku a od určitého sloupce. V těchto případech použijeme příkaz CURS /čti "kurz"/ doplněný číslem s l o u p c e a číslem ř á d k u:

CURS X,Y číslo X určuje číslo sloupce /od Ø do 39/,
          číslo Y určuje číslo řádku /od Ø do 23/.

Příklad 39: text POCITAC ONDRA se má zobrazit od 3 sloupce v 7 řádku na vymazané obrazovce.

```
CLS: CURS 3,7 : PRINT "POCITAC ONDRA"  CR
```

4.10 Relační operátory

Relační operátory se používají v programovém režimu a vyjadřují vztahy mezi čísly, například že číslo A je větší než 1Ø. V BASIC-EXP se používají tyto relační operátory:

A > B	číslo A je větší než číslo B;
A = B	číslo A je rovno číslu B;
A < B	číslo A je menší než číslo B;
A <= B	číslo A je menší nebo je rovno číslu B;

A >= B číslo A je větší nebo je rovno číslu B;

A <> B číslo A není rovno číslu B /je větší nebo menší/.

V posledních 3 uvedených případech do počítače vkládáme oba operátory následně po sobě.

4.11 Podmíněné příkazy

Relační operátory se spojují se splněním nebo nesplněním určité podmínky. Pro vyjádření podmínky se používá klíčových slov

IF jestliže je podmínka splněna,

THEN pak se vykoná činnost uvedená za příkazem THEN,

ELSE jinak se vykoná činnost uvedená za příkazem ELSE.

Za klíčovým slovem IF /čti "if"/ následují klíčová slova THEN /čti "dzen"/ a ELSE /"else"/. Tomuto typu podmíněného příkazu říkáme ú p l n ý podmíněný příkaz.

Příklad 40: s využitím úplného podmiňovacího příkazu zapišme tuto situaci: jestliže si ušetřím 25 Kčs, pak si koupím vstupenku přímo na stadion, jinak se budu dívat na televizní přenos.

Řešení: výši úspor označíme proměnnou U :

```
IF U >= 25 THEN "vstupenka na stadion" ELSE "televizní přenos".
```

Příklad 41: jestliže budu na vysvědčení mít z matematiky jedničku nebo dvojku, pojedu na tábor, jinak se budu doma učit matematiku. Tento výrok /větu, které něco tvrdí/ je třeba převést do tvaru vhodného pro zpracování na počítači.

Analýza úkolu: čísla /= známky/ 1 nebo 2 jsou menší než číslo 3. V podmínce proto budeme porovnávat /testovat/ vloženou známku s číslem 3:

```
1Ø INPUT "ZNAMKA"; Z              vložení známky z matematiky;
2Ø IF Z<3 THEN PRINT "TABOR"      při jedničce nebo při dvojce;
   ELSE PRINT "UCENI DOMA"        při horší známce než dvojka.
```

U n e ú p l n é h o podmiňovacího příkazu chybí větev začínající klíčovým slovem ELSE. Při splnění podmínky se provedou příkazy za klíčovým slovem THEN a přejde se do dalšího programového řádku. Při nesplnění podmínky se příkazy za klíčovým slovem THEN "přeskočí" a přejde se rovnou do dalšího programového řádku.

Příklad 42: program pro sečítání čísel je třeba ukončit, jestliže je vložena nula; jinak se zobrazí mezivýsledek /dílčí součet/ a program se opakuje.

```
PROG-12      NEW   CR
  1Ø INPUT "VLOZ CISLO"; A          nápověda, vložení čísla A;
  2Ø IF A = Ø THEN PRINT W: END     při vložení čísla Ø se zobrazí
                                    výsledek W a program se ukončí;
  3Ø W = W + A: PRINT W             zobrazování mezivýsledků /A ≠ 0/;
  4Ø PRINT: GOTO 1Ø                 prázdný řádek, opakování od ř.2Ø.
```

Příklad 43: do BASIC je třeba převést výrok
"Odpoledne půjdu ven. Jestliže bude pršet, vezmu si pláštěnku".
Analýza: do BASIC je třeba převést druh počasí. Třeba tak, že nejdříve zobrazíme dotaz na počasí - jestliže prší, vložíme do počítače číslo 1:

```
PROG-13      NEW   CR
  1Ø PRINT "JESTLIZE PRSI, VLOZ 1; JINAK VLOZ Ø"     nápověda;
  2Ø INPUT "POCASI: PRSI"; A      dotaz na počasí: vložíme 1 nebo Ø;
  3Ø IF A = 1 THEN PRINT "PLASTENKA"                 při dešti;
  4Ø PRINT "JDI VEN"              jestliže neprší, jdu ven;
  5Ø END                          při dešti jdu ven s pláštěnkou.
```

Kontrolní otázka: jaký je rozdíl v obsahu řádků 2Ø v PROG-12 a v řádku 3Ø v PROG-13 ?
V PROG-12 se v ř. 2Ø při splnění podmínky program ukončil /END/.
V PROG-13 se vždy přejde do ř. 4Ø.

4.12 Příkaz skoku

S příkazem skoku GOTO /čti "goutů"/ jsme se seznámili při spouštění některých programů. Příkazem GOTO však můžeme měnit i pořadí prováděných operací - pořadí programových řádků. Tyto změny je třeba provádět velmi uvážlivě.

Příklad 44: vysvětleme význam programu /k čemu je určen/ PROG-14:

PROG-14

```
1Ø INPUT A
2Ø GOTO 5Ø
3Ø PRINT B
4Ø GOTO 1Ø
5Ø B = A*A
6Ø GOTO 3Ø
```

PROG-15

```
1Ø INPUT A
2Ø GOTO 5Ø
3Ø PRINT B
4Ø GOTO 1Ø
5Ø B = A*A
6Ø GOTO 3Ø
```

PROG-16

```
1Ø INPUT A
2Ø PRINT A↑2
3Ø RUN
```

Řešení: abychom odhalili /dešifrovali/ program PROG-14, doplníme jej čarami, z nicž bude zřejmé, které příkazy a v jakém pořadí se provádějí. Takto doplněný program je označen jako PROG-15.

Z PROG-15 je zřejmé, že po vložení hodnoty A /ř.1Ø/ vypočítáme v ř.5Ø druhou mocninu čísla A a v ř. 3Ø ji zobrazíme. Pak opět přejdeme do ř.1Ø. PROG-14 je tedy poněkud zmatený a lze jej zapsat mnohem elegantněji - viz PROG-16.

Při vysokých nárocích na úspornost programu lze PROG-16 zapsat i do jediného programového řádku:

```
1Ø INPUT A: PRINT A↑2: RUN
```

Který ze všech uvedených programů pro výpočet druhé mocniny je nejpřehlednější?

Příkaz skoku se velmi často používá u podmíněných příkazů. Jestliže za klíčová slova THEN nebo ELSE vložíme příkaz skoku, můžeme činnost při splnění určité podmínky i činnost při nesplnění této podmínky uvést v jiných programových řádcích.

Příklad 45: je třeba sestavit parametricky řízený program, v němž se vypočítá buď obvod nebo plocha obdélníka.
Analýza: u parametricky řízeného programu předem nevíme, zda se bude vypočítávat obvod obdélníka nebo jeho plocha. Druh výpočtu /obvod nebo plocha/ se určí vložením parametru, například písmena O pro výpočet obvodu nebo písmena P pro výpočet plochy. Parametricky řízený program musí být sestaven pro oba druhy výpočtů.

```
PROG-17     NEW  CR
  1Ø PRINT "VYPOCET OBDELNIKA"      nadpis /hlavička/ na obrazovce;
  2Ø INPUT "VLOZ STRANU A"; A       nápověda - vložení délky A;
  3Ø INPUT "VLOZ STRANU B"; B       nápověda - vložení výšky B;
  4Ø PRINT "VLOZ DRUH VYPOCTU:      nápověda - určení druhu výpočtu:
     OBVOD = O,  PLOCHA = P"        obvod nebo plocha;
  5Ø INPUT A$                       vložení písmene O nebo písmene P;
  6Ø IF A$ = "O" THEN 2ØØ           při vložení O přechod na ř. 2ØØ;
  1ØØ PRINT "PLOCHA ="; A*B: END    zobrazení textu a výsledku;
  2ØØ PRINT "OBVOD ="; 2*(A+B): END zobrazení textu a výsledku.
```

Poznámka: v BASIC-EXP není třeba za klíčovými slovy THEN a ELSE uvádět úplný příkaz skoku, tedy ..THEN GOTO 2ØØ. Za klíčová slova THEN a ELSE stačí vložit jen číslo příslušného programového řádku.

4.13 Programové přepínače

Programové přepínače používáme tehdy, jestliže postup výpočtu je závislý na vložení řídícího parametru nebo na hodnotě určitého výsledku.

Jako přepínače programů používáme příkaz skoku typu

ON proměnná GOTO číslo řádku, číslo řádku, číslo řádku, ...

například:

```
ON A GOTO 1ØØ, 12Ø, 2ØØ, 3ØØ, atd.
```

Jestliže A=1, přejde se do toho programového řádku, jehož číslo je uvedeno jako p r v n í za příkazem GOTO. Jestliže A=2, přejde se do toho programového řádku, jehož číslo je uvedeno jako d r u h é za příkazem GOTO. Jestliže A=4, přejde se do toho programového řádku, jehož číslo je uvedeno jako č t v r t é za příkazem GOTO.

Pokud pro určitou hodnotu A nebude v přepínači programů nalezen odkaz na příslušný programový řádek, program pokračuje dalším příkazem, který následuje jako první za příkazem ON A GOTO ...,...

Příklad 46: znázorněme funkci přepínače programů pro A=1 /ř. 5Ø/; A=2 /ř. 1ØØ/; A=3 /ř. 15Ø/; A=4 /ř. 17Ø/. Schema doplňme simulační tabulkou vysvětlující činnost přepínače programů.

```
          1Ø  PRINT "PREPINAC PROGRAMU"
          2Ø  PRINT: INPUT "VLOZ RIDICI PROMENNOU A"; A
          3Ø  ON A GOTO 5Ø, 1ØØ, 15Ø, 17Ø
                |                          |
                |                          |
          4Ø  PRINT "HODNOTA A="; A; "NENI V PREPINACI": RUN 2Ø
při A=1 ->5Ø  PRINT"A="; A; "RADEK 5Ø": RUN 2Ø
    A=2 ->1ØØ PRINT"A="; A; "RADEK 1ØØ": RUN 2Ø
    A=3 ->15Ø PRINT"A="; A; "RADEK 15Ø": RUN 2Ø
    A=4 ->17Ø PRINT"A="; A; "RADEK 17Ø": RUN 2Ø
          18Ø END
```

Program do počítače vložíme běžným způsobem, tedy je jednotlivé programové řádky. Program spustíme povelem RUN 2Ø .

Simulační tabulka:

Vložené A:	přechod do řádku číslo:	zobrazí se:
1	5Ø	A= 1 RADEK 5Ø
2	1ØØ	A= 2 RADEK 1ØØ
4	17Ø	A= 4 RADEK 17Ø
5	4Ø	HODNOTA A= 5 NENI V PREPINACI

Příklad 47: postup řešení kvadratické rovnice je závislý na hodnotě diskriminantu D. Předpokládejme, že pokud

D > 0, pokračuje řešení rovnice od programového řádku 1ØØ,

D = 0, pokračuje řešení rovnice od programového řádku 2ØØ,

D < 0, pokračuje řešení rovnice od programového řádku 3ØØ.

Naznačme možnost použití programového přepínače při řešení této rovnice.

Analýza úkolu: v závislosti na hodnotě diskriminantu je třeba určit hodnotu řídící proměnné /např. proměnné R/:

```
5Ø IF D > Ø  THEN R = 1
55 IF D = Ø  THEN R = 2
6Ø IF D < Ø  THEN R = 3
7Ø ON R GOTO 1ØØ, 2ØØ, 3ØØ
```

4.14 Zvuková signalizace

Zvukovou signalizaci využíváme nejčastěji tehdy, jestliže je třeba obsluhu počítače upozornit například na chybu v odpovědi, na nesprávně vložený parametr apod. Zvukovou signalizaci spustíme příkazem

BEEP m, n kde číslo m určuje výšku tónu a číslo n jeho délku.

Výšku tónu lze nastavit čísly Ø až 7: Ø znamená "žádný tón",
1 znamená nejhlubší tón,
7 znamená nejvyšší tón.

Délka /doba trvání/ tónu se vyjadřuje v násobcích 20 milisekund a může být v rozsahu čísel od Ø do 255.

Lze kombinovat i různé tóny mezi sebou, ale pak každá dvojice parametrů m,n musí být oddělena středníkem.

Příklad 48: obrazovou signalizaci chybně vloženého parametru je třeba doplnit i zvukovou signalizací:

```
BEEP 1,2Ø; 7,5Ø: PRINT "CHYBNE VLOZENY PARAMETR"
```

5. Parametricky řízené programy

Jak již bylo naznačeno, lze stejný parametrický program pouhou změnou řídícího parametru využívat pro různé účely. Není proto třeba po každé sestavovat nový program, ale stačí jen změnit hodnotu řídícího parametru.

5.1 Celočíselná část čísla

Někdy je třeba převést číslo s desetinnými místy na číslo celé, tedy odříznout desetinnou část tohoto čísla. Pro splnění tohoto požadavku použijeme příkaz INT /čti "intýžr"/, například:

```
PRINT INT(1Ø.37)   CR          zobrazí se:   1Ø
PRINT INT(Ø.37)    CR          zobrazí se:   Ø
```

Příklad 49: sestavme program pro převod reálných čísel na čísla celá.

Analýza: reálná čísla jsou celá i desetinná kladná a záporná čísla. Pro převod desetinných čísel na čísla celá použijeme funkce INT :

```
PROG-18      NEW   CR
  1Ø INPUT "VLOZ REALNE CISLO"; A    nápověda, vložení čísla A;
  2Ø A = INT(A)                      převedení čísla A na celé číslo,
  3Ø PRINT "CELE CISLO ="; A         zobrazení celého čísla A.
```

5.2 Generátor pseudonáhodných čísel

Pseudonáhodné čísla jsou čísla, která se vyskytují přibližně se stejnou pravděpodobností jako například čísla vylosovaná ve Sportce. Pseudonáhodná čísla se v počítači ONDRA generují příkazem

RND(M) CR /čti "erendé"/ pseudonáhodné číslo od 0,00..1 do 0,99.

Příklad 5Ø: sestavme program pro zobrazování pseudonáhodných čísel.

```
PROG-19      NEW   CR
  1Ø X = RND(M)      generování pseudonáhodného čísla od Ø do Ø.999.
  2Ø PRINT X: RUN                zobrazení X, spuštění programu.
```

Na číslo větší než 1 převedeme generované pseudonáhodné číslo tak, že jej vynásobíme jiným číslem /například číslem 1Ø/. Příkazem INT pak z tohoto nového čísla odřízneme jeho desetinnou část.

Příklad 51: sestavme program generující pseudonáhodná čísla od nuly do 10.

```
PROG-20    NEW   CR
  1Ø X = RND(M)                  generované pseudonáhodné číslo;
  2Ø X = X*11                    převod na číslo od nuly do 10,99;
  3Ø X = INT(X)                  převod na celočíselné číslo;
  4Ø PRINT X: RUN                zobrazení, opakované spuštění.
```

Příklad 52: sestavme program pro hru "hledání tajného čísla". Tajné číslo je generované pseudonáhodné číslo upravené např. tak, aby se měnilo od nuly do 10.

Řešení: z počítače vymažeme předchozí programy a zobrazíme název nového programu:

```
PROG-21    NEW   CR
  1Ø PRINT "HLEDANI TAJNEHO CISLA": WND 2    název programu a vy-
                                             tvoření okna pro název;
  2Ø A = RND(M): A = A*11                    vytvoření čísla od nuly do 10,99;
  3Ø A = INT(A)                              převedení čísla A na celočíselné;
Vložení odpovědi a porovnání s číslem A:
  4Ø INPUT "URCI TAJNE CISLO"; B             nápověda - vložení odpovědi;
  5Ø IF B = A THEN PRINT "SPRAV              při uhádnutí tajného čísla;
     NE": END
  6Ø BEEP 7,2Ø: PRINT "CHYBNE                signalizace při chybě, vložení
     CISLO": GOTO 4Ø                         jiné odpovědi, opakování testu.
```

Příklad 53: upravme PROG-21 tak, aby mohl být předem určen interval pro tajnou číslici, například: Ø až 5, nebo Ø až 2Ø, atd. Jedná se tedy o parametricky řízený program.

Řešení: úprava spočívá v doplnění PROG-21 o řádek 15 a ve změně ř. 2Ø. V ř. 15 budeme požadovat vložení horní meze /HM/ pro tajné číslo. V ř. 2Ø nahradíme násobení číslem 11 násobením parametrem HM+1:

```
15 INPUT "VLOZ HORNI MEZ"; HM          vložení horní meze;
2Ø A = RND(M): A = A*(HM + 1)          bez přičtení 1 bychom ne-
                                       dosáhli horní meze.
```

Příklad 54: sestavme parametricky řízený program pro zkoušení žáků ze sečítání a násobení. Jednotlivá čísla X a Y se nebudou do počítače vkládat, ale počítač je bude generovat. Jejich horní mez se bude nastavovat paramtrem HM.

PROG-22 NEW CR

```
1Ø INPUT "1=SECITANI, 2=NASOBENI"; A  volba druhu výpočtů;
2Ø INPUT "VLOZ HORNI MEZ"             nápověda - jaká má být horní mez;
3Ø X = RND(M): X = INT(X*(HM + 1))    generování proměnné X;
4Ø Y = RND M : Y = INT(Y*(HM + 1))    generování proměnné Y;
5Ø ON A GOTO 6Ø, 1ØØ                  přepínač programů - podle A;

6Ø PRINT X; "+"; Y; "="               zobrazení součtu čísel X + Y;
7Ø W = X + Y                          výpočet výsledku při sečítání;
8Ø GOTO 2ØØ                           pokračuje se od ř. 2ØØ;

1ØØ PRINT X; "*"; Y; "="              zobrazení součinu čísel X*Y;
11Ø W = X*Y                           výpočet výsledku při násobení;

2ØØ INPUT "VLOZ VYSLEDEK"; C          vložení odpovědi žáka;
21Ø IF W = C THEN BEEP 1,1Ø:          signalizace při správné odpově-
    GOTO 3Ø                           di; opakování s jinými X a Y;
22Ø BEEP 7,5Ø: PRINT "CHYBA           signalizace při chybné odpově-
    - SPRAVNY VYSLEDEK:"; W:          di, zobrazení správného výsled-
    GOTO 3Ø                           ku, opakování s jinými X a Y.
```

Poznámky k PROG-22:

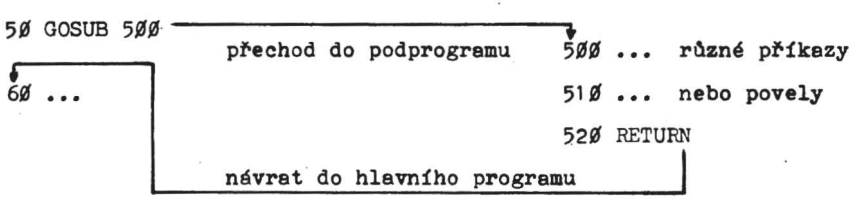
1. Nepodmíněný skok v ř. 8Ø "GOTO 2ØØ" zajišťuje, aby ihned po sečítání nenásledoval ř. 1ØØ - násobení.

2. Při rozšíření PROG-22 o operace "odečítání", "násobení", apod. bude třeba programově řešit /"ošetřit"/ situace, kdy Y bude větší než X, neboť výsledkem by byla záporná nebo desetinná čísla.

5.3 Podprogramy

Často se stává, že v programu se opakuje určitá posloupnost /sled/ stejných příkazů. V BASIC není třeba tuto posloupnost příkazů po každé opisovat. Stačí ji zapsat jen jednou - jako podprogram a v případě potřeby ji vyvolat příkazem GOSUB /čti "gousab"/. Za příkazem GOSUB musí následovat číslo řádku, od něhož podprogram začíná, tedy například GOSUB 5ØØ .

Každý podprogram začíná vstupním bodem - číslem řádku /např. řádkem 5ØØ/. Výstupním bodem každého podprogramu je příkaz RETURN, který musí být posledním příkazem každého podprogramu:



Jestliže se v programu narazí na příkaz GOSUB /např. GOSUB 5ØØ/,
- do zvláštní části paměti /zásobníku/ se poznamená číslo řádku a místo v řádku, kde je uveden příkaz GOSUB;
- provede se skok na vstupní bod podprogramu /např. na řádek 5ØØ/ a postupně se vykonají všechny příkazy uvedené v podprogramu;
- po nalezení výstupního bodu podprogramu /RETURN/ se přejde zpět do hlavního programu na první příkaz, který následuje za GOSUB.

Příklad 55: sestavme podprogram pro signalizaci chybné odpovědi, který je součástí jiného, delšího programu. Správná odpověď je uložena v řetězcové proměnné W$, odpověď žáka se vkládá do řetězcové proměnné Z$. Podprogram musí začínat na řádku 3ØØ.

PROG-23

```
 1Ø  ....                              různé příkazy a povely;
 5Ø GOSUB 3ØØ                          přechod na podprogram;
 6Ø ...                                další příkazy a povely;
 1ØØ END                               konec hlavního programu;

 3ØØ INPUT "VLOZ ODPOVED"; Z$          nápověda - odpověď žáka;
 31Ø IF Z$ = W$ THEN BEEP 1,1Ø: GOTO 32Ø  při správné odpovědi;
     ELSE BEEP 7,5Ø:PRINT "CHYBA"      při chybné odpovědi žáka;
 32Ø RETURN                            návrat do hlavního programu.
```

Příklad 56: sestavme program, který by nahradil kostku u hry "člověče nezlob se". Generování náhodného čísla řešme podprogramem.

Řešení: nejdříve sestavíme podprogram pro generování čísel od 1 do 6 /zjišťujeme, zda vůbec lze tento podprogram sestavit/: NEW CR

```
 1ØØ  X = INT(RND(M)*6 + 1)   zkrácený zápis generování X od 1 do 6,9
 11Ø  RETURN                  návrat do hlavního programu.
```

Sestavíme hlavní program:

PROG-24

```
 1Ø CLS: PRINT "KOSTKA"        vymazání obrazovky, hlavička;
 2Ø PRINT: INPUT "VLOZ CR"; Q$ prázdný řádek, přerušení;
 3Ø GOSUB 1ØØ                  vyvolání /spuštění/ podprogramu;
 4Ø PRINT X                    zobrazení některé z číslic 1 - 6;
 5Ø RUN 2Ø                     opakování "házení kostkou".
```

Program můžeme do počítače vložit tak, jak je zapsán; počítač si "sám" jednotlivé řádky správně zařadí.

Názorně si přechod z hlavního programu do podprogramu a zpět ověříme v režimu "trasování programem". Do počítače vložíme

TRON: RUN: CR přepnutí na "trasování", spuštění.

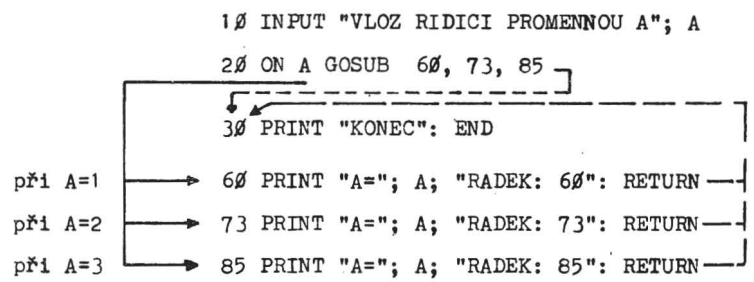
/přerušení chodu programu: CTRL C

ukončení trasování: TROFF CR /.

Z určitého podprogramu můžeme přejít příkazem GOSUB ... do dalšího podprogramu. Z něho se příkazem RETURN vrátíme do původního podprogramu. Z původního podprogramu se příkazem RETURN vrátíme zpět do hlavního programu.

5.4 Přepínače podprogramů

Přepínače podprogramů fungují obdobně jako přepínače programů. V závislosti na hodnotě řídící proměnné /například proměnné A/ umožňují umožňují vyvolat příslušný podprogram. Návrat do hlavního programu je automatický - příkazem RETURN .



Příklad 57: sestavme program pro násobení nebo sečítání čísel X a Y.

```
PROG-25     NEW   CR
 10 INPUT "VLOZ CISLA X, Y"; X,Y   nápověda - vložení čísel X a Y;
 20 INPUT "1=SECITANI,  2=NASOBENI"; A  volba druhu výpočtu  - A;
 30 ON A GOSUB 100, 200     volba podprogramu podle hodnoty A;
 40 PRINT "W="; W: RUN      zobrazení výsledku, opakování programu;
100 W = X + Y : RETURN      podprogram pro sečítání čísel X a Y;
200 W = X * Y : RETURN      podprogram pro násobení čísel X a Y.
```

5.5 Komentáře - poznámky v programu

Až dosud jsme si uváděli vysvětlující text /komentáře, poznámky/ jen v textu příručky. Lze je však vkládat i do programu. V jednotlivých programových řádcích se komentáře oddělují znakem "!". Jestliže řádek obsahuje před komentářem nějaký příkaz nebo povel, musí být před vykřičníkem oddělovač "dvojtečka".

Při běžném chodu programu se text za vykřičníkem nezobrazí. Zobrazí se však při výpisu programu povelem LIST , takže umožňuje snadnou orientaci ve výpisu programu.

Příklad 58: doplňme program PROG-25 vysvětlujícím komentářem.

```
PROG-26
  5  !DOPLNENY PROGRAM PROG-25
  1Ø INPUT "VLOZ X,Y"; X,Y :!VSTUPNI DATA
  2Ø INPUT "1=SECITANI,  2=NASOBENI"; A :!VOLBA DRUHU VYPOCTU
  3Ø ON A GOSUB 1ØØ, 2ØØ :!PREPINAC PODPROGRAMU
  4Ø PRINT W: PRINT: RUN :!VYSLEDEK, NOVE SPUSTENI PROGRAMU
  1ØØ W = X + Y: RETURN:!PODPROGRAM-SOUCET
  2ØØ W = X * Y: RETURN :!PODPROGRAM-SOUCIN
  21Ø !KONEC PROG-26
```

5.6 Logické operátory

Logické operátory vyjadřují vztah mezi proměnnými v programu a umožňují jeho řízení. BASIC-EXP využívá těchto operátorů:

AND	průnik /logický součin/,
OR	sjednocení /logický součet/,
NOT	negace - inverze,
XOR	neekvivalence.

V jednoduchých příkladech zpravidla vystačíme jen s logickými operátory pro logický součin AND /čti "end"/ a pro logický součet OR /čti "or"/.

Příklad 59: v určitém programu je třeba vložit do řádku 1ØØ test, podle něhož se výpočet ukončí, jestliže číslo A = Ø nebo B = Ø. V opačném případě výpočet pokračuje od ř. 13Ø.

Analýza úkolu: výpočet se má ukončit, jestliže <u>některé</u> z čísel A nebo B je rovno nule. Jedná se tedy o sjednocení /logický součet/. Pro řízení programu proto použijeme logický operátor OR .

```
1ØØ IF A = Ø  OR  B = Ø  THEN  END    při A=Ø nebo B=Ø "konec";
11Ø GOTO 13Ø                          při jiných hodnotách skok na 13Ø.
```

Příklad 6Ø: výpočet je třeba ukončit, jestliže v proměnné A je číslo Ø a s o u č a s n ě v řetězcové proměnné A$ je uložen znak K.

```
13Ø IF A = Ø  AND  A$ = "K"  THEN  END      logický součin.
```

Příklad 61: generátor pseudonáhodných čísel je zařazen v řádku 2ØØ a generuje čísla od X=Ø do X=2Ø. Z těchto čísel je třeba vybrat jen ta čísla X, která jsou větší než 5 a s o u č a s n ě jsou menší než 15. Pokud je generováno jiné číslo /například 3 nebo 18/, znova spustíme generátor čísel.

Analýza úkolu: jestliže mají být vybrána čísla X, která jsou větší než 5 a současně jsou menší než 15, jedná se o logický součin.

```
21Ø IF  X>5  AND  X<15  THEN  PRINT X ELSE GOTO  2ØØ
```

5.7 <u>Pole a prvky pole</u>

V řadě případů je třeba pracovat s prvky, které jsou součástí určité množiny. Je například třeba sestavit jmenný seznam žáků určité třídy. S našimi znalostmi můžeme postupovat tak, že jednotlivá příjmení budeme vkládat do různých řetězcových proměnných:

```
A$ = "ADAMEK" <u>CR</u>          B$ = "BERKA" <u>CR</u>
C$ = "BOROVSKY" <u>CR</u>        D$ = "BLAHOVA" <u>CR</u>      atd.
```

Jak bychom ale postupovali, kdybychom měli sestavit jmenný seznam žáků za celou školu - stačil by ném vůbec počet proměnných?

Programovací jazyky umožňují pracovat s poli a s prvky těchto polí. Zjednodušeně lze říci, že "pole" znamená určitou "tabulku". Každé pole musí mít svůj název - musí být označeno identifikátorem pole. Každý prvek pole je označen stejným identifikátorem a pořadím:

PZ$(1) první prvek pole "PZ$" - například příjmení prvního žáka,
PZ$(2) druhý prvek pole "PZ$" příjmení druhého žáka,
PZ$(3) třetí prvek pole "PZ$" příjmení třetího žáka.

Pokud je počet žáků /tedy prvků pole "PZ$" větší než 10, je třeba pro prvky tohoto pole předem vymezit příslušné místo v paměti počítače. Musíme provést d e k l a r a c i určitého pole.

Deklaraci /vymezení/ určitého pole provádíme příkazem DIM, za který uvedeme název tohoto pole a do závorky počet jeho prvků.

Příklad 62: sestavme program pro vložení příjmení 23 žáků určité třídy.

Analýza: pole, do něhož budeme vkládat jednotlivá příjmení, si označíme například symbolem P$. Protože prvků je více než 10 /23 žáků/, musíme nejdříve provést deklaraci tohoto pole.

```
PROG-27    NEW   CR
  1Ø DIM P$(23)                    deklarace pole P$ s 23 prvky;
```

Program sestavíme tak, aby sám určoval pořadí jednotlivých prvků. Pro tuto činnost použijeme pomocnou číselnou proměnnou "I", kterou budeme postupně zvyšovat o jedničku. Nejdříve však musíme tuto pomocnou proměnnou vynulovat.

```
  2Ø I = Ø                         vynulování pomocné proměnné I;
  3Ø I = I + 1                     zvýšení hodnoty I o jedničku;
  4Ø PRINT I                       zobrazení hodnoty /pořadí/  I;
  5Ø INPUT P$(I)                   vložení příjmení I-tého žáka;
  6Ø IF I= 23 THEN END     test: bylo již vloženo všech 23 příjmení?
     ELSE GOTO 3Ø          jestliže ano, pak konec; jinak řádek 3Ø.
```

Příklad 63: zobrazme příjmení prvního, druhého a pátého žáka ze seznamu, který jsme do počítače vložili v průběhu PROG-27.

Analýza: pořadí prvku v každém poli je určeno pořadovým číslem uvedeným v závorce za příslušným identifikátorem pole. Protože pořadí žáků je uvedeno v zadání příkladu 63, stačí jen do počítače vložit

```
PRINT P$(1)   CR        zobrazí se příjmení 1. žáka;
PRINT P$(2)   CR        zobrazí se příjmení 2. žáka;
PRINT P$(5)   CR        zobrazí se příjmení 5. žáka.
```

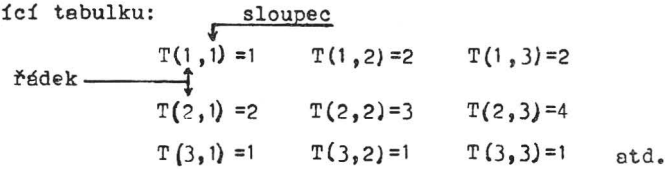
Příklad 64: navrhněte postup pro vložení známek 5 žáků ze 3 předmětů.

Analýza: nejdříve si sestavíme pracovní tabulku, kterou si přiblížíme zadání příkladu 64:

Pořadí žáka:	matematika:	fyzika:	dějepis:
první	1	2	2
druhý	2	3	4
třetí	1	1	1
čtvrtý	4	3	5
pátý	3	2	4

Jak například zjistíme známku druhého žáka z dějepisu /tedy třetího předmětu/? Ve druhém řádku nalezneme třetí sloupec; na průsečíku druhého řádku a třetího sloupce je uvedena požadovaná známka.

Známka každého žáka z libovolného předmětu je tedy určena dvěma údaji: pořadovým číslem řádku a pořadovým číslem sloupce v tabulce. Jestliže si každý prvek v tabulce označíme identifikátorem T a v závorce pořadovým číslem řádku a sloupce, získáme následující tabulku:



Abychom vůbec mohli tabulku do počítače vložit, musíme pro ni předem deklarovat /vymezit/ místo v paměti počítače:

```
DIM T(5,3) CR
   |  |  |
   |  |  |_____ počet sloupců tabulky /pole T/
   |  |
   |  |________ počet řádků tabulky /pole T/
   |
   |___________ jméno tabulky /pole T/
```

Nejjednodušší, avšak nejpracnější postup pro vložení známek 5 žáků ze tří předmětů do tabulky v počítači spočívá v ručním vkládání jednotlivých údajů:

T(1,1) = 1	CR	T(1,2) = 2	CR	T(1,3) = 2	CR		
T(2,1) = 2	CR	T(2,2) = 3	CR	T(2,3) = 4	CR		
T(3,1) = 1	CR	T(3,2) = 1	CR	T(3,3) = 1	CR	atd.	

Příklad 64: je třeba zobrazit:

a/ známku prvního žáka ze třetího předmětu,

b/ známku druhého žáka z prvního předmětu,

c/ známku prvního žáka z prvního předmětu.

Řešení:			
PRINT T(1,3)	CR	zobrazí se:	2
PRINT T(2,1)	CR	zobrazí se	2
PRINT T(1,1)	CR	zobrazí se:	1

Poznámky: 1. Deklaraci více polí jediným příkazem DIM provedeme takto: DIM P$(23), T(5,3), Z(27)

2. Po opakovaném spuštění programu povelem RUN se může zobrazit chybové hlášení "DD", které znamená vícenásobnou deklaraci téhož pole. Povel RUN proto doplníme číslem řádku, který následuje za řádkem, v němž je provedena deklarace /například RUN 20/.

3. Do řádku, v němž je uvedena deklarace, nevkládáme komentáře a poznámky /texty po příkazu "!"/.

5.8 Programové cykly

Programové cykly výrazně zkracují programy, v nichž se nějaká činnost opakuje tak dlouho, dokud není splněna určitá podmínka.

5.8.1 Programový cyklus vlastní konstrukce

Příklad 65: je třeba sestavit program pro výpočet druhé mocniny. Výpočet se musí ukončit při vložení čísla Ø.
Sestavení programu by nám nemělo dělat žádné obtíže:

```
PROG-28    NEW  CR
  1Ø INPUT X: IF X = Ø THEN END     vložení čísla X a jeho test na Ø;
  2Ø PRINT X↑2: PRINT: RUN          zobrazení výsledku, volný řádek,
                                    opakování celého výpočtu.
```

Ve složitějších programech se v programových cyklech obtížněji orientujeme. Je proto účelné opticky si jednotlivé cykly vyznačit posunutím textu například o 2 znaky vpravo.

Příklad 66: sestavme program pro vytvoření jmenného seznamu až 40 žáků, doplněného o jejich známky ze tří předmětů.
Analýza úkolu: žáků může být 10, ale i 38. Budeme proto konec vkládání počítači signalizovat vložením znaku "K".
V programu je třeba vytvořit 2 různá pole:

- pole se řetězcovými proměnnými pro příjmení jednotlivých žáků, například pole P$ s prvky od 1 do 40 /deklarace: DIM P$(4Ø)/;
- dvourozměrné pole /tabulka/, v němž budou uloženy známky z jednotlivých předmětů, například pole Z. Protože pole Z bude mít 3 sloupce /J=3/ a může mít až 40 řádků /I =40/, je třeba pole Z deklarovat na maximální možný počet prvků, tedy DIM Z(4Ø,3).

Jako řídící proměnnou v cyklu použijeme proměnnou I, kterou budeme postupně zvyšovat o jedničku. Současně tuto proměnnou využijeme pro signalizaci řádku tabulky - pořadového čísla žáka.

PROG-29 NEW CR

```
1Ø DIM P$(40), Z(4Ø,3)          deklarace obou použitých polí;
2Ø I = 0                        vynulování řídící proměnné I;
3Ø   I = I + 1                  zvýšení řídící proměnné o 1;
4Ø   PRINT I: INPUT P$(I)       signalizace pořadí - vložení
                                příjmení I-tého žáka;
5Ø   INPUT "VLOZ 1. ZNAMKU"; Z(I,1)    vložení první známky;
6Ø   INPUT "VLOZ 2. ZNAMKU"; Z(I,2)    vložení druhé známky;
7Ø   INPUT "VLOZ 3. ZNAMKU"; Z(I,3)    vložení třetí známky;

8Ø   INPUT "DALSI ZAK"; Q$      vložíme buď CR /další žák/,
                                nebo K /konec vkládání/;
9Ø   IF Q$ = "K" THEN 1ØØ      jestliže je konec vkládání,
     ELSE PRINT: GOTO 3Ø        jinak prázdný řádek, další žák;
1ØØ PRINT "KONEC VKLADANI"      konec vkládání.
```

Při prvním průchodu cyklem se řídící proměnná v ř. 2Ø vynuluje. V ř. 3Ø se její hodnota zvýší o jedničku, tedy z Ø na 1. V ř. 4Ø nám hodnota I signalizuje, že budeme vkládat údaje týkající se prvního žáka.

Pokud nebude v ř. 8Ø vložen znak K , program se vrátí z ř.9Ø do ř. 3Ø. V ř. 3Ø se hodnota I zvýší z 1 na 2. V ř. 4Ø nám hodnota I /I=2/ signalizuje, že budeme vkládat údaje týkající se druhého žáka.

Uvedený cyklus se opakuje tak dlouho, dokud v ř. 8Ø do počítače nevložíme znak K . Při vložení znaku K se zobrazí text uvedený v ř. 1ØØ "KONEC VKLADANI".

Posunutí vkládaných příkazů v řádcích 3Ø až 9Ø nám opticky signalizuje programový cyklus.

Příklad 67: je třeba zobrazit příjmení a známky jednotlivých žáků, které byly do počítače vloženy v předchozím příkladu.
Analýza úkolu: řídící proměnná "I" má po ukončení vkládání v PROG -29 stejnou číselnou hodnotu, jako je počet vložených příjmení. Pro cyklus "zobrazování" použijeme jinou řídící proměnnou /například. "U"/ a její hodnotu budeme po každém zvýšení o jedničku porovnávat s hodnotou proměnné I. Při shodě obou těchto proměnných /U=I/ se zobrazování ukončí.

PROG-29 pokračování:

```
110 U = 0                          vynulování řídící proměnné U;
120   U = U + 1                    zvýšení hodnoty U o jedničku;
130   PRINT U; P$(U);              zobrazení pořadí a příjmení;
      Z(U,1) ; Z(U,2); Z(U,3)      zobrazení jednotlivých známek;
140   IF U = I  THEN 150           test: dosáhla U hodnoty I ?
      ELSE 120                     jestliže ano, pak řádek 150,
                                   jinak ř. 120 - další příjmení;
150 PRINT "KONEC PROGRAMU": END    ukončení zobrazování.
```

Poznámka: počítač "vychrlí" jednotlivá příjmení a příslušné známky. Pokud je žáků více než volných řádků na obrazovce, první údaje se z obrazovky ztratí ještě dříve, než si je stačíme zapsat. Program můžeme cyklicky přerušovat například tím, že do něho vložíme řádek

```
125   INPUT Q$                     další údaje se zobrazí až po
                                   stisknutí libovolného tlačítka.
```

Programový cyklus vlastní konstrukce používáme zpravidla tehdy, jestliže předem neznáme počet opakování cyklu. Sami však v něm musíme zajistit změny řídící proměnné a zajistit testování na podmínku, která ukončuje cyklus.

5.8.2 Programový cyklus typu FOR - TO - STEP/NEXT

Programový cyklus typu FOR-TO-STEP/NEXT používáme tehdy, jestliže předem známe počet opakování. Cyklus je řízen proměnnými, například proměnnými I a K, které následují za klíčovými slovy:

FOR I = cyklus začíná od čísla /dolní meze/, které následuje za řídící proměnnou I: například FOR I=1 ; FOR I=5, atd.

TO cyklus končí horní mezí /včetně!/, která je určena číslem následujícím za příkazem TO , např. TO 1Ø ; TO 5Ø;

STEP K číslo za příkazem STEP určuje změnu proměnné I; např. při K = .5 se hodnota řídící proměnné I postupně zvyšuje o Ø.5; jestliže STEP /krok/ = 1, není třeba jej uvádět;

NEXT I řídící proměnná I se zvýší o krok K a testuje se, zda I nepřekročila horní mez: při překročení horní meze se pokračuje dalším příkazem, který následuje za NEXT, jinak se cyklus opakuje.

/čti "for ... tů ... step ... next"/

Příklad 68: je třeba vytvořit jmenný seznam 15 žáků.
Řešení: jednotlivá jména budeme ukládat například do pole "LØ". Jako řídící proměnnou použijeme proměnnou H. Protože krok v pořadí mezi jednotlivými žáky činí 1, není třeba jej uvádět.

```
PROG-3Ø    NEW   CR
  1Ø DIM LØ(15)                  deklarace použitého pole LØ 15 ;
  2Ø   FOR H = 1  TO  15         určení horní meze cyklu;
  3Ø   PRINT H: INPUT LØ(H)      zobrazení poř. čís., vlož. příjmení;
  4Ø   NEXT H                    konec cyklu;
  5Ø PRINT "KONEC VKLADANI"      další programové příkazy.
```

Upozornění:

a/ Určitý cyklus musí být ukončen stejným identifikátorem, jaký byl použit za klíčovým slovem FOR , například

```
300 FOR I=1   TO 3   STEP 0.1   cyklus od 1 do 3 s krokem 0,1;
302 PRINT"I="; I                příkazy uvnitř cyklu /tělo cyklu/;
304 NEXT I                      konec cyklu;

310 FOR AY=-5   TO 5            cyklus od -5 do +5 s krokem 1;
320 PRINT "AY="; AY             tělo cyklu;
330 NEXT AY                     konec cyklu.
```

b/ Příkaz cyklu může být zapsán i v jediném programovém řádku:

```
350 FOR U=2 TO 20 STEP 2: PRINT INT(SQR(U)):NEXT U
```

c/ Cyklus je možno řídit i parametricky. Před vstupem do cyklu je však třeba určit konkrétní parametry cyklu:

```
400 INPUT "DM"; DM              DM  znamená "dolní mez";
410 INPUT "HM"; HM              HM  znamená "horní mez";
415 INPUT "K"; K                K   znamená "krok";
417 FOR J= DM TO HM STEP K      nastavení parametrů cyklu;
418 PRINT J↑2: NEXT J           tělo cyklu;  konec cyklu.
```

Příklad 69: vytvořený seznam 15 žáků /př. 68/ je třeba zobrazit. Dále je třeba zjistit, kolik je v tomto seznamu hochů a kolik je dívek.

Analýza úkolu: v češtině se příjmení dívek /téměř vždy/ liší od příjmení hochů koncovkou "á". Stačí tedy příkazem RIGHT$(A$,1) oddělit poslední písmeno z každého příjmení a podle tohoto písmene určit, zda se jedná o dívku nebo o hocha. Počet dívek budeme připočítávat do proměnné D, počet hochů do proměnné H. Program bude navazovat na předchozí PROG-30.

PROG-31 pokračování PROG-3Ø

```
 55 H = Ø: D = Ø                    vynulování proměnných H a D;
 6Ø FOR  I = 1  TO 15               určení parametru cyklu;
 7Ø PRINT I; L$(I)                  poř. číslo I, příjmení Itého žáka;
 8Ø A$ = L$(I)                      převedení příjmení z L$(I) do A$;
 9Ø A$ = RIGHT$(A$,1)               z příjmení uloženého v A$ odříz-
                                    neme 1 znak zprava a tento odříz-
                                    nutý znak vložíme zpět do A$;
 1ØØ IF A$ = "A" THEN D = D + 1     při A$="A", jde tedy o příjmení
                                    dívky, zvýší se obsah D o jedničku
    ELSE H = H + 1                  jinak se zvýší obsah H o jedničku;
 11Ø NEXT I: CLS                    konec cyklu; vymazání obrazovky;
 12Ø PRINT "CELKEM DIVEK:"; D       zobrazení počtu dívek;
 13Ø PRINT "CELKEM HOCHU:"; H       zobrazení počtu hochů,
 14Ø END                            konec programu.
```

Poznámky: příkaz END je do ř. 14Ø vložen proto, aby chod programu nepokračoval řádkem 3ØØ /viz str. 69/.
Pokud již byla jednotlivá příjmení vložena v PROG-3Ø do L$(I), musíme PROG-31 spustit příkazem GOTO 55, neboť povelem RUN 55 bychom všechna příjmení vymazali.

Kontrolní otázka: jak je třeba upravit PROG-31, aby se po zobrazení každého příjmení chod počítače přerušil /zaznamenání výsledků/? Použijeme pomocnou proměnnou, například U$ a do PROG-31 vložíme například řádek

```
 1Ø5 INPUT U$                       přerušení chodu programu.
```

5.8.3 Programové cykly typu WHILE-DO/WEND a REPEAT-UNTIL

Uvedené typy cyklů jsou převzaty z programovacího jazyka PASCAL /čti "paskal"/. Cyklus WHILE-DO/WEND /čti "vajl-dů-vend"/ probíhá tak dlouho, dokud je splňována předem určená podmínka.

Příklad 70: s využitím cyklu typu WHILE-DO/WEND sestavme program pro výpočet druhé odmocniny z čísel větších než 5.

Analýza úkolu: výpočet se bude opakovat tak dlouho, dokud vložené číslo A bude větší než 5. Mezi klíčová slova WHILE a DO je třeba vložit podmínku A>5 .

Abychom do cyklu vůbec vstoupili, nastavíme předem hodnotu A na 6:

```
PROG-32    NEW   CR
  1Ø A = 6                          nastavení počáteční hodnoty A;
  2Ø WHILE   A>5   DO               určení podmínky cyklu;
  3Ø INPUT A: PRINT SQR(A)          vložení čísla, výpočet odmocniny;
  4Ø PRINT: WEND                    prázdný řádek, konec cyklu;
  5Ø PRINT "CISLO NESPLNUJE PODMINKU"  výstup z cyklu při A<=5.
```

Po ukončení cyklu se přejde na první příkaz za příkazem WEND .

Cyklus typu REPEAT-UNTIL probíhá tak dlouho, dokud n e n í s p l n ě n a předem určená podmínka /dokud nenastane určitá situace/.

Příklad 71: opakovaným vkládáním čísel z klávesnice je třeba určit, které náhodné číslo bylo vygenerováno a vloženo do proměnné B.

Analýza úkolu: náhodné číslo B vygenerujeme použitím příkazů RND a INT . Cyklus REPEAT-UNTIL /čti "ripít-antil"/ bude probíhat tak dlouho, dokud číslo A vložené klávesnicí nebude rovno číslu B.

```
PROG-33
  1Ø B = RND(M): B = INT(B*1Ø)   vygenerování neznámé číslice;
  2Ø   REPEAT                    začátek cyklu;
  3Ø   INPUT "VLOZ CISLICI"; A   nápověda - vložení číslice;
  4Ø   P = P + 1                 v P zjišťujeme počet pokusů;
  5Ø   UNTIL  A = B              konec cyklu - podmínka;
  6Ø PRINT "POCET POKUSU:"; P    zobrazení počtu pokusů při A = B.
```

Po skončení cyklu se přejde na první příkaz za podmínkou UNTIL... .

5.8.4 Vnější a vnitřní cykly

Programové cykly lze i vkládat /vnořovat/ do sebe. Je však třeba dodržet zásadu, že vnitřní cyklus n e s m í mít svůj koncový bod za koncovým bodem vnějšího cyklu.

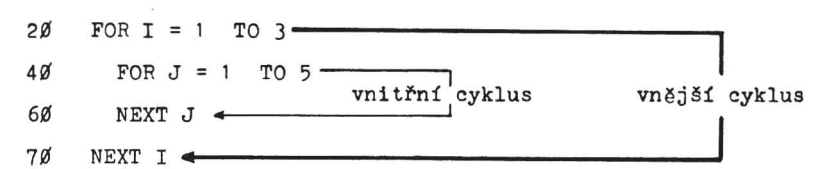
Příklad 72: sestavme program, který by demonstroval průběh vnitřního /řídící proměnná J/ a vnějšího /řídící proměnná I/ cyklu. Analýza úkolu: je třeba sestavit vnější a vnitřní cyklus. Činnost jednotlivých cyklů lze patrně nejlépe demonstrovat zobrazením hodnot jednotlivých řídících proměnných. Změnu vnější řídící proměnné navíc vyjádříme dvěma prázdnými řádky.

PROG-34 NEW CR

```
1Ø PRINT "VNEJSI A VNITRNI CYKLUS": WND 3    hlavička, "okno";
2Ø   FOR I = 1  TO 3                 parametry vnějšího cyklu;
3Ø   PRINT: PRINT                    vytvoření dvou prázdných řádků;
4Ø     FOR J = 1  TO 5               parametry vnitřního cyklu;
5Ø     PRINT "I="; I;  "J="; J       zobrazení řídících proměnných;
6Ø     NEXT J                        konec vnitřního cyklu;
7Ø   NEXT I                          konec vnějšího cyklu;
8Ø PRINT "KONEC"                     konec programu.
```

Navržený program je opticky rozčleněn na běžné programové řádky, na programové řádky vnějšího cyklu a na programové řádky vnitřního cyklu.

Příklad 73: graficky znázorněme vnější a vnitřní cyklus z PROG-34.



Program PROG-34 je sestaven správně. Koncový bod vnějšího cyklu NEXT I leží až za koncovým bodem vnitřního cyklu NEXT J .

Příklad 74: sestavme program pro hru "Cestovatelská: až pojedu na prázdniny, vezmu si sebou ...".
Postupně se přidávají jednotlivé předměty a kdo je nedokáže přesně a ve správném pořadí vyjmenovat, prohrává.
Analýza úkolu: jednotlivé předměty budeme vkládat do pole "A$(I)".
Počet předmětů omezíme na 10. Pole "A$(I)" využijeme dvakrát:
- při vkládání dalšího předmětu INPUT A$(I),
- při porovnávání odpovědi K$ po každém vloženém předmětu A$(J)

Do programu musíme vložit 2 cykly: vnější pro postupné vkládání jednotlivých předmětů a vnitřní pro porovnávání správnosti odpovědi.

```
PROG-35    NEW  CR
  5 PRINT "CESTOVATELSKA": WND 2     název programu, "okno";
 10  FOR I = 1  TO 10                parametry vnějšího cyklu;
 20  PRINT "VLOZ"; I; " PREDMET"     nápověda;
 30  INPUT A$(I): CLS                vložení názvu předmětu; výmaz;
 40    FOR J = 1  TO I               parametry vnitřního cyklu;
 50    PRINT "OPAKUJ"; J; "PREDMET"  nápověda;
 60    INPUT K$                      vkládání názvů předmětů;
 70    IF K$ = A$(J)  THEN 100       při správné odpovědi;
 80    BEEP 7,40: PRINT "CHYBA -     při chybné odpovědi signalizace,
       SPRAVNE: "; A$(J)             zobrazení správné odpovědi;
 90    INPUT Q$: RUN                 přerušení /vložíme CR/, znova;
100    NEXT J                        konec vnitřního cyklu;
110 PRINT: NEXT I                    prázdný ř., konec vnějšího cyklu;
120 PRINT "MAS DOBROU PAMET": END    ukončení hry po vyjmenování
                                     všech 10 vložených předmětů.
```

Poznámka: "okno" na obrazovce zrušíme vložením WND 0 CR .

5.9 Přiřazování konstant ze seznamu DATA

Za příkaz DATA /čti "dejta"/ lze do programového řádku vložit číselné nebo řetězcové konstanty. Příkazem RESTORE /"ristór"/ určujeme, od kterého programového řádku mají být tyto konstanty vyvolávány.

Jednotlivé konstanty /data/ ze seznamu vyvoláváme příkazem READ . Při prvním použití příkazu READ se vyvolá první konstanta, při druhém použití tohoto příkazu se vyvolá další konstanta, atd.

Příklad 75: sestavme program pro zkoušení žáků /test/ z názvů hlavních měst jednotlivých států.

Analýza úkolu: dvojice řetězcových proměnných "stát", "město" budou tvořit jednotlivé programové řádky seznamu DATA. Seznam DATA ukončíme znakem "K".

```
PROG-36    NEW   CR
  1Ø PRINT "HLAVNI MESTA": WND 2    název programu, okno;
  2Ø DATA "CSSR", "PRAHA"           první dvojice "stát", "město";
  22 DATA "SSSR"; "MOSKVA"          druhá dvojice;
  24 DATA "NDR", "BERLIN"           třetí dvojice, atd.;

  44 DATA "K"                       poslední prvek v seznamu DATA;
  5Ø RESTORE 2Ø                     data je třeba číst od ř. 2Ø;
  6Ø READ A$          přečte se konstanta a přiřadí se do A$;
  7Ø IF A$ = "K" THEN END           ukončení programu při A$ = "K";
  8Ø PRINT A$: READ B$ zobrazení, další konstanta se přiřadí do B$;
  9Ø INPUT "HLAVNI MESTO"; C$       nápověda - odpověď žáka;
  1ØØ IF C$ = B$ THEN PRINT: GOTO 6Ø   při správné odpovědi;
  11Ø BEEP 7,5Ø: PRINT "CHYBA"      signalizace při chybné odpovědi;
  12Ø PRINT "SPRAVNA ODPOVED";. B$  zobrazení správné odpovědi;
  13Ø PRINT: GOTO 6Ø                další dvojice "stát", "město".
```

Jestliže v jednom programovém řádku je více konstant typu "data",

oddělují se jednotlivé konstanty od sebe oddělovačem "," . Na konci programového řádku s konstantami "data" se oddělovač neuvádí.

Jestliže je třeba v programu vyvolávat různé soubory konstant, například první soubor pro program "Hlavní města", druhý soubor pro "Chemické značky" apod., m u s í m e příkazem RESTORE čís. řádku určit, od kterého programového řádku mají být "data" vyvolávána.

> Programování osobních počítačů není složité.
> Stačí jen
> - znát nebo vymyslet postup řešení,
> - vědět, který povel nebo příkaz kdy a jak použít,
> - správně sestavit program a vložit jej do počítače,
> - program spustit.

6. Základy grafického zobrazování

Grafické zobrazování vychází z obdobných zásad jako zobrazování /kreslení/ na papíru. Počátek grafického zobrazování je v l e v é m s p o d n í m r o h u obrazovky. Počítač dokáže zobrazit libovolný bod na obrazovce jen tehdy, jestliže počítači sdělíme souřadnice tohoto bodu.

Abychom mohli tyto souřadnice počítači sdělit, musíme znát měřítko, které se používá v základním grafickém režimu pro udání místa na obrazovce:

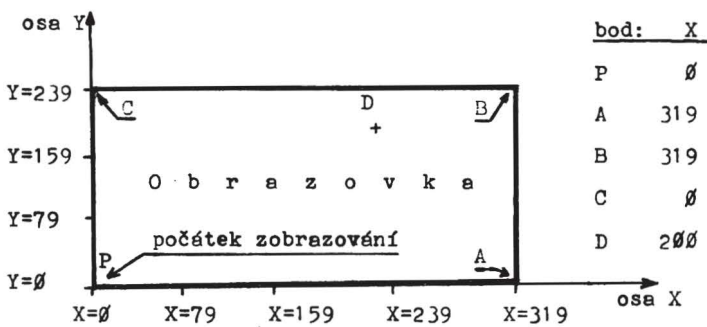
- vodorovná osa /osa X/ se na obrazovce dělí na 320 bodů,
- svislá osa /osa Y/ se na obrazovce dělí na 240 bodů.

Zjišťování souřadnic na obrazovce si můžeme představit třeba tak, že papírovým měřítkem, které má na délku obrazovky 320 bodů, odměřujeme vodorovnou a svislou vzdálenost určitého bodu od počátku zobrazování. Každý bod je pak určen dvěma souřadnicemi:
naměřená hodnota v ose X, naměřená hodnota v ose Y.

Souřadnici bodu P, který je v počátku zobrazování, zapíšeme takto:

Ø, Ø
- "naměřená" hodnota v ose Y,
- "naměřená" hodnota v ose X .

Počítači "je jedno", jakým způsobem tento bod označujeme: pouze přesune g r a f i c k ý k u r z o r do místa, které je určeno těmito souřadnicemi. Princip zobrazování v grafickém režimu je uveden na obrázku:



souřadnice:

bod:	X	Y
P	Ø	Ø
A	319	Ø
B	319	239
C	Ø	239
D	2ØØ	19Ø

Základním příkazem pro přesun grafického kurzoru do určitého místa je příkaz MOVE X,Y /čti "mův"/:

MOVE Ø,Ø CR přesunutí kurzoru do počátku zobrazování;

MOVE 319,239 CR přesunutí kurzoru do místa označeného bodem B,

MOVE 2ØØ,19Ø CR přesunutí kurzoru do místa označeného bodem D.

Počítač nekontroluje, zda není překročena nejvyšší možná hodnota souřadnic: 319 v ose X, 239 v ose Y.

Zobrazí se na obrazovce bod o souřadnicích 239,319 /MOVE 239,319/?

Hodnota X vyhovuje, X je menší než 319,

Y nevyhovuje, Y je větší než 239.

Příkazem MOVE 239,319 jsme zadali bod, který "je nad obrazovkou".

Po vložení příkazu MOVE X,Y se na obrazovce nic neukáže. Počítač si jen zapamatuje polohu bodu o souřadnicích X,Y.

6.1 Spojování bodů čarami

Činnost počítače při spojování bodů určených souřadnicemi X,Y lze přirovnat k činnosti řidiče vozu TAXI-služby:

a/ Řidiče vozu zastavíme na ulici a nasedneme. Řidič z taxametru vymaže údaje o předchozí jízdě; my vymažeme obrazovku a grafickou paměť počítače:

CLS: CLSG CR /CLSG je příkaz k vymazání grafické paměti/;

b/ Taxíkář nastaví na taxametru počátek jízdy. My nastavíme počátek grafického zobrazování:

MOVE Ø,Ø CR

c/ Taxíkáři sdělíme adresu, kam nás má dovézt. Počítači sdělíme souřadnice bodu, do něhož má provést čáru z bodu, kde se n y n í nachází grafický kurzor, například do bodu A /319,Ø/:

PLOT 319,Ø CR vytvoří se čára z bodu Ø,Ø do bodu 319,Ø

d/ Taxíkáři sdělíme další adresu. Počítači sdělíme souřadnice dalšího bodu, například bodu D /2ØØ,19Ø/:

PLOT 2ØØ,19Ø CR čára z bodu 319,Ø do bodu 2ØØ,19Ø.

Jestliže za souřadnice příkazu PLOT vložíme další číslo, zobrazí se takový druh čáry, který odpovídá tomuto třetímu číslu /parametru čáry/. Pokud je tímto parametrem Ø, nezobrazí se žádná čára; při vložení čísla 255 se zobrazí plná čára. Čísla od 1 do 254 vytvářejí různě přerušované čáry: tečkovanou, čárkovanou apod.

Příklad 76: sestavme program pro ukázku různých druhů čar.

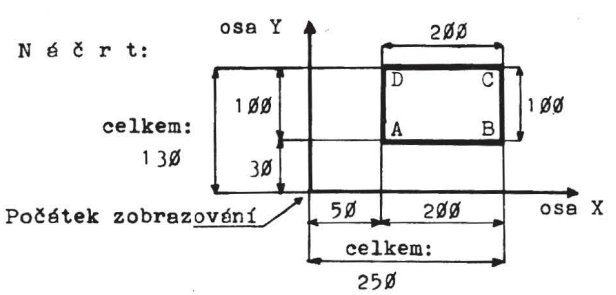
```
PROG-37     NEW   CR
 1Ø FOR I = Ø   TO 256: CLS: MOVE Ø,Ø   nastavení cyklu, počátku;
 2Ø PRINT "PARAMETR CARY:"; I           zobrazení parametru čáry;
 3Ø PLOT 3ØØ,Ø, I                  zobrazení čáry určené parametrem;
 4Ø INPUT QØ: NEXT I               přerušení, opakování cyklu.
```

Program PROG-37 spustíme povelem RUN a řídíme jej opakovaným vkládáním CR . U druhů čar, které nás zaujaly, si zaznamenáme jejich parametr. Program přerušíme vložením CTRL C .

Příklad 77: nakresleme na obrazovku televizoru obdélník o stranách AB = 2ØØ bodů základního souřadnicového systému /základního rastru na obrazovce/, AD = 1ØØ bodů tohoto rastru.
Vrchol A obdélníka je od počátku grafického zobrazování posunut v ose X o 5Ø bodů rastru, v ose Y o 3Ø bodů rastru.

Analýza úkolu: nejdříve si na papír nakreslíme /stačí jen náčrt/ osu X a osu Y. Mezi tyto osy zakreslíme obdélník, tak aby alespoň přibližně splňoval požadavky zadání:

- posunutí vrcholu A: v ose X o 5Ø bodů rastru /X=5Ø/;
  v ose Y o 3Ø bodů rastru /Y=3Ø/;
- vzdálenost AB = CD = 1ØØ bodů rastru,
- vzdálenost AD = BC = 1ØØ bodů rastru.



Výpočet souřadnic:

A: X=5Ø Y=3Ø
B: X=5Ø+2ØØ Y=3Ø
C: X=5Ø+2ØØ Y=3Ø+1ØØ
D: X=5Ø Y=3Ø+1ØØ

Souřadnice jednotlivých vrcholů vypočítáme tak, že určíme jejich vzdálenost od počátku zobrazování jak v ose X, tak v ose Y.
Vlastní zobrazení můžeme provést v dialogovém nebo v programovém režimu počítače:

Dialogově: CLS: MOVE 5Ø,3Ø: PLOT 25Ø,3Ø; 25Ø,13Ø; 5Ø,13Ø; 5Ø,3Ø CR

Programově: NEW CR

```
1Ø CLS: MOVE 5Ø,3Ø: PLOT 25Ø,3Ø; 25Ø,13Ø; 5Ø,13Ø; 5Ø,3Ø
```

Souřadnice za příkazem PLOT lze zadávat i parametricky, tedy dosazením do proměnných za příkazy MOVE a PLOT.

Příklad 78: sestavme parametrický program pro zobrazení obdélníka. Do programu zařaďme i signalizaci případného překročení nejvyšších možných hodnot souřadnic X a Y.

Analýza úkolu: jedná se vlastně jen o obecné zadání příkladu 77. Budeme proto postupovat obdobným způsobem. V požadovaném testu budeme zjišťovat, zda souřadnice X nepřekročila hodnotu 319 a souřadnice Y hodnotu 239.

PROG-38 NEW CR

```
1Ø PRINT "PARAMETRICKY OBDELNIK": WND 2    hlavička, okno;
2Ø INPUT "VLOZ POCATEK  X,Y"; MX, MY       parametrické určení poč.;
3Ø INPUT "VLOZ DELKU AB"; AB               vložení délky AB;
4Ø INPUT "VLOZ VYSKU AD"; AD               vložení výšky AD;
5Ø CLS: MOVE MX, MY                  vymazání obrazovky, přesun kurzoru;
6Ø BX = MX + AB:  BY = MY                  výpočet souřadnic bodu B;
7Ø CX = BX:  CY = MY + AD                  výpočet souřadnic bodu C;
8Ø DX = MX:  DY = CY                       výpočet souřadnic bodu D;
9Ø IF BX>319  OR  CY>239  THEN BEEP  signalizace chyby - překro-
   7,5Ø: PRINT "CHYBNE ZADANI - OBRAZ  čení nejvyšších možných X,Y
   JE MIMO OBRAZOVKU": PRINT: RUN          opakování programu;
1ØØ PLOT BX,BY; CX,CY; DX,DY; MX,MY        zobrazení čar.
```

6.2 Změna základního měřítka

V některých případech je zobrazování v základním rastru 32Ø krát 24Ø bodů zbytečně podrobné. Základní měřítko 32Ø krát 24Ø bodů lze změnit na jiné měřítko příkazem SCALE , za nímž následují krajní hodnoty na ose X a krajní hodnoty na ose Y:
SCALE Xmin, Xmax, Ymin, Ymax .

Příklad 79: změníme měřítko v ose X z 32Ø bodů na 10 dílků a v ose Y z 24Ø bodů na 8 dílků.

Analýze úkolu: pro změnu měřítka použijeme příkaz SCALE /"skejl"/

SCALE Xmin, Xmax, Ymin, Ymax tedy:

SCALE Ø, 1Ø, Ø, 8 CR

Příklad 8Ø: v novém měřítku je třeba zobrazit obdélník o stranách AB = 5 dílků, AD = 3 dílky. Vrchol A bude posunut o 2 dílky od osy Y a o 1 dílek od osy X.

Analýza úkolu: souřadnice vrcholu A obdélníka: X=2, Y=1; souřadnice ostatních vrcholů zjistíme stejným způsobem jako v př. 77, resp v př. 78.

MOVE 2,1 CR grafický kurzor přesuneme do vrcholu A;

PLOT 7,1; 7,4; 2,4; 2,1 CR zobrazení obdélníka.

Poznámka: do Xmin a do Ymin nemusí být vždy vkládána hodnota nula, ale lze do nich vložit i jiné celé /kladné nebo záporné/ číslo.

6.3 Zobrazení souřadnicových os

Souřadnicové osy již umíme zobrazit příkazem PLOT , za nějž vložíme jednotlivé souřadnice příslušných bodů.

Jednodušší je však použití příkazu AXES /čti "exis"/, který přímo zobrazí osu X a osu Y. Tyto osy se zobrazí i tehdy, když jsou posunuty od počátku zobrazování.

AXES X, Y, p X je vodorovná a Y je svislá vzdálenost obou os od počátku zobrazování; číslo p určuje druh čáry.

Příklad 80: plnou čarou zobrazme základní souřadnicové osy X a Y. Tečkovaně /p=1/ zobrazme pomocné osy vzdálené 3Ø bodů od osy X a 5Ø bodů od osy Y v základním grafickém rastru.

SCALE Ø, 319, Ø, 239 CR "přepnutí" do základního rastru,

AXES Ø,Ø, 255: AXES 5Ø, 3Ø,1 CR zobrazení hlavních a pomocných os.

6.4 Generátor grafických znaků

Generátor grafických znaků umožňuje zobrazovat písmena, číslice a znaky i v jiném, než v základním tvaru:

LABEL S, V; "text" parametr S určuje, kolikrát má být text širší, parametr V kolikrát má být text vyšší, než je základní velikost písmen a číslic.
/čti "lejbl"/

Zobrazování příkazem LABEL z a č í n á od místa, do něhož je právě nastaven grafický kurzor.

Příklad 81: od bodu určeného souřadnicemi X =18Ø a Y=1ØØ je třeba zobrazit text "ONDRA" ve trojnásobné velikosti oproti základní velikosti znaků.

Řešení: nejdříve přesuneme kurzor na určené místo:

MOVE 18Ø, 1ØØ CR — přesunutí kurzoru do určeného bodu;
LABEL 3,3;"ONDRA" CR — zobrazení textu ve trojnásobné velikosti /trojnásobná výška i šířka/.

6.5 Zjištění souřadnic místa, na němž je kurzor

Po ukončení zobrazování textu "ONDRA" v předchozím příkladu zůstal grafický kurzor za slovem ONDRA. Jeho a b s o l u t n í souřadnice zjistíme příkazem GET X, GET Y :

PRINT "X="; GET X — zobrazí se: X= 285
PRINT "Y="; GET Y — zobrazí se: Y= 1ØØ

Číselné údaje představují vzdálenost grafického kurzoru od počátku zobrazování, který je v levém spodním rohu obrazovky.

6.6 Vytvoření ucelené plochy na obrazovce a vložení textu

Plochu pro výtvarné odlišení určitého nápisu, otázky apod. lze vytvořit příkazem „FILL /čti "fil"/:

FILL MX, MY, P MX určuje délku vytvářené plochy, MY výšku vytvářené plochy. Parametr P určuje charakter plochy.

Pokud se parametr P rovná číslu 255, vytvoří se ucelená bílá plocha; při P = 1 zobrazí se 2 rovnoběžky, atd.

P o č á t e k vytvářené plochy /její levý spodní roh/ je určen souřadnicemi za příkazem MOVE .

Příklad 82: souvislá bílá plocha začíná v bodu o souřadnicích X=4Ø, Y=6Ø. Délka činí 6Ø bodů v ose X, výška plochy činí 2Ø bodů v ose Y. Do této plochy je třeba vložit nápis "ONDRA" ve dvojnásobné velikosti.

Analýza úkolu: nejdříve je třeba vypočítat souřadnice pro příkaz FILL : MX = 4Ø /počátek zobrazování/ + 6Ø /délka plochy/ = 1ØØ;
MY = 6Ø /počátek zobrazování/ + 2Ø /výška plochy/ = 8Ø.

Protože není přesně určeno, od kterého bodu má být zobrazen text "ONDRA", zvolíme pro počátek zobrazování souřadnice X=6Ø, Y=7Ø.

```
PROG-39     NEW   CR
  1Ø CLSG: MOVE 4Ø,6Ø     vymazání obrazovky a grafické paměti, poč.;
  2Ø FILL 1ØØ, 8Ø, 255    zobrazení ucelené bílé plochy;
  3Ø MOVE 6Ø,7Ø           přesunutí grafického kurzoru do místa, od-
                          kud má začít zobrazování textu "ONDRA";
  4Ø LABEL 2,2;"ONDRA"    dvojnásobné zvětšení textu "ONDRA".
```

Program lze samozřejmě upravit i pro parametrické vkládání vstupních údajů. Parametrický program je velmi výhodný při počátečních úvahách o zaplnění obrazovky /velikost a charakter plochy, vhodný nápis a jeho umístění, apod./.

6.7 Zjištění souřadnic určitého místa na obrazovce

Při práci v grafickém režimu je třeba často znát souřadnice bodu, od něhož se má začít vytvářet ucelená plocha, začít zobrazovat čára nebo text, atd. Zjišťování těchto souřadnic odhadem nebo přepočítáváním řádků a sloupců na grafický režim je zdlouhavé a velmi pracné.

Příklad 83: sestavme program pro zjištění libovolného místa na obrazovce.

Analýza úkolu: určité místo na obrazovce si můžeme "zviditelnit" tak, že na ně najedeme grafickým kurzorem /příkaz DRAW INPUT/. Na toto místo vložíme písmeno H, které má výrazný tvar. Pro zjištění souřadnic písmene H použijeme příkazy GET X a GET Y.

```
PROG-4Ø      NEW   CR
  1Ø CLS: CLSG: PRINT "SOURADNICE"   vymazání obrazovky a gr. paměti;
  2Ø DRAW INPUT A$: LABEL 1,1;"H"    "najetí " kurzorem, písmeno H;
  3Ø PRINT "X="; GET X; " Y="; GET Y zobrazení souřadnic konce
                                     písmene H;
  4Ø  RUN 2Ø                         opakované spuštění programu.
```

Program spustíme povelem RUN. Grafický kurzor dopravíme stisknutím tlačítek se šipkami do požadovaného místa na obrazovce. Po vložení CR se na určeném místě zobrazí písmeno H a pak i souřadnice. /Případným vložením "V" lze jednotlivá místa propojovat čarami/.

6.8 Kombinované grafické příkazy

Využívání kombinovaných grafických příkazů výrazně usnadňuje práci v grafickém režimu - ne všechny osobní počítače jsou těmito příkazy vybaveny!

6.8.1 Automatické vkládání souřadnic do řetězcové proměnné

DRAW INPUT A$ tlačítky pro ovládání grafického kurzoru dopravíme kurzor na požadované místo na obrazovce. Pak stiskneme tlačítko s písmenem "V" nebo s písmenem "S".

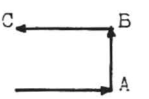
Stisknutím tlačítka "V" se zobrazí čára spojující současné a předchozí umístění grafického kurzoru.

Stisknutím tlačítka "S" se čára nevytvoří, jen se zaznamenají souřadnice bodu, v němž bylo stisknuto tlačítko "S".

Souřadnice míst, na nichž bylo stisknuto tlačítko "V" nebo tlačítko "S", se ukládají do proměnné A$. Zobrazit tyto souřadnice můžeme příkazem PRINT A$ CR .

Písmeno S před souřadnicí X znamená, že na tomto místě bylo stisknuto tlačítko "S". Písmeno V znamená, že bylo stisknuto tlačítko "V". Písmeno M určuje souřadnice počátku ukládání do A$.

Příklad 84: zjistěte a vysvětlete souřadnice lomené čáry typu




```
PROG-41    NEW  CR
  1Ø CLS: MOVE Ø,Ø          vymazání obrazovky, počátek zobrazov.;
  2Ø DRAW INPUT A$          kombinovaný grafický příkaz;
  3Ø PRINT A$               zobrazení souřadnic X a Y vložených bodů;
  4Ø RUN                    opakované spuštění programu.
```

Klávesnici přepneme do režimu "velká písmena". Tlačítky pro pohyb grafického kurzoru a opakovaným stisknutím tlačítka "V" nakreslíme požadovanou čáru. Po vložení CR se například zobrazí:

```
  V61,Ø: VØ,46: V-61,Ø
```

V61,Ø znamená, že v prvním vrcholu bylo stisknuto tlačítko "V". souřadnice tohoto prvního bodu jsou: X=61, Y=Ø.

VØ,46 znamená, že při dalším stisknutí tlačítka "V" se souřadnice
X oproti předchozímu bodu A změnila ve vrcholu B o Ø bodů základního grafického rastru,
Y oproti předchozí souřadnici Y změnila o 46 bodů rastru.

V-61,Ø znamená, že při třetím stisknutí tlačítka "V" se souřadnice
X změnila v o p a č n é m smyslu /znaménko"-"/ o 61 bodů rastru,
Y změnile oproti předchozí souřadnici Y o Ø bodů rastru.

Souřadnice X a Y, ukládané v průběhu příkazu DRAW INPUT do řetězcové proměnné n e j s o u zjišťovány od počátku zobrazování. Jsou zjišťovány od p ř e d c h o z í polohy grafického kurzoru.

Souřadnice uložené v řetězcové proměnné jsou **r e l a t i v n í .** Znaménko "-" před písmenem V nebo S znamená opačný směr čáry: "minus" před souřadnicí X znamená posun k levému okraji obrazovky, "minus" před souřadnicí Y znamená posun k jejímu spodnímu okraji.

A b s o l u t n í souřadnice jednotlivých bodů získáme příkazy

PRINT GET X: PRINT GET Y: CR

6.8.2 Vyvolání obrazců uložených v řetězcové proměnné

Obrazec vložený do řetězcové proměnné příkazem DRAW INPUT můžeme z této řetězcové proměnné opět vyvolat na obrazovku příkazem DRAW"XA$" /písmeno za první uvozovkou je velké písmeno "X"/.
U p o z o r n ě n í : programový řádek obsahující příkaz DRAW"XA$" nesmíme spouštět povelem RUN , neboť tímto povelem bychom vymazali obsah řetězcové proměnné, v níž jsou uloženy souřadnice X a Y.

Jestliže obrazec nakreslený příkazem DRAW INPUT A$ měl tolik rohů, že bylo třeba doplnit řetězcovou proměnnou A$ o další řetězcovou proměnnou, musíme o tuto další řetězcovou proměnnou doplnit i příkaz DRAW .

Příklad 85: sestavme program využívající úplný kombinovaný grafický příkaz. Souřadnice jednotlivých bodů se budou vkládat do řetězcových proměnných A$ a U$. Po dokončení kresby je třeba obrazovku vymazat a po krátké přestávce opět na obrazovku vyvolat původní obrazec.

```
PROG-42    NEW  CR
  1Ø MOVE Ø,Ø: DRAW INPUT A$, U$   počátek zobrazování, vkládání;
  2Ø CLS: PRINT "VYMAZANI OBRAZOVKY"   vymazání obrazovky, text;
  3Ø BEEP Ø,2ØØ: CLS: MOVE Ø,Ø   programovaná pouza, počátek;
  4Ø DRAW"XA$:XU$": BEEP Ø,2ØØ:RUN vyvolání obrazce, spuštění.
```

Program spustíme vložením RUN ; přerušíme jej vložením CTRL C .

7 Magnetofon jako vnější paměť počítače

Po vypnutí počítače se všechny programy i další údaje z počítače ztratí. Je proto vhodné nahrávat si jednotlivé programy na kazetový /nebo i cívkový/ magnetofon. Činnost počítače při spolupráci s magnetofonem je řízena systémovým programem MIKOS a zahrnuje především

- inicializaci kazety,
- zavádění programů z magnetofonu do počítače,
- nahrávání programů z počítače na magnetofon.

Dále uváděný postup se vztahuje k magnetofonům, které nejsou vybaveny dálkovým spouštěním a zastavováním motorku magnetofonu, takže u nich musíme řídit jejich chod sami.

U typů s dálkovým ovládáním stačí po zobrazení

cteni - hotovo? nebo zapis! hotovo?

nastavit příslušnou funkci na magnetofonu. Vlastní chod motorku magnetofonu řídí relé umístěné v počítači ONDRA.

> Při dobré znalosti programování toho dokážeme i na levném počítači vždy mnohem více, než bez těchto znalostí na nejdokonalejším osobním počítači.

7.1 Inicializace kazety

Inicializace kazety není nic jiného, než poznamenání určitého textu na kazetu /magnetofonovou pásku/. Tento text není součástí programu, ale počítač jej dokáže zobrazit. Text může obsahovat až 64 znaků. Libovolný program lze na magnetofon nahrát i bez inicializace kazety.

Textem může být příjmení majitele kazety, autora programu, obsah určité nahrávky /např. BASIC.1/, nebo označení celé kazety /PRACOVNI KAZETA - MATEMATIKA/.

Inicializaci kazety je možno provádět ještě před zavedením BASIC do počítače, nebo při zavedeném BASIC a libovolném vloženém /zavedeném/ programu.

Při zavedeném BASIC je třeba před inicializací vyvolat základní hlášení operačního systému "._" a to buď vložením BYE CR nebo stisknutím červeného tlačítka na levém boku počítače.

Obrazovka:	naše činnost:
._	vložíme písmena KI
previn - zapni zapis! hotovo?	kazetu přetočíme na místo, od něhož má být nahrána poznámka; vložíme CR;
zadej nazev	vložíme název, například PRACOVNI, na magnetofonu spustíme funkci "nahrávání", do počítače vložíme CR .

Asi za 20 sekund obrazovka ztemní, poznámka se nahraje na magnetofonovou kazetu nebo pásku a obrazovka se opět rozjasní. Je-li třeba přejít do již zavedeného BASIC, vložíme do počítače písmeno B, jinak do počítače zavedeme BASIC.

7.2 Vytvoření pracovní kopie BASIC

Nespoléhejme na to, že původní nahrávka BASIC nám vydrží po celou dobu využívání počítače - vytvořme si proto i kopii BASIC.

BASIC na kazetu nahráváme jiným způsobem než běžné programy. Musíme použít příkazů operačního systému počítače

KA pro vytvoření názvu programu a jeho nahrání,

KS pro výpis obsahu paměti počítače na kazetu.

Dále musíme do počítače vložit počáteční a koncovou adresu nahrávaného souboru:

- u původní verze počítače ONDRA: 44ØØmezera6FFF
- u počítače ONDRA SPO 186: 1ØØØmezera3FFF

Stisknutí mezerníku označíme znakem $_{v}$.

O b r a z o v k a :	naše činnost:
READY	vložíme: BYE CR
._	vložíme: KA
zapis! hotovo?	kazetu nastavíme na místo, od něhož má být nahrán BASIC; vložíme CR
novy nazev	vložíme název programu /až 11 znaků/, např. BASIC.1 /= první kopie BASIC/, na magnetofonu nastavíme funkci "nahrávání", do počítače vložíme CR

Na kazetu se nahraje název programu, tedy BASIC.1 .

._	vložíme: KS
=_	první verze ONDRA: vložíme $44ØØ_{v}6FFF$ ONDRA SPO 186: vložíme $1ØØØ_{v}3FFF$ /znak $_{v}$ znamená stisknutí mezerníku/; na magnetofonu spustíme funkci "nahrávání"; do počítače vložíme: CR

BASIC se nahraje na kazetu /pásku/ v magnetofonu.

Po ukončení nahrávky buď celý postup opakujeme a vytvoříme si další kopii BASIC nebo vložíme písmeno B /"vyvoláme BASIC"/.

Poznámka: pokud při vkládání uvedených symbolů uděláme chybu, nic se neděje. Stiskneme červené tlačítko na boku počítače /případně přetočíme kazetu o kousek zpět/ a celou činnost opakujeme.

Při spolupráci počítače a magnetofonu m u s í m e nejdříve spustit ten přístroj, do něhož programy /data/ v k l á d á m e !

7.3 Nahrání programu z počítače na magnetofon

Existuje více způsobů nahrávání programu z počítače na magnetofon. Všechny způsoby jsou popsány v příručkách dodávaných výrobcem počítače ONDRA. Pro naše potřeby postačí základní způsob.

Pásku /kazetu/ v magnetofonu nastavíme na místo, od něhož má být program nahrán.

Do počítače vložíme: SAVE B "@ N A Z E V" CR

slovo "N A Z E V" znamená skutečný název programu, například MATEMATIKA-1 nebo SKOLNI nebo PRISTANI NA MESICI atd.

zobrazí se:

zapis! hotovo? magnetofon přepneme do funkce "nahrávání"; do počítače vložíme CR

Po ukončení nahrávání se na obrazovce objeví hlášení **READY** na magnetofonu vypneme funkci "nahrávání".

Poznámky: v příkazu pro nahrání programu písmeno B znamená "binární tvar nahrávky". Znak @ znamená připojení magnetofonu k výstupu z počítače. Znak @ a název programu musí být v uvozovkách.

7.4 Zavádění programu z magnetofonu do počítače

Základní způsob zavádění programů do počítače předpokládá, že v počítači již je zaveden BASIC. Magnetickou pásku nastavíme před místo, od něhož je na pásce nahrán program.

Do počítače vložíme: LOAD B "NAZEV" CR

slovo NAZEV znamená skutečný název programu, například "MATEMATIKA" nebo "SKOLNI" nebo "PRISTANI NA MESICI" atd.

zobrazí se: **cteni - hotovo?** vložíme CR

Na magnetofonu spustíme funkci "přehrávání".

Nejčastější obtíže při zavádění programů do počítače a způsob jejich odstranění je popsán v části 1.3 této příručky.

Další způsoby zavádění programů z magnetofonu do počítače jsou popsány v příručkách dodávaných výrobcem počítače ONDRA.

8. Knihovny programů a podprogramů

Je zbytečné znova a znova si vymýšlet již ověřené programy a podprogramy. Zapisujme si je proto do zvláštního nejvhodnější je "čtverečkovaný" sešit, v němž na levé polovině je vlastní program, na pravé polovině je uveden příslušný komentář; tedy tak, jak je postupováno v této příručce.

8.1 Knihovna programů vedená na kazetě

Často je výhodné mít v jednom uceleném záznamu /programu/ na kazetě více dílčích programů nebo podprogramů, například z určitého oboru. Pak jen volbou z nabídky programů vybereme ten, který právě potřebujeme. Princip je jednoduchý a byl již v této příručce popsán, proto si jej jen připomeneme:

```
1Ø PRINT "1 = název prvního programu"
2Ø PRINT "2 = název druhého programu", atd.
5Ø INPUT "VLOZ CISLO ZVOLENEHO PROGRAMU"; A
6Ø ON A GOTO počáteční řádek prvního programu, druhého programu,
   atd.
```

Postupujeme tak, že základní program, který chceme doplnit o další program, zavedeme do počítače a základní program ukončíme povelem RUN /nebo GOTO 1Ø/. Další program vložíme za základní program. Doplníme nabídku programů a programový přepínač. Pak celý program /tedy základní i doplňující/ nahrajeme na kazetu pod jedním společným názvem, například MATEM-TEST .

8.2 Knihovna podprogramů vedená na kazetě

Jestliže programujeme častěji, brzy zjistíme, že převážnou část podprogramů můžeme použít v nejrůznějších programech. Je zbytečné vkládat příslušné podprogramy do počítače podle zápisu v sešitu, když jsou již někde nahrány a prakticky ověřeny. Stačí si

jen vytvořit samostatnou nahrávku se všemi podprogramy. Nahrávku si vytvoříme tak, že

- úplný program, tedy program obsahující i podprogramy, o které máme zájem, si nahrajeme na kazetu s běžnými programy;
- opakovaným povelem DELETE M,N z paměti počítače vymažeme všechny programové řádky, o které nemáme zájem, takže v paměti počítače zůstanou jen jednotlivé podprogramy;
- povelem SAVE B "/@ PODPROGRAM" CR si tyto podprogramy nahrajeme na příslušnou kazetu.

Před sestavováním dalších programů si do počítače nejdříve zavedeme BASIC a pak záznam /nahrávku/ podprogramů. Teprve potom začneme sestavovat nový program, v němž využíváme některé z nahraných podprogramů.

Po dokončení nového programu a ověření jeho správné funkce si můžeme vytvořit další /rozšířenou/ nahrávku podprogramů.

Jestliže budeme mít v našich programech více podprogramů, než jich skutečně využíváme, nevadí. Jen se nepatrně prodlouží doba jejich nahrávání /zavádění/. I nadbytečné podprogramy však můžeme vymazat povelem DELETE .

Pro usnadnění práce si jednotlivé podprogramy číslujme od vysokého čísla, například od 5000. Vytvoříme si tak potřebnou rezervu v číslech řádků jak pro knihovnu programů, tak pro knihovnu podprogramů.

S osobními počítači je to obdobné jako s osobními automobily. Jestliže v autoškole získáme řidičský průkaz na "škodovce", dokážeme -po seznámení se s jejich odlišnostmi- řídit i jiné automobily. Rozhodující jsou jen naše znalosti.

9 Rejstřík základních klíčových slov a instrukcí

Rejstřík používáme tehdy, jestliže nás zajímá konkrétní využití určitého příkazu nebo povelu. V rejstříku jsou v abecedním pořadí uváděna jednotlivá klíčová slova, jejich výslovnost a stránka, na níž je uveden základní způsob jejich využití.

klíčové slovo		str.:
ON GOTO	/on goutů/	51
OR	/ór/	60
PLOT	/plot/	77
PRINT	/print/	17, 45
READ	/ríd/	74
READY	/redy/	10
RENUM	/rinem/	35
RETURN	/ritérn/	57
RESTORE	/ristor/	74
REPEAT	/ripít/	70
RIGHT$	/rajt-string/	25, 54
RND	/erende/	22
RUN	/ran/	29, 30
SAVE	/sejv/	89
SGN	/sajn/	22
SCALE	/skejl/	79
SEEK	/sík/	36
SIN	/sinus/	26
SPC	/spejs/	46
SQR	/esqér/	22
STOP	/stop/	31
STEP	/step/	68
STR$	/str-string/	26
TAB	/tab/	46
TAN	/tan/	26
THEN	/dzen/	48
TO	/tů/	68
TROFF	/trof/	36
TRON	/tron/	36

klíčové slovo		str.:
TRSTEP	/trstep/	36
UNTIL	/antil/	70
USING	/jůsing/	46
VAL	/val/	25
WEND	/vend/	70
WHILE	/vajl/	70
WND	/vnd, vindou/	43
XOR	/xor/	60
!	/rem/	60
?	/zkráceně print/	18
@	/edit/	33, 89

Základní instrukce operačního systému:

B	spuštění BASIC
KI	inicializace kazety
KS	zápis binárního souboru
KL	zavedení bin. souboru
L	zkrácené zavádění
CTRL C	přerušení chodu
CTRL D	vymazání znaku
CTRL R	opačné zobrazení .

Příloha č.1

Vkládání některých klíčových slov zkráceným způsobem

BASIC-EXP V5.Ø/G umožňuje vkládat některá klíčová slova zkráceným způsobem. Stačí jen stisknout šipku ⟶ písmeno a vložit CR.

Písmeno:	klíčové slovo:	význam:
A	AUTO	automatické číslování řádků
B	BEG	/začátek bloku/,
C	CLOSE	/uzavření výstupního souboru/,
D	DATA	definice konstant pro příkaz READ,
E	ELSE	součást příkazu IF-THEN-ELSE,
F	FOR	součást příkazu cyklu FOR-TO-STEP/NEXT,
G	GOSUB	příkaz pro vyvolání podprogramu,
H	SEEK	vyhledání zadaného textu v programu,
I	INPUT	vkládání vstupních dat,
J	RUN	spuštění programu,
K	DELETE	zrušení části programu,
L	LIST	výpis programu na obrazovku,
M	MOVE	určení počátku grafického zobrazování,
N	NEXT	součást příkazu cyklu FOR-TO-STEP/NEXT
O	OPEN	/otevření výstupního souboru/,
P	PROC	/definice procedury/,
Q	BYE	přechod z BASIC do MONITORu,
R	RETURN	příkaz pro návrat z podprogramu,
S	STEP	součást cyklu FOR-TO-STEP/NEXT; trasování,
T	THEN	součást podmíněného skoku IF-THEN-ELSE,
U	USING	formátování čísel v příkazu PRINT,
V	FIND	/otevření výstupního souboru/,
X	DRAW	kombinovaný grafický příkaz,
Y	LABEL	zobrazení textu - měřítko: výška, šířka,
Z	PLOT	grafický příkaz pro kreslení čar.

Chybová hlášení

Příloha č. 2

Při zjištění chyby přeruší překladač BASIC chod programu a zobrazí chybové hlášení ve tvaru

ERROR XX IN LINE číslo nebo XX ERROR IN číslo

Slovo ERROR znamená "chyba"; písmena XX určují druh chyby, číslo znamená číslo řádku v němž překladač nalezl chybu.

XX	druh chyby
NF	v cyklu FOR-TO-STEP/NEXT chybí "FOR" k příkazu "NEXT",
SN	syntaktická chyba - špatně zapsaný příkaz /například PRIVT/,
OD	příkaz pro čtení dat READ nenalezl další "data",
FC	chyba v zápisu funkce, například PRINT SQR(-16),
OV	hodnota proměnné je mimo dovolený rozsah /"přetečení"/,
OM	nedostatek místa v paměti /"přetečení" paměti/,
US	neznámý příkaz nebo odkaz na řádek, který není v programu,
BS	chyba v indexování pole - špatný zápis nebo "přetečení" indexu,
DD	vícenásobná /opakovaná/ deklarace stejného pole - např. po RUN,
/Ø	dělení číslem, které má /právě nyní/ hodnotu Ø.
ID	nepřípustný zápis výrazu,
TM	nesouhlas typů proměnných ve výrazu, např. A = "KAREL",
OS	přetečení paměti vyhrazené pro řetězcové proměnné /CLEAR/,
LS	příliš dlouhý řetězec - více než 255 znaků v řetězci,
ST	příliš složitý řetězcový výraz,
CN	chyba v navazování příkazů,
UF	volání funkce, která dosud nebyla definována,
WE	k příkazu WEND chybí odpovídající příkaz WHILE-DO,
UN	k příkazu UNTIL chybí příkaz REPEAT,
M#	příliš mnoho znaků # v masce u příkazu USING,
INPUT ERROR	místo čísla byl vložen řetězec nebo naopak.

BASIC-EXP V5.0/G pro počítač ONDRA SPO 186
(příručka pro začátečníky)
Autor:Ing.Karel Haupt
Určeno pro začínající uživatele počítače ONDRA SPO 186-pro žáky a učitele základních a středních škol.
Schváleno jako příručka pro základní školy a pro střední školy všech typů komisí schvalování počítačových programů dne 16.6.1988.
Vydalo Komenium,n.p.,Praha,pod ČKL 0040/15 v roce 1988.
1.vydání.
Odpovědný redaktor:Ing.Jaroslav Feytis
Náklad: 900 výtisků
K-180/88