

A K T U A L I T Y 1 8

B A S I C Q B A S I C

Obsah:	strana:
Slovo vydavatele	2
Nabídka C2717-PMD: SWD71-Matematika 5.....	3
OBasic -dialog s počítačem a programem.....	5
SCREEN - nastavení typu zobrazení.....	5
COLOR - nastavení barev zobrazení.....	7
PALETTE - změna jedné nebo více barev.....	8
CLS - mazání stínítka nebo okna.....	8
LOCATE - umístění kurzoru na stránce.....	9
PRINT - zobrazení výstupních dat.....	10
SPC() - funkce pro vynechání mezer.....	11
TAB() - funkce pro nastavení kurzoru.....	11
STRING() - pro řetězec stejných znaků.....	11
CHR\$() - vrací znak v ASCII kódu.....	13
ASC() - vrací ASCII kód prvního znaku.....	13
WIDTH - změna počtu znaků na řádku.....	13
VIEW PRINT - vytvoření textového okna.....	13
PRINT USING - formát zobrazení dat.....	14
WRITE - zápis dat.....	15
INPUT - vstup dat nejen z klávesnice.....	15
INPUT\$() - vstup určeného počtu znaků.....	16
INKEY\$ - načtení jednoho znaku	16
Typy dat a proměnných.....	17
Číselné proměnné.....	18
Řetězcové proměnné.....	20
Nabídka PC: Elementární geometrie.....	21
Základní matematika pro ZŠ.....	23
Matematika ZŠ.....	24

Vážená čtenářko,
vážený čtenáři.

Přinášíme slíbené pokračování kursu moderního programovacího jazyka QBasic, který mají prakticky všichni uživatelé PC k dispozici ve svém počítači pod systémem MS DOS 5.0 nebo vyšším "zadarmo". Už jste se také zkusili podívat ?

Tento jazyk umožňuje podobně jako na 8-bitových počítačích nejrychleji znázornit na obrazovce výsledek činnosti jednoho nebo několika příkazů, napsaných na jediném řádku v přímém režimu (Immediate). Úprava takových příkazů je možná okamžitě (bez čekání na překlad překladačem) a odzkoušené řádky lze jednoduše přenést do vytvářeného programu. QBasic nepracuje s číslováním řádků a proto nutí strukturovat program, používat procedury a funkce, sdílet proměnné a pod.

Proto se lze používání a zdokonalování jazyka učit postupně i metodou pokus-chyba, byť to není metoda vhodná a správná, ale snad nejbližší zvědavosti i tvořivosti všech nedávno i dříve narozených.

Protože je QBasic značně rozsáhlý, musím jej přibližovat co nejstručněji. Pokud některé věci budou nesrozumitelné právě pro stručnost, prosím o Vaše připomínky pro připravovanou příručku.

Protože řada učitelů v předmětech informatiky zkoušela algoritmizovat řadu problémů již v Basicu na 8-bitových počítačích, bude pro ně jednoduché přenést tyto miniprogramky do prostředí QBasicu na PC. Máte-li takovou řadu ucelených lekcí na toto nebo i jiné téma, zkusme ji společně vydat pro ostatní - autorská práva Vám zůstanou zachována vedle podílu z prodeje.

Přesto, že počet objednávek na dosavadní nabídku programů pro CONSUL-PMD povážlivě poklesl asi z více důvodů (?) budeme podle možností nabídku doplňovat. Dnes to je soubor Matematika 5 A. Poštulky z Opavy, příště to budou snad nové hry (po nich je relativně největší poptávka, jakoby počítače nebyly použitelné pro výuku a procvičování), a další myšlenkové tělocviky s popisem, jak řešit některé IQ testy.

Nově jsou nabízeny programy pro PC, které si lze levně prohlédnout na vratných demodisketách. Elementární geometrie vznikla přeprogramováním 10 lekcí ze souborů SWD 20,22 pro Consuly. Základní matematika pro ZŠ vyšla ze zkušeností M. Pospíšila (SWD 32), zatímco Matematiku ZŠ vymyslel pro dceru M. Kuneš k procvičování příkladů postupně od 1. do 4. třídy.

Podobné programy jsou i v nabídkách:

GRADA Bohemia, Uralská 6, 16000 Praha 6
Škola 2000, Na bateriích 37, 16200 Praha 6

Přeji Vám nové poznatky při zkoušení práce s QBasicem a Vaším žákům úspěšnou práci s nabízenými programy.

Pavel Hlaváček

Programové vybavení CONSUL 2717/PMD85-2,3

Kč 198,-

SWD71/SWK71

MATEMATIKA - 5

SWD71/SWK71

Autor: Alois Poštulka, Opava

Soubor programů pro podporu výuky matematiky a kontrolu pochopení probírané látky žáky na nižším stupni základních škol.

JEDCASU, JEDDELKY, JEDHMOI, JEDNOTKY

Testy převádění jednotek času, délky a hmotnosti nebo jejich kombinací. Testy obsahují vždy 8 částí po 5 úkolech výběrem z 80 možných příkladů, umožňují opravu po chybné odpovědi, ale bez zisku bodu do hodnocení. V úvodu testů je upozornění, že je nutné místo desetinné čárky používat desetinnou tečku. Hodnocení je v bodech.

DOSIA, DOTISICE

Počítání do sta nebo do tisíce s možností: sčítání, odčítání, násobení, dělení a dělení se zbytkem. Výběr se provádí stiskem klávesy číslíce vedle nabídky. Po startu je požadováno jméno žáka pro nastavení generátoru náhodných čísel. Výsledek počítání z paměti je označen hvězdičkou při správné odpovědi, nebo otazníkem a uvedením správného výsledku početní operace. Průběžné hodnocení je po každých 8 příkladech s možností návratu do nabídky, z níž si lze pak vybrat přehled hodnocení ve tvaru: jméno žáka a dílčí známka.

DESCIS

Program vypíše postupně 8 náhodných příkladů na počítání s desetinnými čísly (sečítání, odčítání a násobení dvou nebo i tří čísel z paměti), úvodem je upozorněno na to, že znakem násobení je hvězdička. Správná řešení jsou zobrazena, ale nekomentována, při chybě je vypsán správný výsledek, na závěr celkový počet správných odpovědí z osmi a navržená známka s možností pokračování v testu.

DELDESCI

Procvičování dělení desetinných čísel. Je zapsán příklad a je nejprve nutno vynásobit dělence i dělitele vhodným násobkem (10, 100, 1000, 10000). Potom je nutno přepsat dělence i dělitele vynásobenými hodnotami a vypočítat výsledek. Za každý správný krok je v dolní části zobrazena hvězdička, nebo při chybě otazník. Celkem pro tři zobrazené příklady s možností pokračování.

ROZKLAD1, ROZKLAD2

Rozklad čísla na součin prvočísel postupně jak program napovídá, při chybě je napsáno jen co není čím dělitelné a očekává se zadání jiného prvočísla dokud není výsledkem postupných dělení 1. Potom je inverzně zobrazen ještě celý rozklad a nabídka dalšího rozkladu nebo ukončení programu.

Druhý program je složitější o to, že musí žák vypočítat podíl rozkládaného čísla nebo jeho zbytku a určeného prvočísla, dokud není zbytkem jednička.

ZLOMKY 1-6

Programy na zjednodušování zlomků (1), úpravy na většího jmenovatele a čitatele (2), převádění složených zlomků na jednoduché (3) a naopak (4), sečítání zlomků (5) a násobení zlomků s nutným krácením (6). Postup činnosti žáka při práci je stručně uveden vždy v dialogovém řádku, blikající otazník je na místě číslíce, která má být vypočtena, upravena nebo jen přepsána.

STEJNACI, TRETIMOC

Program vypíše číslo, vzniklé vynásobením dvou shodných neznámých čísel. Úkolem je určit neznámé číslo (t.j. odmocninu) na několik pokusů, v nichž se zobrazuje, zda žák hádá málo nebo moc, nebo správně a zobrazí se i počet pokusů. Podobně v programu TRETIMOC se jedná o hádání třetí odmocniny ze zobrazeného čísla.

OBRAZCE

Program nabídne výklad obsahu a obvodu se vzorečky pro čtverec a obdélník, jichž je pak využito pro výpočty nepravidelných pravoúhlých obrazců nakreslených ve čtvercové síti (jde o sečítání ploch obdélníků a čtverců).

KVADR

Podobný program pro určování objemu a povrchu různých kvádrů tvořených jednotkovými krychlemi. Nejprve je vykreslen kvádr a má být určen jeho objem a potom jeho povrch. Jak při správné odpovědi tak i při chybě je uveden vzoreček, dosazení hodnot stran a správný výsledek.

QBasic - dialog s počítačem a programem.

Tato část bude pojednávat o vstupu dat do počítače a výstupu jím zpracovaných informací na obrazovku monitoru (nebo tiskárnu). Pro tento dialog slouží především následující příkazy a funkce, vztahující se k textovému režimu. Grafice se věnujeme později.

Příkazy pro výstup dat:

CLS	-mazání obrazovky nebo její části a nastavení kursoru
COLOR	-nastavení barvy zobrazení textu a jeho pozadí
LOCATE	-umístění kursoru na určený řádek a sloupec obrazovky
LPRINT	-výpis dat na tiskárnu LPT1
PALETTE	-změna jedné nebo více barev ve zvolené paletě barev
PRINT	-výpis dat na výstupní zařízení (obrazovku, soubor aj)
PRINT USING	-formátovaný výpis dat na výstupní zařízení
SCREEN	-nastavení základních charakteristik (módu) zobrazení
VIEW PRINT	-vymezení části monitoru pro výpis textových dat
WIDTH	-volba počtu řádků a znaků na řádku v textovém režimu
WRITE	-výstup dat v jiné formě než v příkazu PRINT

Funkce spojené s výstupem dat:

CSRLIN	-řádková souřadnice kursoru na obrazovce
CHR\$()	-znak, jehož ASCII-kód je uveden v závorce (argumentu)
POS	-sloupcová souřadnice kursoru na obrazovce
SCREEN	-číselný kód ASCII-znaku na pozici kursoru obrazovky
SPC()	-vynechání mezer v počtu uvedeném v závorkách
STRING\$()	-řetězec tvořený zvoleným počtem znaků z ASCII-kódu
TAB()	-nastavení kursoru na sloupec uvedený v závorkách

Příkazy pro vstup dat z klávesnice:

INPUT	-vstup číselných nebo textových dat do určené proměnné
LINE INPUT	-vstup řetězce dat do určené proměnné

Funkce pro vstup dat z klávesnice:

INPUT\$()	-vstup řetězce znaků, jejichž počet je v závorkách
INKEY\$	-předá proměnné poslední znak vložený z klávesnice

Toto jsou základní příkazy a funkce. Protože funkce HELP editoru jazyka QBasic umožňuje přesné zobrazení syntaxe každého příkazu v angličtině a je proto nejrychlejší a nejprístupnější nápovědou tehdy, nejsme-li si jisti, uvedeme si u každého příkazu nebo funkce syntaxi jak anglicky, tak i česky pro porovnání.

SCREEN Nastavení typu (módu) zobrazení.

Tímto příkazem zvolíme nastavení základních charakteristik zobrazení na monitoru počítače:

```
SCREEN mode% [, [colorswitch%] [, [activepage%] [, [visualpage%]]]
           mód [, [přep.barvy] [, [akt.stránka] [, [zobr.stránka]]]
```

Parametry příkazu uvedené v hranatých závorkách jsou nepovinné, lze použít jen některé z nich, nebo je všechny vynechat. Zadáte-li samotný příkaz SCREEN bez parametrů, nestane se nic, ale při pokusu o spuštění programu nebo řádku příkazů v přímém režimu (Immediate), vypíše interpret QBasicu chybové hlášení:

Illegal function call (Volání neznámé funkce)

a nabídne dvě možnosti v lomených závorkách: <OK> <Help>
 <OK> můžeme potvrdit pomocí ENTER, pokud si uvědomujeme chybu
 <Help> potvrdíme při dobré znalosti angličtiny, nápověda uvádí řadu možností chyb v různých situacích, hlavně zadání argumentu nevhodného nebo mimo povolený rozsah.
 To byl náš případ: vynechání čísla módu v příkazu SCREEN.

Správné použití příkazu je v minimální podobě následující:

SCREEN 0 - jen textový mód dle typu zobrazovací karty
 SCREEN 1 - grafika 320x200 nebo text 25 řádků po 40 znacích
 SCREEN 2 - grafika 640x200, text 25x80, karty CGA/EGA/VGA/MCGA
 SCREEN 3 - monochromatický monitor typu Hercules (720x348, 25x80)
 SCREEN 4 - grafika 640x400, text 25x80 pro Olivetti počítače
 SCREEN 7 - grafika 320x200, text 25x80, 16 barev, karty EGA/VGA
 SCREEN 8 - grafika 640x200, text 25x80, 16 barev, karty EGA/VGA
 SCREEN 9 - gr. 640x350, text 25x80 nebo 43x80, 16 barev, EGA/VGA
 SCREEN 10 - stejné jako SCREEN 9 ale monochromatický monitor
 SCREEN 11 - grafika 640x480, text 30x80 nebo 60x80, VGA/MCGA
 SCREEN 12 - grafika 640x480, text 30x80 nebo 60x80, VGA
 SCREEN 13 - grafika 320x200, text 25x40, VGA/MCGA, až 256 barev

Výběr je značný, jak se rozhodnout je někdy obtížné. Buď podle toho, jakou máte zobrazovací kartu, nebo chcete-li mít velká písmena (320x200), nebo program univerzálnější pro karty EGA i VGA a maximální rozlišení (640x350). To bych si zvolil: SCREEN 9.

Příkaz SCREEN n musí být z uvedeného důvodu jedním z prvních v programu, aby při pozdějším použití nezpůsobil problémy.

Pouze v módech 0 a 1 lze uplatnit volitelný parametr přepínače prep.barvy%, umožňující volbu mezi barevným a monochromatickým (černobílým) zobrazením takto:

Mód:	0	0	1	1
Přep.barvy:	0	Ne 0	0	Ne 0
Výsledek:	Bez barev	Barevně	Barevně	Bez barev

Parametr akt.stránka% je číslo, určující stránku ve videopaměti, do níž se zapisuje text, nebo do níž se kreslí pomocí grafických příkazů (pokud režim více stránek umožňuje dostatečná kapacita videopaměti na kartě, nutno zjistit z dokumentace karty).

Parametr `zobr.stránka%` je opět číslo té stránky videopaměti, která se trvale zobrazuje na stínítku monitoru.

Většina módů umožňuje jedinou stránku (0), některé dvě stránky (0,1), větší videopaměti tvoří čtyři (0-3) nebo až osm (0-7). Informace anglicky je opět v Helpu QBasicu pod heslem `Screen modes` nebo `SCREEN Statement`.

Příklad, který si můžete zapsat do okna Immediate a zkusit:

```
SCREEN 0,,1,0: PRINT "screen 0,,1,0 - akt.str.=1 - zobr.str.=0"
    0 - zobrazí se stránka 0 (asi to bude jiný text)
    1 - zapisujeme do stránky 1, která se nezobrazuje
SCREEN 0,,0,1: PRINT "screen 0,,0,1 - aktuální 0 - zobrazena 1"
    1 - zobrazí 1.videostránku dříve zapsaného textu:
        screen 0,,1,0 - akt.str.=1 - zobr.str.=0
    0 - zapisujeme do stránky 0, která se nezobrazuje...
SCREEN 0,,1,0 - zobrazí 0.videostránku s textem 2.příkazu PRINT:
    screen 0,,0,1 - aktuální 0 - zobrazena 1
```

Iato vymoženost QBasicu umožňuje při dostatečné velikosti videopaměti připravit text nebo grafiku do nezobrazované stránky a pak je jediným příkazem zobrazit, a na schovanou předchozí obrazovku připravit nějaké jiné překvapení.

Vynechaný `přep.barvy%` má číselnou hodnotu 0, a není jej nutno zapisovat. Symbol % určuje typ proměnné, v tomto případě tzv. krátké celé číslo (integer), jak si vysvětlíme později.

COLOR Nastavení barev zobrazení.

Nastavení barev výstupu závisí opět na módu zobrazení takto:

```
COLOR [foreground%] [, [background%] [, border%]]... screen 0
    [popředí%   ] [, [pozadí%   ] [, rámeček%]]
COLOR [background%] [, palette%] ..... screen 1
COLOR [foreground%] ..... screen 4/12/13
COLOR [foreground%] [, background%] ..... screen 7-10
```

`popředí%` - barva textu v rozsahu 0-31 nebo pera při kreslení grafiky 0-15,

`pozadí%` - barva podkladu 0-15 (podložky textu nebo grafiky),

`rámeček%` - barva rámečku po obvodu obrazovky.

`pozadí&` - fyzická barva pozadí jako dlouhé (&) celé číslo,
&h00mmzzcc&, kde mm = modrá (&h00 - &hff),
zz = zelená, cc = červená

`paleta%` - číslo palety se 4 barvami (sudými nebo lichými):
paleta 0: 0-pozadí, 1-zelená, 2-červená, 3-hnědá
paleta 1: 0-pozadí, 1-zel.modrá, 2-fialová, 3-bílá

LOCATE Umístění kursoru na řádek a sloupec.

Tento příkaz umožní nastavení kursoru do zvoleného místa na stínítku před provedením dalších příkazů. Jeho obecný tvar je:

LOCATE [row%] [, [column%] [, [cursor%] [, start% [, stop%]]]]
 [řádek] [, [sloupec] [, [kursor] [, start [, stop]]]]

řádek - textový řádek, na nějž má být kursor nastaven (1-25 a j.)
sloupec - pozice písmene na řádku (1-40 nebo 1-80 podle SCREEN mód)
kursor - viditelnost kursoru: 0-neviditelný, 1-viditelný,
start - horní mikrořádek, od něhož začíná obdélník kursoru (0-31)
stop - dolní mikrořádek, na němž končí obdélník kursoru (>start)

Zkuste si zobrazit kursor na řádku 20 ve sloupci 40 jako obdélník od mikrořádku 1 do mikrořádku 8. Aby zůstal zobrazen 3 sekundy, použijeme příkaz SLEEP 3. Pomocí klíče F8 se nastavíme do přímého režimu (Immediate) a zapíšeme dle zadání:

CLS: LOCATE 20,40,1,1,8: SLEEP 3

Výsledkem je blikající poloviční kursor na uvedené pozici. Pokud je za LOCATE příkaz tisku (PRINT) nebo vstupu dat (INPUT), posouvá se kursor za tato data jen v případě, že jsou zakončena středníkem. Pokud bychom chtěli zobrazit kursor v tomto místě, postačuje vynechat v LOCATE řádek a sloupec a specifikovat jenom viditelnost a velikost kursoru. Zkuste si to, třeba se vám to někdy hodí:

locate 5,1,1,1,4: sleep 2: print "Kursor"; :locate, ,1,5,9: sleep 2

Vynecháte-li středník, usadí se kursor na následujícím řádku. Pozici kursoru na obrazovce můžeme zjistit pomocí funkcí:

CSRLIN - vrací do proměnné běžnou pozici řádku na němž je kursor:
 radek = csrlin

POS(x) - vrací do proměnné číslo sloupce s kurem, x je pouze formální libovolný bezvýznamný výraz: sloupec = pos(1)

Tyto funkce používáme tehdy, když potřebujeme nastavit kursor na nějakou novou pozici, vztahující se k té předchozí, např. o 5 řádků níž a 10 sloupců vpravo bude:

LOCATE csrlin + 5, pos(1) + 10

Pokud hodnoty funkcí přiřadíme proměnným, můžeme nastavovat kursor na určité místo dané hodnotou proměnné, např:

radek = csrlin: sloupec = pos(x): locate radek, sloupec

PRINT Zobrazení výstupních dat na monitoru.

Toto je snad nejčastěji používaný příkaz QBasicu, jehož syntaxe je následující:

PRINT [#filenumber%,] [expressionlist] [{; | ,}]
 [#číslo_souboru] [seznam_výrazů] [středník nebo čárka]

- #číslo_souboru - číslo otevřeného souboru pro výstup (např. #1)
 - není-li uvedeno, píše PRINT na stínítko.
seznam_výrazů - jeden nebo více číselných nebo textových výrazů
 které budou zapsány do souboru nebo na stínítko.
oddělovače: ; - znamená, že se kurzor nastaví za poslední znak,
 je to důležité i na řádku 24 nebo 25, pokud se
 píše až do posledního sloupce (40 nebo 80), aby
 kurzor zůstal na tomto řádku a neposunul text na
 obrazovce o jeden řádek nahoru.
 , - značí, že se kurzor nastaví na začátek následující
 tiskové zóny (14-sloupcové): 1,15,29,43,57 pro
 režim 80 znaků na řádek, jinak jen 1,15 (40 zn/ř).

Zápisem do souborů se budeme zabývat později.

Každý samostatně použitý příkaz PRINT bez oddělovače zapisuje buď na místo kursoru (LOCATE), nebo na nový řádek. Po příkazu CLS je kurzor nastaven na 1.sloupec 1.řádku. Zkuste si to následujícím sledem příkazů, zadaných v přímém režimu (Immediate):

```
cls: print "1234567890....5....0": print -5: print +6: print "7"
```

První PRINT vypíše textový řetězec s pozicemi sloupců na prvním řádku. Druhý příkaz vypíše konstantu -5 na druhém řádku a následující příkaz konstantu +6 na třetím řádku, bez znaménka. Jak vidíte na stínítku, je první sloupec vyhrazen pro znaménko. Čtvrtý PRINT vypíše textový řetězec s jedním znakem hned od prvního sloupce, zde o znaménko nejde (kladné a záporné texty QBasic nezná a nerozlišuje).

Z příkladu je zřejmé, že textové řetězce z písmen nebo čísel je nutno uvádět v uvozovkách, konstanty nebo proměnné se uvádějí bez uvozovek. Pokud se má nějaký text vícekrát vypisovat pomocí příkazu PRINT, je vhodné jej zapsat do tzv. řetězcové proměnné a tu pak uvést jako argument příkazu. Například:

```
a$ = "opakovaný text": locate 10,12: print a$
```

Podobně se místo číselných konstant v příkazu PRINT používají číselné proměnné, nebo výrazy s číselnými proměnnými a konstantami, například:

```
a = 2: b = a + a: print "a ="; a, "a - a ="; a - a, "b ="; b
```

Používání proměnných a operace s nimi si uvedeme později.

Zkuste si nyní tiskové zóny úpravou předchozího příkazového řádku - za konstanty zařadíme čárky, + můžeme vynechat.

```
cls: print "1234567890....5....0....5....0": print -5, 8, "7"
```

Znaménko + u 8 by bylo ve sloupci 15, znak 7 je ve sloupci 29. Změníme-li oddělovače na středníky, můžeme ze stínítka zjistit, že za vypsání čísla je vynechána jedna mezera jako oddělovač. Texty nejsou oddělovány, to je nutno zajistit vložení mezer přímo do textu.

SPC(n%) Funkce pro vynechání n mezer při výpisu.

Používá se v příkazech PRINT a LPRINT k vynechání určeného počtu mezer, umožňuje tak vytvořit v textu na obrazovce volné místo pro jiný text nebo obrázek, selektivně smazat část obrazovky. Ale pozor při mazání konců řádků od pozice určené LOCATE r,s. Nastavíme-li např. LOCATE 1,70 a zadáme PRINT SPC(10); zůstane nesmazán poslední 80 sloupec. Mohli bychom zadat SPC(11) a bylo by to v prostředí QBasicu dokonce funkční, ale nefunkční po překladi programu v QuickBasicu nebo VisualBasicu, které si určené sloupce sečítají, a když zjistí, že např. 70+11=81, což je více než je povoleno znaků na řádku, funkci neprovedou, aby něco nepokazili, a tím není v textu volné místo vytvořeno. Proto musíme respektovat překladač, a zadat správné PRINT SPC(10); " "; se dvěma středníky a mezerou v uvozovkách, která maže 80. sloupec. Příklady si uvedeme později.

TAB(sloupec%) Funkce pro nastavení cursoru na určitý sloupec.

Používá se v PRINT a LPRINT a podobně jako SPC(n) maže texty mezi definovanými sloupci v argumentech funkce TAB(). Funkci TAB můžeme použít vícekrát v jediném příkazu PRINT, oddělujeme středníkem. Pokud nejsou argumenty v rostoucí posloupnosti, platí TAB() s nižším číslem sloupce pro následující řádek. Podobně při stejných argumentech se texty vypíší na různé řádky:

```
print tab(10);10; tab(20);20; tab(40);40; tab(79);"79"  
print tab(30);30; tab(30);30; tab(29);29; tab(79);7
```

Poslední funkce tab nastavila kurzor na předposlední sloupec, ale protože u čísla je nutno počítat se sloupcem na znaménko a dalším sloupcem na oddělení čísla, zobrazí se 7 na dalším řádku (zobrazovanou informaci nelze rozdělit).

STRING\$(n) Funkce vytvářející řetězec stejných znaků.

Pomocí této funkce a příkazu PRINT můžeme popsat celý řádek

nebo i celou obrazovku jedním znakem, neboť tato funkce předává řetězec určené délky zadaného znaku z ASCII kódu nebo prvního znaku z řetězce:

```
SIRING$(length%,(ascii_code% | stringexpression%))  
      (délka,(ascii_kód_znaku | řetězcový_výraz))
```

Parametry v () nejsou volitelné, ale povinné, naopak v { | } je nutno zvolit jednu z možností

délka - počet znaků v řetězci,
ascii_kód_znaku - číselný kód (1-255) z ASCII tabulky znaků, ne všechny od 1 do 32 jsou zobrazitelné, nebo:
řetězcový_výraz - libovolný řetězcový výraz v uvozovkách nebo v řetězcové proměnné, využije se jen první znak.

Zkusíme si vyplnit celou obrazovku například hvězdičkami. V textovém režimu 25x80 je počet zobrazitelných znaků 2000.

```
cls: print string$(2000,"*");: locate 20,10: print spc(70);" "
```

Poslední dva příkazy ukazují možnost vymezerování části textu, o němž bylo psáno výše. Zapomenete-li středník za PRINT SIRING\$(); a na konci příkazového řádku, posune se "text" o jeden řádek a zůstane nepopsán řádek 24, a řádek 25 je přepsán zprávou interpreteru QBasic: Press any key to continue. Poslednímu výpisu můžeme zabránit vložení čekání příkazem SLEEP n, nebo jinou čekací smyčkou, např. podobnou té z popisu editoru QBasic.

Problémem je někdy nedostupnost tabulky znaků v ASCII kódu. Nejjednodušší je zobrazit si ji pomocí HELP (ALI + H), nastavením na <Contents> a potom pomocí klávesy TAB vybereme v oddíle Quick Reference řádek <ASCII Character Codes> a po stisku ENTER se zobrazí dolní polovina tabulky, a po PAGE DOWN její horní polovina. V tabulce je v několika sloupcích uveden nejprve kód, vedle něj znak a v závorce zkratka tzv. služebních znaků. Tak například kód hvězdičky je 042, české znaky mohou být pouze v horní polovině tabulky mezi semigrafickými a jinými znaky.

Rychleji se ke kódové tabulce dostanete tehdy, nastavíte-li kursor na příkaz SIRING\$, stisknete klíč F1, zobrazí se nápověda tohoto příkazu a v její dolní polovině je <ASCII Character Codes> na nějž se dostanete stiskem TAB (4 krát) a pak stačí jen ENTER.

Potřebujeme-li vypsát některý znak, který není dostupný klávesou, musíme použít jednu z možností:

- použít funkci CHR\$(ASCII_kód), nebo
- na místě tohoto znaku v textu přidržet klávesu ALI a na numerické části klávesnice vypsát číslo kódu, například 227 pro symbol Ludolfova čísla nebo 251 pro odmocninu.

CHR\$(ascii_code%) Funkce vrací znak odpovídající ASCII kódu.

ASC("text") Funkce vrací ASCII kód prvního znaku řetězce.

Jedná se o funkce, které se navzájem doplňují, jak se můžeme přesvědčit následujícími příkazovými řádky:

```
print asc("ASCII"): REM vypíše na obrazovce kód znaku A, t.j. 65
print chr$(65)            'vypíše znak odpovídající kódu 65, t.j. A
```

Použití apostrofu ' na příkazovém řádku odděluje komentář od jednotlivých příkazů. Je možno použít i rezervované slovo REM jako ve starších verzích jazyka Basic, před REM však musí být dvojtečka. Apostrof respektuje počet mezer za koncem příkazů na řádku.

WIDTH Příkaz změny počtu řádků a sloupců na monitoru.

Tento příkaz je obecnější pro výstupní zařízení, kdy určuje počet znaků v řádku, jak uvidíme v části o datových souborech.

```
          WIDTH [columns%] [,rows%]
                  [sloupců] [,řádků]
sloupců - určený počet sloupců, musí být 40 nebo 80,
řádků   - určený počet řádků na stínítku, závisí na použité zobrazovací kartě, může být: 25, 30, 43, 50 nebo 60.
```

VIEW PRINT Příkaz vytvoření textového okna na obrazovce.

Pomocí tohoto příkazu můžeme vymezit vodorovný pruh na obrazovce jako textové okno. Jeho syntaxe je jednoduchá:

```
VIEW PRINT [toprow% TO bottomrow%]            'toprow = horní řádek
          [od_řádku po dolní_řádek]
```

Rozsah vymezených řádků závisí na typu zobrazovací karty a připojeného monitoru. Ve vytvořeném okně lze používat nastavení kursoru pomocí LOCATE jen na povolených řádcích a příkaz CLS 2 maže také jen toto okno. To znamená, že na "nedostupné" části stínítka můžeme vypsát nebo vykreslit předem stálé informace (nápovědu nebo popisovaný obrázek) a komunikovat s uživatelem pouze ve vymezeném pracovním okně. Příklad:

```
print string$(1920,208): view print 10 to 15: print 1: print ,2
print ,3: print ,,,4: print 5: print "text";: sleep 2: cls 2
```

Příkazové řádky mohou být dva, a provedete je postupně, nebo lze zadané příkazy napsat na jediný řádek a provést najednou, po 2 sekundách se pak textové okno vymaže.

Pokud si vytvoříme užší okno pomocí `PRINT SPC(n)`, musíme je plně ošetřovat, tj. mazat stejnými příkazy `PRINT SPC(n)` a vypisovat pomocí `LOCATE r,s: PRINT...`, nelze použít `CLS`

PRINT USING Příkaz pro zobrazení dat v zadaném formátu.

Největší potíží při vypisování tabulek čísel dělá určité nutnost zarovnávání k pravému okraji sloupce. Právě pro tento případ a nejen pro něj je určena přípona `USING` (používá se i pro výstup na tiskárnu nebo zápis do souboru).

`PRINT USING formatstring; expressionlist [{; | ,}]`
 formát.řetězec; seznam_výrazů

`formát.řetězec` - uvozovkami určené formátovací předpisy tvořené některými speciálními znaky,

`seznam_výrazů` - vyjmenovaná data, konstanty, text aj., oddělená čárkami; počet má odpovídat počtu formátů.

Znaky formátující číselné výrazy (tvar za `PRINT USING`):

- # - pozice číslic v čísle se zarovnáním vpravo: "###";
- , - pozice tisíců a milionů vlevo od desetinné tečky: "n,###.nn";
- + - pozice znaménka čísla (kladného nebo záporného): "+###";
- - vypíše exponenciální tvar: "nn.##----";
- ##- - znaménko (-/+) na konci skupiny se tam zobrazí: "##-";
- ** - doplní číslo zleva tímto znakem: "*****";
- n - před číslem se zobrazí znak měny: "n,###.nn";
- **n - kombinace předchozích možností v daném pořadí: "*****";

Znaky formátující řetězcové výrazy:

- ! - výstup jen prvního znaku řetězce: "!"; a#;
- & - celý řetězec se vypíše beze změny: "&"; c#
- \\ - vypíše 2 znaky nebo 2+n znaků, kde n je počet mezer mezi zpětnými lomítky, zarovnává se vlevo "\\ "; d#
- _ - podtržítka způsobí, že se následující znak píše jako znak a ne jako formátovač: např. vykřičník "_!"

Zkuste si například:

```
print using "-*****.##"; +13.2488      'vypíše: -*****13.25
print using "\\ \\=!_!"; "abcdef", "123"  'vypíše: abc=1!
```

Z tohoto příkladu plyne, že formátovací uvozovky za `USING` platí jen před středníkem, za ním lze použít řetězce v uvozovkách a také funkce `spc()` a `tab()`, nutné pro úpravu výpisu dat.

Formátovací řetězec platí jak pro jednu konstantu či proměnnou (číselnou, textovou), tak pro jejich libovolný počet, pokud se vejdu na řádek. Může být předem uložen do řetězcové proměnné před středníkem, aby nezabíral tolik místa na příkazovém řádku:

```
a#="+###.nn Kc": b=13.50: print using a#; b      'vypíše: +13.50 Kc
```

Zkuste si dva formátovací příkazy na zkrácení seznamu jmen, kde desetinná tečka je oddělovačem i symbolem zkrácení jména:

```
print using "! \           \"; "Josef", "Nos"      'vypíše: J. Nos
```

WRITE Příkaz pro zápis dat na obrazovku nebo do souboru.

Dalším příkazem pro výstup dat na obrazovku nebo do souboru je **WRITE**, který vypíše seznam_výrazů numerických nebo řetězcových oddělených čárkou. Je-li jeho parametr vynechán, vypíše prázdný řádek. Řetězcové proměnné se zobrazují včetně uvozovek, číselné proměnné se zobrazují bez znaménkové mezery při + a oddělovací mezery za proměnnou. Za poslední znaky vloží interpret QBasic automaticky řídící znaky CR/LF a přejde na začátek následujícího řádku nebo na zápis další věty do výstupního souboru. Syntaxe:

```
WRITE [(#)filenumber%,] expressionlist  
      [ číslo_souboru] seznam_výrazů
```

číslo_souboru - je číslem otevřeného sekvenčního souboru,
 je-li vynecháno, vypisuje příkaz na obrazovku
seznam_výrazů - jedna nebo více proměnných nebo výrazů, odděle-
 ných čárkami nebo středníky.

WRITE zapisuje hodnoty do souboru ve tvaru, který může být čten pomocí příkazu **INPUT**, jak si vysvětlíme později.

LPRINT, LPRINT USING Příkazy pro výstup dat na tiskárnu.

Tyto dva příkazy mají shodnou syntaxi jako příkaz **PRINT**, včetně možnosti použití funkcí **IAB()** a **SPC()**. Jejich výstup je směrován na standardní rozhraní pro tiskárnu **LPT1: (PRN:)**. Tento výstup je možno přesměrovat otevřením výstupního souboru, ale je nutno používat příkaz **PRINT #n**, nebo **PRINT #n, USING**, např.

```
OPEN "LPT2:" FOR OUTPUT AS #2  
.....  
PRINT #2, seznam_výrazů  
PRINT #2, USING "###.###"; 12.34567
```

Pokud máte připojenu tiskárnu, tak zkuste experimentovat.

INPUT Příkaz pro vstup dat nejen z klávesnice.

Pro vstup dat z klávesnice je určen příkaz:

```
INPUT [;] ["prompt";] variablelist  
          [nápověda;] seznam_proměnných  
LINE INPUT [;] ["prompt";] variable  
          [nápověda;] řetězcová_proměnná
```

; za INPUT - způsobí, že po ukončení vstupu dat z klávesnice zůstává kurzor na stejném řádku,
nápopěda - text, zobrazený před místem očekávaného vstupu dat, není-li uveden, může se zobrazit jen otazník,
{; | ,} - pokud za ním následuje středník, - následuje-li čárka, otazník se nezobrazí.
seznam_proměnných - jedna nebo více proměnných libovolného typu oddělených čárkami, vstup dat pro každou proměnnou musí být ukončen stiskem ENTER.
řet.proměnná v příkazu LINE INPUT je jediná, do níž jsou uložena všechna zapisovaná data včetně uvozovek i čárek, do celkového počtu 255 znaků ukončených klávesou ENTER.

Zkuste si v přímém režimu následující příkazy:

```
line input "Zadej cokoliv:";a$: print a$
```

Podobné příkazy jsou určeny pro vstup dat ze souboru uvedeného čísla:

```
INPUT #filenumber%, variablelist  
LINE INPUT #filenumber%, variable$
```

do seznamu proměnných nebo jediné řetězcové proměnné.

INPUT\$(n) Vstup určeného počtu znaků nejen z klávesnice.

Tato funkce se od příkazu liší tím, že je předem určen počet vkládaných znaků bez potvrzení klávesou ENTER. Určený počet znaků může být přečten i ze souboru uvedeného čísla (soubor může být i delší, přečte se pouze uvedený počet znaků):

```
INPUT$(n[, (#)filenumber%])  
n                    - počet znaků, které budou čteny,  
číslo_souboru - číslo dříve otevřeného souboru, je-li vynecháno, čte se z klávesnice.
```

Protože tato funkce nezobrazuje otazník ani vkládané znaky na obrazovce, musí být "ošetřena" jinými příkazy (LOCATE, PRINT). Do celkového počtu stisknutých kláves se započítávají i funkční klávesy včetně ESC a ENTER.

Příklad, ve kterém je lomítka naznačen počet vkládaných údajů:
locate 10,5: print "Zadej rok /rrrr/:"; r\$ = input\$(4): print r\$
locate 11,5: print "Zadej měsíc /mm/:"; m\$ = input\$(2): print m\$
locate 12,5: print "Zadej den /dd/:"; d\$ = input\$(2): print d\$

INKEY\$ Načtení jednoho znaku z klávesnice.

Zvláštní funkce bez parametrů obsahuje prázdný řetězec "" pokud nebyla od posledního vstupu INPUT nebo posledního použití

INKEY# stlačena žádná klávesa, jinak obsahuje kód první stisknuté klávesy. Obvykle se používá ve smyčce čekání na stisk nějaké klávesy:

```
print "Pro pokračování stiskni klávesu  
do  
klavesa# = inkey#  
loop while klavesa# = ""  
print "Stisknuta klávesa "; klavesa#
```

Iato funkce rozlišuje normální klávesy od funkčních, u nichž předává dva kódy: 00 a kód některé normální klávesy. Použití této speciality si uvedeme později, až poznáme některé další příkazy.

Typy dat a proměnných.

Každý programovací jazyk rozlišuje dva základní typy dat: čísla a znaky (většinou utvářející slova nebo texty, pak hovoříme o znakových řetězcích). V příkazech QBasicu se čísla uvádějí přímo, zatímco znaky je nutno uvádět v uvozovkách.

Oba typy dat mohou být uloženy dočasně nebo trvale do proměnných také dvou typů, které musí začínat písmenem:

- číselné proměnné pro čísla, např. a, b3, celkem, slovo
- řetězcové proměnné pro znaky: a#, b3#, celkem#, slovo#

Proměnné musí být "jednoslovní", proto se musí víceslovní názvy proměnných spojovat pomocí podtržítka: celkem_k_výplatě. S tímto podtržítkem jsme doposud uváděli některé překlady parametrů nebo argumentů příkazů a funkcí. Počet znaků proměnných je omezen na 40, ale doporučuje se co nejkratší, aby se vešel na řádek. Nesmí se používat rezervovaná (klíčová, vyhrazená) slova, použitá pro příkazy, funkce a operátory QBasicu, ani nesmí začínat jako funkce FN. Nerozlišuje se mezi velkými a malými písmeny.

```
x = 10: X = 1+5: print x          'vypíše: 6  
ale i% = 5: i = 0.55: i& = 3e3: i# = 2d2: print i%; i; i&; i#  
                                'vypíše: 5 .55 3000 200
```

Tento příkazový řádek ukazuje, že číselné proměnné odlišuje také jejich přípona, určující typ čísla v proměnné uchovávaného. Mimo tento způsob stanovení (deklarace) proměnné lze použít také speciální příkazy typu AS^{typ} (jako ^{typ}), DEF^{typ} (definuj ^{typ} od-do), COMMON (globální proměnné), SHARED (sdílené procedurami), STATIC (statické v procedurách a funkcích) a u polí potom DIM a REDIM.

Editor QBasicu kontroluje písemnou úpravu programu a sjednocuje zápis proměnných, je-li v programu mimo velkých X použito x budou všechny stejnojmenné proměnné změněny na malé x či naopak.

Před prvním přiřazením hodnoty proměnné jsou všechny numerické proměnné rovny nule a řetězcové proměnné mají hodnotu prázdného řetězce.

Jednoduchá proměnná se vztahuje k jednotlivému číslu, řetězci nebo strukturovanému typu záznam (viz kapitolu o souborech). Pole nebo indexovaná proměnná je představována množinou hodnot označených stejným jménem (identifikátorem) a odlišených indexem v závorkách

Číselné proměnné.

Jeden druh číselných proměnných by byl nevhodný, neboť by musel i pro jednociferné proměnné rezervovat v paměti tolik místa jako pro proměnné s nejvyšším počtem míst. Proto i QBasic rozlišuje čtyři typy číselných proměnných podle označení typu u jména proměnné (%,&!,#), které v paměti zabírají 2, 4 nebo 8 bytů a mohou mít uvedené rozmezí:

Čísla:	Typ:	RAM:	Minimální...Maximální
-krátká celá (Integer)	x%	2B	-32768...+32767
-dlouhá celá (Long Integers)	x&	4B	-2147483648...2147483647
-reálná jednoduchá (Single)	x!	4B	+/- 2.8E-45...3.4E38
-reálná dvojitá (Double)	x#	8B	+/- 4.9D-324...1.8D308

Pokud zapomenete u proměnné označit typ, bude jí reálné číslo s jednoduchou přesností (!), tzv. standardní typ (implicitní) zabírající v paměti 4 byty. To znamená nejen větší spotřebu paměťových míst, ale i více cyklů na čtení nebo zápis do paměti a tím zpomalení programu.

Zápis mezních hodnot reálných čísel je proveden formou pevné řádové čárky a proměnného exponentu: mantisaExponent, např.

$$\begin{aligned}0.00123 &= 1.23 * 10^{-3} = 1.23E-3 \\9876.54 &= 9.87654 * 10^3 = 9.87654E3\end{aligned}$$

Název jednoduché přesnosti plyne z toho, že číslo platí na 7 desetinných míst, zatímco číslo dvojitě přesnosti má až 16 míst.

Čísla mohou být zapsána buď v desítkové soustavě, nebo soustavě šestnáctkové nebo zastaralé osmičkové. Odlišení šestnáctkových čísel je provedeno předponou &H nebo &h (h - hexadecimální). Tato čísla používají číslice 0-9 a dále: A=10, B=11, C=12, D=13, E=14, F=15 a lze používat jak malá, tak i velká písmena. Nutno dát pozor na to, o jak velké číslo se jedná v různých proměnných:

Integer: &h0 = 0, &h7FFF = 32767, &h8000 = -32768, &hFFFF = -1
Long: &h7fffffff& = 2147483647, &h80000000& = -2147483647

Zvláštní místo mezi znaky mají tzv. operátory, které předepisují provedení určitých operací s dvěma nebo jedním operandem.

Aritmetické operátory pro provádění matematických operací:

+	sčítání	print 2+9	'vypíše: 11
-	odčítání	print 2-9	-7
*	násobení	print 2*9	18
/	dělení	print 2/9	.2222222
-	umocňování	print 9^2	81
\	celočíselné dělení	print 9\2	4
MOD	zbytek po celočísl.dělení	print 9 mod 2	1
=	přiřazení hodnot proměnným	dph = 0.23	

() závorky - používají se např. k oddělení operandů tam, kde si nejsme jisti pořadím priorit v provádění operací

Pořadí priorit těchto operátorů od nejvyšší k nejnižší je:

() - * / \ MOD + - =

Relační operátory pro porovnávání dvou výrazů. Výsledkem je buď nepravda (FALSE) vyjádřená hodnotou nula, nebo pravda (TRUE) vyjádřená nenulovou hodnotou, obvykle -1. Relační operátory mají větší prioritu než aritmetické:

=	rovnost	print 2 = 2	'vypíše: -1 (pravda)
<>	nerovnost	print 2 <> 2	0 (nepravda)
<	menší než	print 2 < 9	-1
>	větší než	print 2 > 9	0
<=	menší nebo rovno	print 9 <= 9	-1
>=	větší nebo rovno	print 9 >= 2	-1

Tyto operátory se používají především v rozhodovacích příkazech pro rozvětvení dalšího průběhu programu, podobně jako následující logické operátory. Ty se používají mezi operandy, které mohou mít dvě logické hodnoty: pravda (1) nebo nepravda (0):

		x = 1	1	0	0	
	Bude-li:	y = 1	0	1	0	
	Vypíše se ...					
AND	logický součin (konjunkce)	print x AND y ...	1	0	0	0
NOT	logický doplněk (negace)	print x NOT x ...	0	0	1	1
OR	logický součet (disjunkce)	print x OR y ...	1	1	1	0
EQV	logická ekvivalence	print x EQV y ...	1	0	0	1
XOR	výlučný součet-neekvivalence	print x XOR y ...	0	1	1	0
IMP	implikace	print x IMP y ...	1	0	1	1

OBASIC dále umožňuje vytváření funkčních operátorů jako uživatelem definovaných funkcí pouze v jeho programech (nebo ve speciálních knihovnách programů, šířených různými firmami). Podrobněji o těchto funkcích pojednáme později.

Některé číselné proměnné mohou obsahovat konstanty a to buď přímé (literály), tj. přímo uvedené v programu nebo podprogramu, nebo předem pojmenované (symbolické) pomocí příkazu CONST. Tyto konstanty nelze v programu změnit nebo náhodně přepsat, editor by okamžitě ohlásil DUPLICATE DEFINITION-pokus o definici duplikátu. Pomocí symbolických konstant můžeme definovat další symbolické konstanty, k jejich definování však nelze použít proměnné, které nemají konstantní hodnotu a mohly by se v programu změnit.

Řetězcové proměnné.

OBasic rozlišuje dva typy těchto proměnných s příponou $\$$ (nebo znaménko měny, používané v předlistopadových tiskárnách):

- znakové řetězce s proměnnou délkou (0 - 32767) ... typ STRING,
- znakové řetězce s pevnou délkou n bytů typ STRING*n

Obsah těchto proměnných (znakový řetězec) se definuje v uvozovkách, aby písmena a číslice odlišil od číselných výrazů.

```
konec$ = "Konec hry.":print konec$;konec      'vypíše: Konec hry. 0
```

protože číselné proměnné 'konec' nebyla dosud přiřazena hodnota a je proto nulová.

Mezi řetězci nelze provádět aritmetické operace, ale lze je spojovat pomocí znaménka +, je však nutné pamatovat na mezery:

```
jm$ = "Karel": pr$ = "Rak": print jm$+pr$      'vypíše: KarelRak  
Zkusíme to opravit na: print jm$+" "+pr$      Karel  Rak
```

Je možné spojovat řetězcové proměnné se znakovými řetězci v uvozovkách a naopak lze mezi řetězce vkládat řetězcové proměnné, Řetězce lze mezi sebou porovnávat pomocí relačních operátorů, neboť jsou v paměti reprezentovány ASCII kódy písmen. Protože např. kód A=65 a B=66, potom výpis výrazu "A" < "B" bude pravda:

```
print "A" < "B"      'vypíše: -1
```

Tento poznatek je použit pro třídění řetězcových proměnných podle abecedy. Ale pozor: malá písmena jsou větší než velká, neboť mají ASCII kódy od 97 do 122 a největší problém je s češtinou s kódy někde mezi 128 a 255 (podle typu kódování). Pokud nedojde k porovnání počátečních písmen, pokračuje se druhými písmeny a tak dále až po odlišné pozice. Např. "AAA" > "AA" protože třetí A je svým kódem 65 skutečně větší než nic (prázdné místo = 0) na 3. místě druhého řetězce. Je to způsob podobný třídění hesel ve slovnících a encyklopediích.

S řetězci znaků a jejich proměnnými je možno pracovat pomocí řady příkazů a funkcí, které si probereme později.

Obsah lekcí kursu geometrie pro PC.

Autor: Pavel Hlaváček

Kurs elementární geometrie obsahuje pět grafických programů s definicemi a dynamickým znázorněním vzniku osvětlovaných pojmů nebo změn geometrických útvarů. Lekce mají 7-14 kapitol označených písmeny (A-N). Po úvodní obrazovce s číslem lekce je zobrazena nabídka z níž lze vybrat stiskem klávesy písmene na konci řádku kapitoly. Konec každé kapitoly pak umožňuje buď návrat do nabídky (N), nebo pokračování (P) v lekci další kapitolou.

GEOM1: Elementární geometrie	1	GEOM3: Elementární geometrie	3
-----		-----	
Historie axiomů geometrie.....A		Konvexní geometrické útvary....A	
Přímka určená body.....B		Úhel konvexní a nekonvexní....B	
Úsečka.....C		Shodnost úhlů.....C	
Cvičení - úsečky.....D		Operace s úhly.....D	
Shodnost úseček.....E		Osa úhlu.....E	
Operace s úsečkami.....F		Kružnice a kruh.....F	
Násobení úseček.....G		Velikost úhlu.....G	
Rozdíl úseček.....H		Cvičení na velikost úhlu.....H	
Definice geometrie.....I		Iětiva a úhly uvnitř kružnice..I	
Měření úseček.....J		Iečna kružnice a úsekový úhel..J	
Cvičení z měření úseček.....K			
Vzdálenost úseček.....L		GEOM4: Elementární geometrie	4
Cvičení na vzdálenosti.....M		-----	
Definice metru.....N		Trojúhelníky.....A	
		Shodné trojúhelníky.....B	
GEOM2: Elementární geometrie	2	Příčky trojúhelníků.....C	
-----		Osy stran a úhlů trojúhelníků..D	
Rovina a její definice.....A		Čtyřúhelníky.....E	
Polorovina.....B		Iětivový čtyřúhelník.....F	
Opakování - určení roviny.....C		Iečnový čtyřúhelníkG	
Trojúhelník.....D			
Rovnoběžnost.....E		GEOM5: Elementární geometrie	5
Cvičení - různorovnoběžky....F		-----	
Prostor.....G		Míra a její vlastnosti.....A	
Čtyřstěn.....H		Opakování měření.....B	
Přímka a rovina.....I		Obsah geometrických útvarů....C	
Vzájemná poloha 2 rovin.....J		Měřitelnost útvarů.....D	
Vzájemná poloha 3 rovin.....K		Shodná zobrazení.....E	
Vzdálenost útvarů.....L		Osová souměrnost.....F	
Konec lekce.....M		Otočení a středová souměrnost..G	
		Posunutí.....H	

Kurs začíná krátkým historickým úvodem od Euklida po Hilberta, kteří se zasloužili o výklad geometrie založený na axiomech, jako tvrzeních, přijímaných bez důkazů, z nichž je postupně vytvářena celá elementární geometrie. Výklad začíná od přímky přes polopřímku k úsečce a je procvičován na jednoduchých příkladech. Potom jsou podrobněji vysvětleny a graficky znázorněny operace s úsečkami, jejich měření a vzdálenosti, včetně obrázku původní definice metru.

Po definicích rovin a polorovin jsou uvedeny možnosti vzájemných poloh přímek, přímek a rovin, rovin v prostoru, vzdálenosti útvarů až po příklad čtyřstěnu jako průniku čtyř poloprostorů.

Po vysvětlení pojmů konvexní a nekonvexní jsou definovány úhly a vysvětleny i procvičovány možné operace s úhly, velikost úhlu je odvozena z délky oblouku kružnice. Cvičení je zaměřeno na určování úhlů mezi ručičkami hodin.

Další lekce je věnována trojúhelníkům a čtyřúhelníkům, poslední pak měřitelnosti útvarů a různým typům shodných zobrazení.

Podkladem pro zpracování kursu byly publikace:
Základy elementární geometrie pro učitelství 1. stupně ZŠ, SPN 1985,
Texty k základům elementární geometrie, Masarykova univerzita 1990.

Kurs využívá češtinu v kódu bratří Kamenických, který umožňuje mimo českých znaků i zobrazení některých písmen řecké abecedy pro označování úhlů, stupňů a Ludolfova čísla. Podprogram češtiny bude expedován na disketě se standardními lekcemi kursu, není součástí demoverse. Programy byly vytvořeny v grafickém režimu 640x350 bodů v prostředí QBasic v MS-DOS 5.0 a přeloženy pomocí VisualBasicu pro DOS. Zdrojové texty v QBasicu lze objednat u distributora.

Programy jsou dodávány ve spustitelném GEOMx.EXE tvaru. Nevyžadují zvláštní nároky na počítač (jednotlivé lekce mají kolem 110kB) předpokládá se grafická karta EGA nebo VGA se 16 barvami, zobrazení na monochromatickém displeji je také vyhovující (nikoli Hercules).

Distributor uděluje multilicenci k instalování souboru programů na počítačích té právnické osoby (školy), která programy zakoupila, tj. programy lze současně provozovat na několika počítačích.

Demoverse na vratné disketě 5.25-3.5 pouze lekce 2., 3. a 4., neumožňuje zastavení během výkladu (je nahrazeno prodlevou 1 sec).

Cena: -zaváděcí do konce června 1994 398,- Kč + 5% DPH
-uživatelé souborů SWD 20/22 Geometrie
při vrácení originálních disket CONSUL.... sleva 50% (199,-Kč)
-demoverse za cenu diskety (bude odečtena
z ceny objednávky po vrácení) a poštovného: asi 30-50 Kč

Distributor: Incotex, s.s r.o., Hybešova 42, 65664 Brno, tel. 4332 1267
fax. 4321 1234

Základní matematika pro ZŠ

R E C O R D

Program procvičuje základní početní operace formou závodu s časem. Jsou evidovány správně vypočítané příklady v řadě za sebou. Časový limit je možné vypnout. Program je vhodný pro žáky 2.-5. tříd základních škol. Premiově je připojen program pro prvňáčky: REKPRV, obsahující sčítání a odčítání do 10 nebo přes 10 do 20.

Nabídka programu RECORD obsahuje možnosti:

- | | |
|---------------------------------|----------------------------|
| F1 - Násobení - malá násobilka | F6 - Odčítání |
| F2 - Násobení a dělení | F7 - Sčítání a odčítání |
| F3 - Dělení | F8 - Odhad podílu |
| F4 - Násobení - velká násobilka | F9 - Čas (zapnout/vypnout) |
| F5 - Sčítání | Esc- Konec programu |

V nabídce je uveden i počet rekordních bodů, součet rekordů, při počítání je zobrazován jak RECORD tak i průběžné SCORE.

Programy se spustí přímo z diskety zadáním jména programu.

Z Á P O R

Program s podobným principem, v němž lze nastavit několik různých forem zápisu sčítání i odčítání jednociferných nebo dvouciferných čísel. Program je vhodný pro 5. a 6. ročník ZŠ a obsahuje:

- | | |
|-------------------------------|--------------------------------|
| F1 - Sčítání jednociferné | F6 - Odčítání dvouciferné |
| F2 - Odčítání | F7 - Sčítání i odčítání |
| F3 - Sčítání i odčítání | F8 - Sčít.i odčít. různý zápis |
| F4 - Sčít.i odčít.různý zápis | F9 - Čas (zapnut/vypnut) |
| F5 - Sčítání dvouciferné | Esc- Konec programu |

Program je spustitelný přímo z diskety zadáním jména.

D Ě L E N Í

Program pro nácvik písemného dělení s jednoduchou animací po jednotlivých krocích příkladu. Je možno nastavit až šestmístného dělence a trojciferného dělitele. Vhodné pro 4.- 6. ročník ZŠ. Průběžně je registrován počet chyb v příkladu i celkem, počet příkladů vypočítaných správně nebo s chybou a procentní úspěšnost.

Např. pro $1772 : 9 =$

se program zeptá, kolik čísel žák zatrhne, je-li to správně, provede zatrhnutí a přesune kurzor na 1. číslici výsledku, po jejím zapsání se vrátí pod dělence a očekává dílčí zbytek po dělení atd. Chybou číslici sice zobrazí, ale vypíše informaci o chybě a smaže ji. Po ukončení práce je uveden stručný komentář s hodnocením.

Program sestává ze dvou částí (úvodní obrazovky a vlastního programu). Proto se musí spustit pomocí vyvolání dávky: START

Zaváděcí cena do června: jeden program: 390,- Kč + 5% DPH,
dva různé programy: 690,- Kč + 5% DPH,
tři různé programy: 890,- Kč + 5% DPH.

Matematika ZŠ

Program je určen k mechanickému rocvičování matematiky u dětí navštěvujících 1.-4.ročník ZŠ. Učitel nebo rodič žáka může sestavit úlohu obsahující vybrané typy početních úkonů a dítě příklady jednou nebo opakovaně počítá. Výběr úloh obsahuje všechny druhy příkladů předepsané osnovami pro všechny třídy 1.stupně ZŠ. Program postupně generuje konkrétní příklady předepsaných typů, žák je počítá a program jej kontroluje, upozorňuje na chyby a registruje je pro následnou kontrolu, včetně průměrného času počítání, data a času úlohy apod.

Příklady používají data z generátoru náhodných čísel, aby při opakovaném spuštění nebyly opakovány shodné příklady. Výhoda programu proti písemné nebo ústní formě zkoušení je v tom, že se dítě jen soustřeďuje na práci - plynule zadávanými příklady. Normální dítě ve 3.třídě je např. schopno spočítat jeden příklad za 5-6 sekund a být soustředěné asi 10 minut. Za tuto dobu spočítá kolem 100 příkladů, což je počet při jiném způsobu zkoušení nebo opakování v daném čase nedosažitelný.

Na distribuční disketě není program nahrán ve spustitelné formě, je nutno jej instalovat na pevný disk příkazy INSIALL směřovaným na disketovou jednotku s distribuční disketou. Instalační program navrhuje disk C: a adresář \ZSMAI, lze zvolit i jinou specifikaci disku a adresáře.

Hlavní nabídka umožňuje vybrat z připravených úloh, např. od každého typu dva příklady (je v demoversi) a limit malé násobilky 7 sekund. Pak lze: spustit, opravit, vytvořit či zrušit úlohu, potvrdit deník, zvolit režim nastavování, zadání hesla nebo konec práce.

Program obsahuje ochranu proti napadení viry, pokud byl napaden objeví se po spuštění na obrazovce: Program je poškozen. Pak je nutno použít některý antivirový program k ošetření.

Nastavení programu umožňuje: oslovení dítěte, volbu kluk nebo holka, změnu hesla, pauzu mezi příklady a uložení tohoto nastavení.

Doprovodná dokumentace na disketě obsahuje podrobný popis práce s programem včetně seznamu všech učebnic, které odpovídají osnovám pro 1.-4.třidu, a jsou uvedeny u jednotlivých typů příkladů, což je vodítkem spíše pro rodiče než žáky.

Například příklady pro 2.třidu mají možnost těchto typů:

- sčítání v oboru do 20, opakování z 1.třídy,
- odčítání, násobení a dělení ve stejném oboru,
- sčítání v oboru do 100 s přechodem přes 10
- odčítání ve stejném oboru,
- násobky čísel 2 a 3,
- násobky čísel 4, 5 až 8 a 9,
- malá násobilka,
- dělení v oboru malé násobilky.

Není zapomenuto ani na nerovnosti, zaokrouhlování a zlomky

Demoverse je na vratné disketě za cenu diskety a poštovného, závisí na typu požadované diskety. Její cena bude odečtena od hodnoty objednaných programů.

Zaváděcí cena programu pro jeden počítač: 490,- Kč + 5% DPH

Multilicence pro síť počítačů bude k dispozici v září 1994.