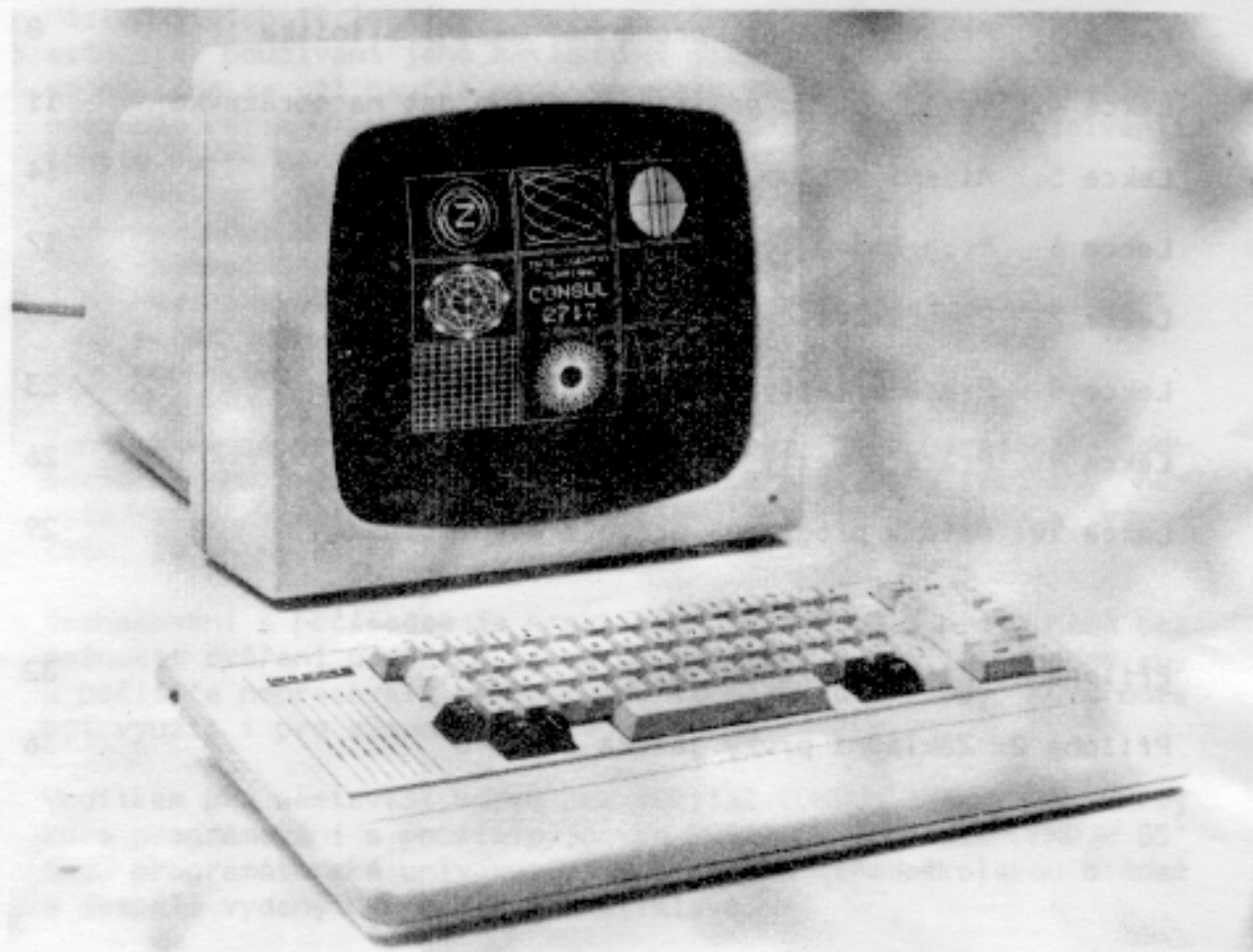


 **CONSUL 2717** 



ÚVODNÍ KURS PRÁCE S POČÍTAČEM

Obsah:

=====

strana:

Lekce 1: Seznámení s počítačem	2
Lekce 2: BASIC - práce v přímém režimu	5
Lekce 3: Program, paměť, proměnné, mazání stínítka	8
Lekce 4: Vstup dat do počítače, výstup dat na obrazovku	11
Lekce 5: Řízení postupu výpočtu	14
Lekce 6: Programový cyklus I	17
Lekce 7: Programový cyklus II	20
Lekce 8: Práce s texty (řetězce) I	23
Lekce 9: Práce s texty (funkce) II	26
Lekce 10: Data v programu, práce s magnetofonem	29
 Příloha 1: Příklady řešení úloh	 32
Příloha 2: Základní prvky jazyka BASIC-G	36

Název: Úvodní kurs práce s počítačem CONSUL 2717
 Sestavil: Ing. Pavel Hlaváček
 Vydal: Incotex, státní podnik,
 Hybešova 42, 65664 Brno

Cena: 7,- Kčs stanovena jako dohodnutá podle výměru
 FCÚ, CCÚ a SCÚ číslo v-6/88, položka 144

UVODNI KURS PRACE S POČITATEM CONSUL 2717:

Následujících 10 lekcí umožňuje seznámení s obsluhou počítače a možnostmi používání jeho základního programového vybavení. Kurs si neklade za cíl naučit programovat v jazyku BASIC-G a nemůže v omezeném rozsahu vysvětlit všechny možnosti počítače. Absolventi tohoto kursu by měli umět:

- ovládat činnost počítače z klávesnice;
- vkládat do počítače publikované programy a data;
- samostatně používat základní příkazy jazyka BASIC-G;
- sestavovat a odzkoušet jednoduché programy v BASICu;
- číst jednoduché programy a porozumět jejich obsahu;
- upravovat programy pro svou vlastní potřebu.

Tento kurs je určen zájemcům o počítač bez předcházejícího odborného vzdělání. Proto je jeho metodika sestavena tak, aby nevyžadovala zvláštní vědomosti z matematiky, informatiky a výpočetní techniky.

Seznamování s počítačem je praktická disciplína a proto nemá bez možnosti ověření poznatků přímo na počítači smysl. Je vhodné, aby u počítače nepracovaly během kursu více jak dvě osoby. Kurs může být využit i pro samostatné studium.

Vodítkem pro sestavení kursu pro počítač CONSUL 2717 byl "Prvý kurs programování s použitím jazyka BASIC na počítači PMD - 85" tzv. programátorské university mladých pro středoškolskou mládež a dospělé vydaný KVTČM SZM v Bratislavě.

TEMA: Seznámení s počítačem a jeho klávesnicí

Lekce: 1

=====

=====

NOVE POJMY: klávesnice, obrazovka monitoru, dialogový řádek

----- kursor, editační klávesy, řídící klávesy

Zapnutí počítače:

- nejprve zkontrolujeme, zda je připojen kabel klávesnice do konektoru počítače na jeho zadní straně, označeného KLAVESNICE; bez tohoto propojení není možno počítač zapnout;
- zkontrolujeme připojení počítače do zásuvky sítě;
- zapneme počítač vypínačem vpravo na jeho zadní stěně;
- zkontrolujeme, zda se vpředu vpravo dole rozsvítí kontrolka;
- po několika sekundách se na obrazovce monitoru zobrazí vlevo nahoře text:

C2717 BASIC-G

a vlevo dole informace o připravenosti počítače přijímat data a příkazy z klávesnice:

OK

Klávesnice a obrazovka:

Klávesnice je podobná klávesnici psacího stroje. Má podobně rozmístěné klávesy s písmeny, číslicemi a dalšími znaky. Pro psaní textu slouží bílé klávesy. Stiskneme-li libovolnou bílou klávesu, počítač zobrazí znak, který této klávese odpovídá, dole vlevo na stínítku obrazovky. Další znaky zadávané z klávesnice se zobrazují vpravo od dříve napsaných znaků, a vytvářejí na stínítku obrazovky řádek textu. Tento řádek se nazývá:

Dialogový řádek

Všechno, co budeme psát na klávesnici, bude zobrazováno v tomto dialogovém řádku, abychom měli informaci o tom, zda jsme text správně zapsali. Do tohoto dialogového řádku vypisuje své informace i řídící program počítače. Lze říci, že tento řádek slouží k dialogu mezi obsluhou počítače a programem v počítači.

Zbývající větší část stínítka obrazovky monitoru nazýváme

Pracovní část obrazovky.

Pomlčka_, která svítí v dialogovém řádku pod OK se nazývá KURSOR. Vypisovaný znak z klávesnice se vždy objeví tam, kde právě svítí kursor. Pro ovládání kursoru při psaní v dialogovém řádku slouží

Editační klávesy

-> posun kursoru vpravo;
<- posun kursoru vlevo;
\ posun kursoru na začátek řádku, tzv. HOME pozice;
END posun kursoru na konec textu;
!<- posun obsahu dialogového řádku vlevo vzhledem ke kursoru;
->! posun obsahu dialogového řádku vpravo vzhledem ke kursoru;
DEL vymazání znaku na místě kursoru, text zprava se přisune;
INS vložení mezery do textu na místě kursoru;
CLEAR vymazání celého dialogového řádku;
! přemykač (shift), umožňuje psaní horních znaků kláves a malá písmena; je nutno jej držet současně s klávesou;
!CLEAR vymazání textu vpravo od kursoru;
EOL ukončení psaní textu (End Of Line); obsah dialogového řádku se předá počítači ke zpracování a zobrazí se do pracovní části obrazovky; je-li text bez chyby zobrazí se kursor
RCL umožňuje přivolat zpět (ReCall) právě odeslaný řádek, a opravit jej, pokud v něm řídící program zjistil chybu;
PRINT zruší vypisování textu do pracovní části obrazovky (nutno použít přemykač, neboť je to horní symbol klávesy).

Systémové klávesy:

!RESET znovunastavení počítače do výchozího stavu po havarii programu; červená klávesa je účinná pouze s přemykačem;
STOP zastavení právě prováděného programu; nutno podržet až do výpisu zprávy o zastavení: ** Stop at line NN **
C-D znovuvylolání poslední systémové zprávy řídícího programu

Cvičení:

1. Napište do dialogového řádku postupně všechny bílé znakové klávesy (číslice, velká písmena a dolní znaky):
1234567890_QWERTZUIOP@^ASDFGHJKL;:[]XCVBNM,./

2. Pokračujte v psaní se stlačeným přemykačem (horní znaky a malá písmena):
!"#\$%&'()-qwertyuiop^asdfghjkl+*]xcvbnm<>?

Pozorujte, jak se text posouvá vlevo, pokud dosáhl kurzor pravý okraj dialogového řádku.

Dialogový řádek má celkem 80 znaků, viditelných je pouze 48. Dosáhne-li kurzor pozici 70., oznámí to počítač pípnutím.

3. Dopište řádek až do konce (text se přestane posouvat vlevo). Stiskem klávesy EOL odešlete text ke zpracování počítači.

Přečtete si v dialogovém řádku zprávu o chybě.

+ + + Syntax err + + +

Kdykoli se v budoucnu tato zpráva objeví, musíte hledat chybu.

Použitím klávesy RCL si vyvolejte zpět napsaný řádek textu.

Vyzkoušejte si na něm funkci editačních kláves: <-, ->, DEL, \ (HOME), INS, !<-, ->!, !CLEAR, CLEAR.

4. Napište svoje jméno a adresu střídáním velkých a malých písmen. Celou adresu dejte do uvozovek, a před ní napište velkými písmeny příkaz PRINT. Například:

PRINT"Josef Halva, Sovinec 76, 62800 Brno"

Odešlete tento text stiskem EOL a pozorujte, co se vypsalo do pracovní části obrazovky. Zkuste si takto vypsát na stínítko další texty. Použijte k tomu zpětné vyvolání: RCL. Potlačte opisování dialogového řádku pomocí přemykače a PRINT.

5. Zkuste si funkci opakování jednotlivých kláves tím, že je podržíte stlačené delší dobu.

TEMA: BASIC - práce v přímém režimu tohoto jazyka
===== Používání matematických operátorů a funkcí

Lekce: 2
=====

NOVE POJMY: aritmetické operátory a výrazy, matematické funkce,
----- příkaz PRINT pro zobrazení informace na stínítku

NOVE PRIKAZY: PRINT výraz
----- PRINT "text"

B A S I C

je programovací jazyk konverzačního typu, umožňující uživateli spolupracovat s počítačem v dialogovém režimu. Uživatel zadává počítači příkazy tohoto jazyka a počítač je okamžitě provádí. Objeví-li počítač chybu v zadání příkazu, okamžitě to oznámí, aby bylo možno chybu napravit. Anglická zkratka jazyka má význam

B - beginner's	- pro začátečníky
A - all purpose	- vícúčelový
S - symbolic	- symbolický
I - instruction	- instrukční
C - code	- kód

Příkazy jazyka tvoří soubor anglických sloves a zkratek. Podle typu počítače jich může být několik desítek nebo mnohem více. Základním příkazem pro zobrazení informace z počítače na stínítko je příkaz (povel)

PRINT (v překladu znamená tisknout).

Nejprve se naučíme používat počítač jako kalkulačku. Znaky, které slouží k aritmetickým operacím nazýváme aritmetické operátory. Jsou to:

+	pro sčítání:	například: 4+2, A+8, 3.14159+X
-	pro odečítání:	9-1, X-Y, 0.99-0.45
*	pro násobení:	2*5, 7*(A+D), Y*1.414
/	pro dělení:	1/3, H/(9-T), 3.14159/X
^	pro umocňování:	2^5, K^2.44, Q^(1+R)

Všimněte si, že v příkladech jsou místo desetinných čárek používány desetinné tečky. Tuto skutečnost je nutno respektovat. Dále je nutno zapamatovat si znaménko pro umocňování, protože příkazy počítači můžeme psát jen na jeden řádek, nikoli do exponentu. Při zápisu složitějších výrazů je vhodné používat kulaté závorky i několikanásobně (jiný typ závorek ve výrazech není dovolen).

Zkuste si nyní vypočítat následující výrazy (a nezapomeňte na konci zápisu výrazu stisknout klávesu EOL (to je: předat zprávu počítači):

PRINT 5+6	a 'EOL'	výsledek:	11
PRINT 6*(7.1+8.7)	- " -		94.8
PRINT 3.33*(5*(7+9))	- " -		266.4
PRINT 8*(17-72)/(4*(2/3))	- " -		-165
PRINT 2^(3*3)/2^(16/2)	- " -		2

Pro výpočty matematických funkcí nemá klávesnice počítače další klávesy a je proto nutno používat anglické zkratky a za nimi v závorce číslo nebo výraz, pro jehož hodnotu funkci počítáme (odborně se výrazu v závorce říká argument funkce):

SQR (X)	-druhá odmocnina:	PRINT SQR(25)	5
ABS (X)	-absolutní hodnota:	PRINT ABS(-77)	77
LOG (X)	-Přirozený logaritmus:	PRINT LOG(2.71829)	1
EXP (X)	-exponenciální funkce: e^X	PRINT EXP(2)	7.38906
SIN (X)	-sinus:	PRINT SIN(0)	0
COS (X)	-cosinus:	PRINT COS(0)	1
TAN (X)	-tangens:	PRINT TAN(1)	1.55741
ATN (X)	-arcustangens:	PRINT ATN(9)	1.46014

Zkuste si vypočítat hodnoty uvedených funkcí. Argumentem funkce může být i složitý aritmetický výraz. Zkuste vypočítat:

PRINT SQR (3^2+4^2)	výsledek:	5
PRINT LOG (EXP(3.2))		3.2
PRINT EXP (COS(0))		2.71828

Pozor: Úhly goniometrických funkcí můžeme zadávat v radianech nebo ve stupních. Počítač předpokládá, že je budeme zadávat v radianech. Chceme-li zadávat úhly ve stupních, musíme to počítači sdělit použitím příkazu: DEG (DEGREE=stupně). Zkuste zadat počítači následující úkoly:

DEG:PRINT SIN (45)	výsledek:	.707107
PRINT TAN (30)		.57735
RAD:PRINT SIN (45)		.850902
PRINT TAN (30)		-6.40536

Pomocí příkazu RAD (radian) jsme sdělili počítači, že bude další výpočty provádět v radianech. Poprvé jsme použili dva příkazy na jednom řádku. Pro jejich oddělení je určen symbol dvojtečky ':'. Nyní už asi chápete, proč je pro operaci dělení určeno lomítko.

Pomocí příkazu PRINT můžeme psát na stínítko nejen čísla, ale i texty, jak jste si to zkusili již v první lekci. Protože při předcházejících příkladech nám počítač sdělil na stínítku pouze výsledek výpočtu, ale současně smazal zadání na dialogovém řádku, pokusíme se, přikázat mu, aby zobrazoval zadání i výsledek.

Zkuste zadat:

PRINT "3+5=";3+5	výsledek: 3+5= 8
PRINT "1/3=";1/3	1/3= .333333
PRINT "SQR (9)=";SQR(9)	SQR (9)= 3

To co je uvedeno v uvozovkách se pouze opiše na obrazovku, počítač to považuje za text. Proto mu za textem musíme ještě jednou napsat totéž bez uvozovek - to již považuje za příkaz k výpočtu.

Zkuste si v posledních příkazech nahradit středník čárkou, a porovnejte výsledky na obrazovce - vypočtené hodnoty budou posunuty více vpravo. Čárka má v příkazu PRINT funkci tabelátoru na pozicích 1,15,29,43.

Potřebujeme-li ve vypisovaných zprávách vynechat řádek, napíšeme samostatný příkaz PRINT.

Cvičení:

1. Vyzkoušejte si operace a funkce uvedené v této lekci pro jiné číselné hodnoty.
2. Vypočtete: $10(5+6)$, $15(5+7(10-8))$, $3+4(3:(4-15))$, $\sin 45.\cos 70$
 $7:((5*6)-7(8-7))$, $[\ln 128(15+78)]:[(15/34+2):4*.2]$
3. Napište na stínítko nadpis: Jmeno:.....Telefon:
Podtrhněte nadpis pomlčkami:-----
Zkuste seznam vyplnit: Jirka 332211
Použijte čárku jako tabelátor: PRINT"Jmeno",,"Telefon:"
Napište podobné jména a data narození rodičů a sourozenců.
4. Vyzkoušejte si spojení textů pomocí operátoru '+':
PRINT "Jaro"+"slav":PRINT,"MIRO"+SLAV":PRINT,,"LADI"+"SLAV"
Pozor na uvozovky a čárky. Dokážete výsledek zdůvodnit ?

TEMA: Program, paměť, proměnné, mazání stínítka
=====

Lekce: 3
=====

NOVE POJMY: Příkazy, program, proměnná, přiřazení hodnoty,
----- spuštění programu, jeho prohlížení a opravy

NOVE PŘÍKAZY: GCLEAR, END, RUN, LIST, LLIST, LET

Doposud jsme počítači zadávali jednoduché příkazy, které se po stlačení klávesy EOL ihned provedli. To je tzv. přímý režim práce v jazyku BASIC. V tomto jazyku však převažuje tzv. programový režim, kdy jsou zapsané příkazy zapamatovány v paměti jako program, a počítač je provede, až mu k tomu dáme pokyn. Pro rozlišení obou režimů bylo autory jazyka zavedeno číslování řádků pro programový režim. Program v jazyku BASIC potom tvoří posloupnost číslovaných řádků uložených v paměti počítače. Při zapisování programu se interpret jazyka BASIC postará o to, aby řádky programu byly uloženy v paměti od řádku s nejnižším číslem až po řádek s číslem nejvyšším. Pokud zapomeneme některý řádek napsat, nebo napsaný program opravit vložením jednoho či více řádků, napíšeme tento řádek a BASIC si jej zařadí podle jeho čísla na správné místo v paměti programu. Abychom to mohli udělat kdykoli, musíme na vložení řádků pamatovat při jejich číslování. Proto se doporučuje číslovat řádky např. s krokem 10 (10,20,30,...).

Zkusíme si napsat jednoduchý program:

10 GCLEAR	a stiskneme 'EOL'
20 PRINT "Nas první program:"	---
30 PRINT "A=", "B=", "A+B="	---
40 PRINT 12,34,12+34	---
50 END	---

V tomto programu se objevily dva nové příkazy jazyka BASIC. Jsou to: GCLEAR (vymazání stínítka obrazovky) a END (konec programu). Program je uložen v paměti počítače a čeká, co s ním jeho autor bude dělat. Nejlepší je takový program okamžitě uložit na kasetu - i když v něm mohou být chyby, zůstane někde uchován k dalšímu použití, zatímco spuštění programu může někdy vést k jeho zničení. Příkaz k uložení programu na magnetofon si vysvětlíme později. Předpokládejme, že náš program je bez chyby, a proto se jej pokusíme spustit pomocí příkazu: RUN (běžet). Předání tohoto příkazu počítači musíme zajistit klávesou 'EOL'.

Zkusili jste to? Spokojeni? Upravíme si program podtrhnutím názvu, tzn., že musíme mezi řádky 20 a 30 napsat nový řádek, označený např. číslem 25 (nebo kterýmkoli od 21 do 29):

```
25 PRINT "-----"          a stiskneme 'EOL'
```

Takto upravený program opět spustíme příkazem RUN (a EOL). Určitě souhlasíte s tím, že by program měl pracovat na čisté obrazovce, abychom výsledky jeho činnosti nemuseli hledat. Proto si zvykněte 'mazat' (GCLEAR) obrazovku vždy před výpisem výsledků.

Nyní si náš program zkusíme vypsát na obrazovku. Pro tuto práci je určen v BASICu příkaz LIST (seznam). Napište jej a uvidíte, že jste zapomněli smazat obrazovku? Nevadí. Napravíme to jednoduše. Napišeme příkaz smazání GCLEAR (a EOL) a potom příkaz LIST (EOL). Oba příkazy můžeme napsat na jeden řádek, ale musíme je oddělit dvojtečkou:

```
GCLEAR:LIST
```

Možná, že jste ve výše uvedeném programu udělali chybu a musíte jej opravit. Chybný řádek je nutno vyvolat na stínítko do dialogového řádku a tam opravit. Pro vyvolání řádku je určen příkaz LLIST, za kterým jako jeho parametr následuje číslo řádku. Není-li toto číslo uvedeno, vypíše se první řádek programu, po stisku EOL následující řádek, atd. až po poslední řádek. Zkuste si to.

Nyní si zkusíme opravit náš první program tak, aby počítal odmocninu ze součinu čísel A a B. K tomu použijeme LLIST 30 (EOL) a zobrazený řádek pomocí kláves pohybu cursoru, INS a DEL upravíme takto:

```
30 PRINT "A=", "B=", "SQR(A*B)="          a stiskneme EOL
```

Ještě upravíme řádek 40 - tisku výsledku - podobným způsobem na:

```
40 PRINT 12,34,SQR(12*34)                a opět stiskneme EOL.
```

Vložíme si ještě jeden řádek, například:

```
45 PRINT 0.3,0.3,SQR(0.3*0.3)
```

a upravený program spustíme příkazem RUN (že je nutno vždy po opravě řádku, napsání dalšího řádku nebo nového příkazu stisknout EOL si už pamatujeme, a proto už to nebude zdůrazňováno).

Zkusme si nyní náš program doplnit a upravit tak, aby místo konstant počítal s libovolnými zadanými čísly. Tato čísla musíme zapsat někde na dohodnuté místo v paměti počítače, kde si je program najde. Dohodnutá místa označíme písmeny velké abecedy. Protože se jejich obsah může měnit, nazývají se 'proměnné'. Pokud by nám nestačila jednoduchá písmena, můžeme k názvu proměnné připojit ještě číslici nebo další písmeno, např. A3, BX, F0, KK, Z9. Bylo by možno napsat i delší názvy, ale počítač by pracoval jen s prvními dvěma znaky, a mohlo by se mu poplést BRNO a BREZNO.

Aby si počítač zapamatoval hodnotu proměnné, musíme mu to přikázat pomocí příkazu LET, následovaného názvem proměnné, rovnítkem a hodnotou, kterou této proměnné přiřazujeme:

```
LET A=7.23:LET B=9:LET C=SQR(A*B)
```

Ale protože psaní LET bylo jaksi navíc, jazyk BASIC připouští i možnost jeho vynechání. Tak si zkusme doplnit náš program o proměnné, které budeme definovat zkráceným způsobem (bez LET):

```
15 A=7.23:B=9:C=SQR(A*B)
35 PRINT A,B,C
```

Spustíme-li nyní program, tak nám vypočítá zadané, ale navíc nám napíše ještě obsahy řádků 40 a 45. Protože nás v dalším již nebudou zajímat, vymažeme je z programu takto: napíšeme číslo řádku a stiskneme EOL. Před tím je vhodné vypsát program příkazem LIST, abychom si byli jisti, které řádky to skutečně jsou. Tak si to zkuste sami bez další nápovědy.

Navíc Vám asi na řádku 15 neuniklo, že můžeme proměnnou definovat pomocí funkce a jiných proměnných.

Cvičení:

-
1. Napište program na výpočet obvodu trojúhelníka o stranách: A=4, B=8, C=12. Obvod se vypočítá jako součet: $OB=A+B+C$
A nezapomeňte popsat program a proměnné na obrazovce.
 2. Napište program pro výpočet přepony pravoúhlého trojúhelníka: $C=SQR(A^2+B^2)$, pro A=6, B=8 a popište výsledky na obrazovce.

TEMA: Vstup dat do počítače,
===== výstup dat na obrazovku.

Lekce: 4
=====

NOVE POJMY: Příkaz vstupu dat, výstup dat do dialogového řádku,
----- editace výstupu dat na obrazovce.

NOVE PŘIKAZY: INPUT A1,A2,...,An kde A1,...,An jsou proměnné
----- DISP výraz kde výraz=proměnná,text aj
PRINT AT R,S kde R=řádek, S=sloupec

Jednoduché programy, které jsme se naučili v minulé lekci, měly vždy stejný výsledek, byly určeny na jedno použití. Pokud jsme chtěli počítat s jinými hodnotami proměnných, bylo nutno změnit program (opravit některý řádek tohoto programu).

Vhodnější než upravování programu je načtení vstupních dat během činnosti programu z klávesnice počítače. Tím vytvoříme univerzálnější obecně použitelný program.

Pro vstup dat má BASIC příkaz INPUT (anglicky: vstup). Za jméno tohoto příkazu je nutno uvést jednu nebo více proměnných, do kterých se hodnoty zadané z klávesnice uloží. Zkusíme si to na jednoduchém programu pro vstup dat:

```
10 GCLEAR
20 INPUT K
30 PRINT K;K+K,K*K,K^K
40 END
```

Program spustíme příkazem RUN. Příkaz na řádku 10 smaže obrazovku, příkaz INPUT vypíše do dialogového řádku kurzor '_' a čeká, dokud na klávesnici nenapišeme číslo (ukončené EOL). Číslo, které jsme napsali, se přiřadí proměnné 'K' uvedené v příkazu INPUT. Novou hodnotu proměnné program použije k výpočtům na řádku 30.

Je to jednoduché, ale co když zapomeneme, na kterou hodnotu program čeká po vypsání otazníku? Proto je dobré, když autor programu myslí i na jeho budoucího uživatele, a napoví mu vhodným způsobem, co očekává, když se v dialogovém řádku objeví otazník.

Podobným způsobem můžeme upravovat (editovat) i výstup dat na obrazovce. Zkuste si program pomocí 'PRINT AT R,S' upravit tak, aby správně nad sebe psal jak 'záhlaví' výpočtu, tak i výsledek. Aby to nebylo tak jednoduché, zkuste to uprostřed obrazovky.

Další možností editace výstupních dat je použití funkce TAB(S), která v příkazu PRINT (nebo DISP) umožní vynechat S sloupců (pozic) v textu. Protože je na řádku 48 znaků, může být $S < 47$. Použití této editační funkce si ukážeme na dalším programu:

```
10 GCLEAR:R=5
20 PRINT AT 25,0;"Zadej K a P (1-999), oddel je carkou !"
30 INPUT K,P:PRINT AT 25,0,,,
40 PRINT AT R,5;"K=";TAB(10);"P=";
50 PRINT TAB(20);"K-P=";TAB(30);"K/P="
60 PRINT AT R+2,5;K;TAB(10);P;
70 PRINT TAB(20);K-P;TAB(30);K/P
80 END
```

V tomto programu jsme využili možnosti rozdělit dlouhý výraz za příkazem PRINT na dva programové řádky (50+60,70+80). Je to možné tak, že výraz na prvním z obou řádků je ukončen středníkem.

Na řádku 10 jsme ve druhém příkazu přiřadili proměnné R hodnotu 5. Tato proměnná je pak využita ve funkci 'PRINT AT' pro určení řádku, na který se bude vypisovat záhlaví výpočtu. Dále je použita pro určení řádku výpisu výsledku - zvětšená o hodnotu 2, jak je uvedeno na řádku 60.

Cvičení:

1. Zkuste, co se stane, zadáte-li jen jednu proměnnou (a EOL), a potom další proměnnou (a EOL).
2. Sestavte program na výpočet obvodu, povrchu a objemu koule ze zadaného poloměru R pomocí INPUT. Pro výpočet použijte hodnotu $PI=3.14$, obvod je $OB=2*PI*R$, povrch je $S=4*PI*R^2$ a objem je $V=(4/3)*PI*R^3$.
3. Sestavte program na výpočet průměrné hodnoty ze 4 zadaných čísel pomocí INPUT.
4. Sestavte program na výpočet povrchu [$S=2*(A*B+A*C+B*C)$] a objemu kváдру [$V=A*B*C$] z hodnot zadaných pomocí INPUT.

TEMA: Řízení postupu výpočtu.

Lekce: 5

NOVE POJMY: Podmíněné skoky, nepodmíněné skoky, relační
operátory, zastavení výpočtu, pokračování výpočtu.

NOVE PŘIKAZY: GOTO číslo řádku
IF podmínka THEN příkaz
STOP
CONT
REM

V praxi se mnohokrát setkáváme s úlohami výpočtu hodnot tabulek, ve kterých se podle změny některé vstupní proměnné mění i závislé hodnoty podle definovaných funkcí. Jako příklad si sestavíme tabulku výpočtu přepony C pravoúhlého trojúhelníka s odvěsnami A a B, a přidáme i výpočet plochy $S=A*B/2$.

```
10 GCLEAR:PRINT"Výpočet C a S pravoúhlého trojúhelníka"
20 PRINT:PRINT" A= "," B= ","Přepona C=","Plocha S="
30 DISP"Zadej délku strany A":INPUT A
40 DISP"Zadej délku strany B":INPUT B
50 C=SQR(A*A+B*B):S=A*B/2
60 PRINT A; B; C; S
70 STOP: GOTO 30
```

V řádku 20 jsme použili samotný PRINT (bez výrazu), který vypíše prázdný řádek a oddělí název programu od záhlaví tabulky. Požadované hodnoty vypočítáme na řádku 50 a vypíšeme řádkem 60. Na řádku 70 jsou dva nové příkazy: STOP (zastav) a GOTO 30 (jdi na řádek číslo 30). Po spuštění programu, zadání hodnot a výpočtu se do dialogového řádku vypíše zpráva:

+ + + Stop at line 70 + + + (zastavení na řádku 70).

Po zastavení programu si můžeme zkontrolovat, zda je na obrazovce všechno v pořádku (tj. zda program pracuje správně) a potom pokračovat v programu tak, že zadáme příkaz CONT (CONTinue = pokračovat). Účinek příkazu 'GOTO 30' se projeví v tom, že program skočí na řádek 30 - a můžeme zadávat další strany trojúhelníka.

Tímto způsobem jsme vytvořili nekonečnou smyčku v programu, neboť se budou trvale opakovat řádky 30-40-50-60-70 (zde se program zastaví a pokračuje po zadání CONT). Nekonečnou smyčku bez zastavování získáte tak, že na řádku 70 vypustíte příkaz STOP. Vytvoříte tabulku stran a ploch trojúhelníků. Program lze kdykoli přerušit klávesami STOP+EOL (současně stisknutými).

POZOR: mezi STOP a CONT nesmíme zadávat jiné příkazy, jinak počítač napíše: + + + Can't cont + + + (nelze pokračovat).

Vícenásobné použití příkazu STOP v rozsáhlejším programu umožňuje kontrolovat, zda program pracuje správně (ladění programu).

Příkaz GOTO lze použít i pro spuštění programu od zadaného řádku programu, podobně jako příkaz RUN. Rozdíl mezi oběma příkazy si vyzkoušíme na našem programu. Spustíte program pomocí GOTO 10. (POZOR: Nelze napsat samotné GOTO, počítač by nevěděl, kam jít!) Zadejte hodnoty stran a po výpisu výsledků a zastavení zadejte příkazem GOTO 60 opakování výpisu výsledků. Vypsaly se řádek stejných hodnot jako v předchozím řádku. Zadejte 'CONT', hodnoty stran A,B a po výpisu výsledku a zastavení zadejte RUN 60. Nevypsaly se hodnoty stejné, ale nulové (RUN zrušil přiřazení).

Zkušenost: RUN ruší přiřazení hodnot proměnných v programu.

Nyní změníme řádek 70 nepodmíněného skoku GOTO 30 na podmíněný skok: IF podmínka THEN příkaz, a to takto:

```
70 IF A>0 THEN GOTO 30
```

Protože příkaz GOTO za THEN je nepovinný, nebudeme jej psát. Pak

```
70 IF A>0 THEN 30
```

V angličtině IF značí 'když' nebo 'jestliže' a THEN má význam 'tak' nebo 'potom'. Příkaz na řádku můžeme interpretovat jako: Když bude proměnná A větší než nula, tak pokračuj na řádku 30. Když proměnná A bude rovna nule nebo <0 (podmínka nesplněna), jdi na následující programový řádek: 80 END (napíšeme si jej).

Spustíme program, zadáme nejprve strany A>0 a B>0 a při dalším požadavku zadání stran vyzkoušíme hodnotu A=0. Vypíše se výsledek s A=0, zadaným B... a program se ukončí zprávou OK v dialogovém řádku.

Znak '>', který jsme použili v podmínce, patří mezi tzv.

Relační operátory:

>	je větší než:	$X > Y$
<	je menší než:	$SIN(U) < 1/3$
=	rovná se:	$A+B+C = 99$
<>	nerovná se:	$SQR(P*Q) <> R^{1/4}$
<=	je menší nebo rovno:	$D <= EXP(3)$
>=	je větší nebo rovno:	$TAN(U) >= 1.5$

POZOR: Je nutno vždy dodržovat pořadí u operátorů '<=' a '>=' !

Pomocí podmíněného příkazu IF...THEN... můžeme dosáhnout větvení programu, srovnávání proměnných nebo výsledků výpočtů se zadanými hodnotami nebo výrazy. Uvedeme si jednoduchý program srovnání dvou proměnných.

```
10 GOTO 10: REM Program srovnání dvou čísel
20 DISP "Zadej 2 čísla <> 0 (0=konec programu)"
30 INPUT A,B: IF A=0 THEN END
40 IF A>B THEN PRINT A;" >";B:GOTO 20
50 IF A=B THEN PRINT A;" =";B:GOTO 20
60 PRINT A;" <";B:GOTO 20
```

Na řádku 10 jsme použili příkaz REM (REMark = poznámka), který umožňuje komentovat program nebo jeho řádky. BASIC jej neprovádí - 'vynechá' ho, stejně jako příkazy následující za ním na stejném řádku programu. Za THEN můžeme psát nejen číslo řádku programu, ale i jiné příkazy, nebo soubor příkazů oddělených ':':

Cvičení:

1. Napište program, který zjistí, kolik za sebou jdoucích celých čísel od 1 je nutno spočítat, aby součet byl větší než 100.
2. Napište program, který zjistí průměrnou známku z 10 předmětů, zařadí žáka podle stupnice:
 - prospěl s vyznamenáním.....průměr do 1.5
 - prospěl velmi dobře průměr do 2.0
 - prospěl průměr nad 2.0
 - neprospěl (propadl) některá známka je 5

TEMA: Programový cyklus

Lekce: 6

=====

=====

NOVE POJMY: Cyklus, počítadlo opakování, příkaz FOR...NEXT,
----- vnořené cykly, logické operátory, semilogaritmicky
 tvar čísla

NOVE PRIKAZY: FOR počítadlo=od TO konec [STEP přírůstek]
----- NEXT počítadlo
 AND

Velmi často je potřebné opakovat několikrát určitou část programu. Pro opakování musíme v programu zadat především co je třeba opakovat a kolikrát (dokdy) je to nutné opakovat. Pro tento účel má BASIC vyhrazenou dvojici příkazů FOR... a NEXT. Opakují se pak všechny příkazy, které jsou v programu umístěny mezi řádkem FOR a řádkem odpovídajícího NEXT. Proto se někdy říká, že FOR a NEXT vytvářejí tzv. programové závorky.

Za příkazem FOR (pro) je nutno uvést proměnnou, která bude sloužit jako počítadlo opakování. Nejprve musí být zadána počáteční (výchozí) hodnota. Za další částí příkazu, kterou tvoří slůvko TO (až do) je nutno uvést konečnou hodnotu proměnné ve funkci počítadla. Počítadlo si připočte další hodnotu vždy, když narazí program na odpovídající příkaz konce cyklu NEXT. Standardně se připočítává +1. Pokud vyžadujeme jiný přírůstek (nebo úbytek), musíme to zapsat za další část příkazu nazvanou STEP (krok).

Zadejte počítači následující program a porovnejte výsledky:

```
10 GCLEAR:REM Příklad na programovy cyklus
20 FOR I=1 TO 10:PRINT I;:NEXT I:PRINT
30 FOR I=-1 TO -10 STEP -1:PRINT I;:NEXT I
40 END
```

Program na řádku 20 vypíše 10 čísel oddělených 2 mezerami. Jedna mezera je oddělovací a druhá je místem pro znaménko. Pokud je číslo kladné, tak se znaménko nepíše. Potvrzuje to výsledek práce programového řádku 30. Výpis hodnot vedle sebe umožňuje ';' jako oddělovač. Protože je i za poslední číslici, museli jsme oddělit výpis záporných čísel prázdným příkazem PRINT (řádek 20). Zkuste jej vynechat, a uvidíte.

S proměnnou, která je počítadlem opakování cyklu, můžeme pracovat jako s každou jinou. Zkusíme si vytvořit tabulku mocnin čísel pomocí programového cyklu:

```
10 GCLEAR:REM Tabulka mocnin desetinných čísel
20 PRINT " A "; " A*A "; " A^2 "; " A*A*A "; " A^3 "
30 FOR A=0.1 TO 2.5 STEP 0.1
40 PRINT A;A*A;A^2;A*A*A;A^3
50 NEXT A:END
```

Spustíme program a vidíme: Nevýznamné nuly před desetinnou tečkou se nevypisují, to znamená, že je nemusíme vypisovat ani my, jestliže zadáváme čísla z intervalu [0,1]. Velmi malá čísla se zobrazují v semilogaritmickém tvaru: znaménko, mantisa, znak E, znaménko exponentu a exponent (2 číslice). Mantisa může mít max. 6 platných číslic. V některých případech se liší mocnina čísla od jeho násobků. To je dáno tím, že mocnina je počítána jiným (méně přesným) způsobem. V semilogaritmickém tvaru se zobrazují i čísla, která jsou >999999. Zkuste si to změnou 30. řádku:

```
30 FOR A=20 TO 500 STEP 20
```

Mezi příkazy v cyklu se může vyskytnout i další opakování, neboli vnořený cyklus (cykl v cyklu). Je samozřejmé, že pro tyto cykly musíme použít různé proměnné, a také zajistit, aby vnořený cykl byl ukončen dříve než vnější cykl. Ukážeme si to na programu:

```
10 GCLEAR: REM Příklad vnořených cyklů
20 FOR X=1 TO 5:PRINT AT X*X,0
30 PRINT,X;". cyklus"
40 FOR Y=1 TO 15:PRINT X;:NEXT Y
50 NEXT X:END
```

Tam, kde ve FOR_TO_ píšeme čísla, lze psát i aritmetické výrazy:

```
10 GCLEAR: REM Matematické výrazy řídí cykl
20 FOR R=1 TO 8
30 FOR S=1 TO R*2-1:PRINT"*";:NEXT S
40 FOR S=R*2+1 TO 31:PRINT"-";:NEXT S
50 PRINT:NEXT R
60 FOR R=9 TO 16
70 FOR S=1 TO 33-R*2:PRINT"*";:NEXT S
80 FOR S=34-R*2 TO 30:PRINT"=";:NEXT S
90 PRINT:NEXT R:END
```

Jako další příklad si uvedeme úkol, který snad nelze 'ručně' ani vyřešit: Máme 100 Kčs a máme nakoupit tužky T (1,-/kus), sešity S (5,-/kus) a propisovačky P (7,-/kus). Celkem všeho musí být 50 kusů (poznámky REM nemusíte opisovat, komentují program):

```
10 GCLEAR: REM Slozity ukol nakupu
20 FOR P=1 TO 14 : REM 100,-/7 = max 14 propisek
30 FOR S=1 TO 20 : REM 100,-/5 = max 20 sesitu
40 FOR T=1 TO 50 : REM Vseho muze byt jen 50 ks
50 IF(P*7+S*5+T*1=100) AND (P+S+T=50) THEN 70
60 NEXT T:DISP"Pocitam: P=";P;" S=";S:NEXT S:NEXT P
70 PRINT"Propisek bude:";P;"Sesitu:";S;"Tuzek:";T
80 END
```

Možná jste se při zadávání programu pozastavili nad zprávou řádku 60, ale po spuštění programu jste zjistili, že je to důležitá informace pro obsluhu - když má počítač plno práce, na stínítku se nic neděje a může se zdát, že se počítač porouchal.

Protože v zadání úkolu byly dvě podmínky, museli jsme jejich současné splnění formulovat pomocí logického operátoru AND (logický součin). Dalšími logickými operátory jsou OR (log.součet) a NOT (log.negace). Naučíme se je používat později.

Cvičení:

1. Napište program pro výpočet faktoriálu zadaného čísla, jestliže faktoriál čísla N je: $N! = 1*2*3*4*...*N$
2. Napište program, který si vyžádá daný počet čísel, vypíše je, najde a vypíše největší z nich.
3. Upravte program z 2.zadání na nalezení největšího a nejmenšího čísla.
4. Sestavte program pro výpočet tabulky malé násobilky (od $1*1$ do $10*10$).

TEMA: Programový cyklus

Lekce: 7

=====

=====

NOVE POJMY: automatické číslování řádků, pseudonáhodná čísla,
----- zaokrouhlení reálných čísel, indexované proměnné,
pole proměnné, rezervování místa v paměti.

NOVE PŘIKAZY: AUTO od,krok

----- RND (1)

INT (výraz)

DIM A(Imax-1) kde Imax je počet prvků pole A(I)

Přímý příkaz AUTO slouží k automatickému číslování řádků během zapisování programu. Urychluje a zjednodušuje práci programátora. Může mít formát samotného příkazu bez parametrů - potom po jeho napsání (a EOL) vypíše do dialogového řádku číslo 10, a očekává zadání příkazů tohoto řádku. Po odeslání řádku do paměti počítače se počítadlo řádků automaticky zvětší o krok 10, tj. vypíše do dialogového řádku číslo 20 atd.

Druhý formát příkazu - AUTO od,krok - do dialogového řádku bude přidělovat čísla řádků počínaje parametrem 'od' a bude je zvětšovat o parametr 'krok'. Pokud některý parametr ne zadáme, počítač za něj automaticky dosadí 10.

Funkci příkazu si vyzkoušejte např. při zápisu dalšího programu.

Zadáme nejprve

AUTO 5,5 (a EOL) a potom napíšeme následující program:

```
5 GCLEAR:REM Využití počítadla cyklu k editaci
10 FOR I=0 TO 25
15 PRINT AT I,I;"*";AT I,47-I;"*"
20 NEXT I
25 END
```

Automatické číslování řádků lze zrušit klávesou CLEAR a zadáním libovolného příkazu, např. RUN, LIST, LLIST. Pomocí AUTO od,krok můžeme vymazat i část programu, pokud má pravidelný přírůstek čísel řádků. Zkuste si to na našem programu po zadání AUTO 10,5 odpovězte na vypisovaná čísla pouze stiskem EOL a po zprávě OK zadejte LIST. Výsledek je zřejmý z výpisu zbytku programu.

Funkce RND(X) generuje náhodné číslo z otevřeného intervalu 0-1. Je-li argument X=0, vygeneruje se vždy stejné náhodné číslo, je-li argument X=1, vygeneruje se vždy jiné číslo. Přesvědčte se o tom pomocí následujícího programu; použijte i příkaz AUTO 10,2:

```
10 GCLEAR:REM Generování náhodných čísel
12 PRINT"RND(0).":FOR I=1 TO 7:PRINT,RND(0):NEXT I
14 PRINT"RND(1).":FOR I=1 TO 7:PRINT,RND(1):NEXT I
16 PRINT" RND(0)   po   RND(1)"
18 FOR I=1 TO 7:PRINT RND(0),RND(1):NEXT I
20 END
```

Zkušenost získaná z práce programového řádku 18: funkce RND (0) zopakuje náhodné číslo, vygenerované předchozí funkcí RND (1).

Desetinná čísla z intervalu (0,1) můžeme převést na náhodná čísla z jiného intervalu (A,B) tak, že použijeme výpočet pomocí vztahu:

$$(B-A) * \text{RND} (1) + A$$

Přesvědčte se o tom pomocí jednořádkového programu (A=10,B=15):

```
FOR I=1 TO 10:PRINT(15-10)*RND(1)+10:NEXT I
```

Zaokrouhlení reálných čísel umožňuje funkce INT (X), (INTEger=celý) jejímž výsledkem je celé číslo, které je menší nebo rovno argumentu X. Například si zkuste:

PRINT INT (3.14159)	vypíše:	3
PRINT INT (-4.4)		-5
PRINT INT (.999999)		0

Nové funkce zkusíme použít v programu na simulování hodu hracími kostkami. Protože funkce INT 'ubere' téměř celý jednotkový interval, je nutné k násobku (B-A) funkce RND(1) tuto jednotku připočítat. Proto je v dalším příkladu: $((6-1)+1)*\text{RND}(1)=6*\text{RND}(1)$

```
10 GCLEAR: REM Simulovani hodu kostkami
14 FOR I=1 TO 5
18 X=INT (6*RND(1) +1)
22 PRINT X;:NEXT I: END
```

Podobným způsobem bychom mohli vygenerovat i čísla ve Sportce, Matesu aj. Mohlo by se však stát, že by některá čísla byla stejná. Proto musíme programově testovat, zda nové náhodné celé číslo není shodné s dříve vygenerovanými. Znamená to, že si musíme čísla označit pomocí nějakého indexu. Skupina indexovaných proměnných se společným jménem vytváří tzv. POLE, např. pole proměnné A(J) bude mít pro Sportku 6 prvků, označených A(0)-A(5). Pole s jedním indexem v závorce nazýváme jednorozměrná. BASIC C 2717 umožňuje i práci s poli dvojrozměrnými D(I,J) a trojrozměrnými T(I,J,K). Pokud bude v polích nejvíce 11 prvků (0-10), rezervuje si pro ně místo BASIC sám. Je-li prvků více, je nutné rezervovat si pro ně místo v paměti pomocí příkazu DIM A(19), DIM D(4,24) nebo DIM T(14,9,5).

Program pro generování čísel Sportky může vypadat následovně:

```
10 GCLEAR:PRINT "Generovani cisel Sportky"
20 FOR I=1 TO 6: REM Celkem bude 6 cisel
30 A=INT (49*RND(1)+1): REM Nahodne cislo mezi 1 a 49
40 FOR J=1 TO I: REM Prohlizi pole platnych cisel
50 IF A(J)=A THEN 30: REM Jsou-li cisla shodna, znovu
60 NEXT J: REM Dalsi cislo z pole pro srovnani
70 A(I)=A:PRINT A(I):NEXT I: REM Vytvoreni vystupniho pole
80 STOP:GOTO 20
```

Vynecháním příkazu STOP můžeme vygenerovat několik řádků 'tahů' Sportky. Zacyklený program lze zastavit pouze klávesou STOP.

Cvičení:

1. Napište program na vyzkoušení, kolika hodů hrací kostou je nutné pro získání součtu hodů nejméně 100.
2. Napište program pro generování 10 náhodných čísel z intervalu (-5,-10).
3. Srovnajte časy trvání 10000 průchodů prázdným programovým cyklem, naprogramovaným pomocí FOR_TO_, a jednoduchou smyčkou, vytvořenou pomocí N=10000:N=N-1:IF N>0 THEN...

TEMA: Práce s texty

Lekce: 8

NOVE POJMY: Textová (řetězcová) konstanta a proměnná, spojování
----- textů, porovnávání textů, textová pole.

NOVE PŘIKAZY: znak * (dolar) na konci jména textové proměnné
----- (nebo také znak 'sluničko'=znak měnové jednotky)

Podle názvu by se zdálo, že počítače jsou stroje jen na počítání. Ve skutečnosti jsou jejich schopnosti mnohem větší, proto je přesněji řadíme ke strojům na zpracování informací. Mnoho informací lze vyjádřit čísly, ale mnohem více je těch, které nutno vyjádřit slovy, tj. psaným textem. Jazyk BASIC umožňuje pracovat s texty podobně jako s čísly.

Při práci s čísly jsme rozeznávali konstanty a proměnné. Podobné je to při práci s texty. Textovou konstantou budeme nazývat takový text, který se v programu nemění, a je uveden v uvozovkách, aby byl odlišen od názvu číselné proměnné. Takové textové konstanty jsme používali již v předcházejících lekcích pro komentování výsledků práce programů na stínítku obrazovky, kdy nám samotná čísla mnoho neříkala (ztratila význam informace).

Proměnná, která může obsahovat různé texty, musí být odlišena od číselné proměnné, neboť bude v počítači také odlišně obsluhována programem. Odlišení je zabezpečeno znakem * (dolar) na konci názvu textové proměnné: AX, BX, CX, PX(6), DX(I,J), TX(I,J,K). Protože některé tiskárny nahrazují znak dolar tzv. sluničkem, (kolečko se 4 paprsky), je nutno mít tuto skutečnost na paměti při analýze výpisů programů. Tento kurs byl vypisován na tiskárně CONSUL 2111.11 a má proto také místo dolaru 'sluničko'.

Přifažení textu k textové proměnné lze provést pomocí příkazu LET a nebo bez tohoto přiřazovacího příkazu. Napište si následující program:

```
10 GCLEAR:PRINT "Přirazení textu do textové proměnné"
20 AX="hlavni ":BX="mesto ":CX="CSSR ":DX="je ":
30 EX="PRAHA ":PRINT AX;BX;CX;DX;EX
40 PRINT AT3,3;EX+DX+AX+BX+CX
50 PRINT AT7,7;DX+EX+AX+BX+CX+"?"
60 END
```

Po spuštění programu vidíme, že lze pomocí kombinací textových proměnných sestavovat různé věty. Spojovat textové proměnné lze buď oddělovačem typu ';' nebo operátorem '+', který má význam spojení a nikoli součtu. Proto také nedává mezeru mezi proměnné, na ni musí pamatovat programátor. V řádce 50 jsme použili spojení textových proměnných a textové konstanty. K vypisování textových proměnných lze používat jak PRINT tak i DISP, vstup textů do textové proměnné v programu je možný pomocí příkazu INPUT AX.

Texty můžeme navzájem porovnávat pomocí relačních operátorů:

- = textY vlevo i vpravo jsou shodné;
- <> texty se liší alespoň v jednom znaku;
- < některé z písmen textu vlevo je v abecedě dříve;
- > některé z písmen textu vlevo je v abecedě později;
- <= pokud mohou být oba texty shodné (došlo by k zacyklení);
- >= ze stejného důvodu.

Tato odlišnost relací je dána tím, že se porovnávají číselné kódy znaků z tabulky ASCII (viz přílohu). Proto je "A" < "a", ale také "12" < "3" -protože se porovnávají znaky jen do délky kratšího textu. Zkuste si v následujícím programu zadat nejprve různá jména, a potom např.: ANNA, ANCA, ANDULA, ANCA, ANNA. Pokud vynecháte v řádce 50 '=', tak se program na stejných jménech zacykli

```
10 GCLEAR:PRINT "Třídění textu podle abecedy"
20 N=5:FOR I=1 TO N:DISP"Zadej ";I;".jmeno"
30 INPUT AX(I):PRINT I,AX(I):NEXT I
40 N=N-1:P=0:FOR I=1 TO N
50 IF AX(I)<=AX(I+1) THEN 70
60 P=P+1:CX=AX(I):AX(I)=AX(I+1):AX(I+1)=CX
70 NEXT I: IF P>0 THEN 40
80 PRINT,"Setříděno podle abecedy:"
90 FOR I=1 TO 5:PRINT I;AX(I):NEXT I:END
```

Vlastní třídění (seřazování) je v řádce 60, kde je využito pomocné proměnné CX k uchování přepisované proměnné AX(I). Pomocná číselná proměnná P je dočasnou pamětí toho, že třídění neskončilo. V řádce 40 je proměnné N přiřazena hodnota o 1 menší proto, že porovnáváme dvojice textů. Současně se můžete přesvědčit o tom, že '=' v přiřazovacím příkazu nemá význam totožnosti. Zkuste si v programu místo jmen zadat čísla s různým počtem číslic.

Jako další program si uvedeme vytvoření jednoduchého telefonního seznamu (max. 11 položek):

```
5 GCLEAR:PRINT"Telefonní seznam"
10 DISP"Zadej počet položek":INPUT K
15 FOR I=1 TO K: DISP"Zadej ";I;". jmeno":INPUT A$(I)
20 PRINT A$(I);:DISP"Zadej jeho telefon:":INPUT B(I)
25 PRINT B(I):NEXT I: PRINT"Konec seznamu"
30 DISP"Napis jmeno, jehož telefon požaduje":INPUT C$
35 PRINT C$;:FOR J=1 TO K
40 IF C$=A$(J) THEN PRINT "ma telefon: ";B(J)
45 NEXT J:DISP"Chcete pokračovat (A=ano,N=ne)?:INPUT D$
50 IF D$="A" THEN 30
55 PRINT"Konec programu":END
```

Programový cykl 15-25 vytváří telefonní seznam, cykl 35-45 tento seznam prohlíží a porovnává se zadanou textovou proměnnou C\$. Rozhodování na řádku 50 porovnává jednu ze zadaných textových konstant (A/N) s textovou proměnnou D\$.

Pozor: Telefonní čísla mohou být nejvíce 6 místná, jinak by se v telefonním seznamu objevila v semilogaritmickém a navíc i zaokrouhleném tvaru. Vícemístná čísla je nutno uchovávat v textových proměnných, tj. místo B(I) použít v programu např. B\$(I).

Poznámka: Textové proměnné jsou často nazývány též jako 'řetěz-cové proměnné', protože je tvoří řetězec libovolných znaků (z anglického 'string'). Obdobně i konstanty. Proměnné, vstupující pomocí INPUT nesmí mít mezi slovy čárku, jinak se vypíše + + + Field lost at line + + +.

Cvičení:

1. Upravte program telefonní seznam na seznam s roky narození a vypište tabulku od nejstaršího po nejmladšího.
2. Napište program pro hádání jména krajského města, které si ze zadaného seznamu (a vypsaného na stínítku) vybere program pomocí funkce RND. Programem registrujte počet pokusů/hádání.

TEMA: Práce s texty
=====

Lekce: 9
=====

NOVE POJMY: Délka textové proměnné, instrukce pro práci
----- s texty, převod čísla na text a naopak.

NOVE PRIKAZY: LEN (X_x) délka textové proměnné X_x;
----- LEFT_x (X_x,I) omezení textu na I znaků zleva;
RIGHT_x (X_x,I) omezení textu na I znaků zprava;
MID_x (X_x,I,J) výběr části textu od I-tého znaku;
STR_x (C) převod číselné proměnné na textovou;
VAL (C_x) převod textové proměnné C_x s čísly
 nebo výrazy na číslo.

V textových proměnných mohou být libovolné znaky mimo uvozovek, které je odlišují od názvů číselných proměnných. Jak je vidět z přehledu nových příkazů této lekce, mohou být textové proměnné použity jako argumenty jiných textových funkcí.

Funkce LEN (X_x) určí počet znaků textové proměnné X_x. Je zřejmé že hodnota funkce LEN bude celé číslo. Vyzkoušejte:

```
10 GCLEAR:Xx="12345678":PRINT"Funkce pracující s texty"  
20 PRINT"Xx='';Xx,'" je text o délce: ";LEN(Xx);" znaku"
```

Funkce LEFT_x(X_x,I) vytvoří z původní textové proměnné novou, která bude tvořena počtem I znaků zleva. Počet znaků I je tzv. parametrem v argumentu funkce:

```
30 PRINT:PRINT "LEFTx (Xx,3)= ";LEFTx(Xx,3)
```

Podobně funkce RIGHT_x (X_x,I) vytvoří z původního textu v X_x nový, obsahující pouze I znaků zprava:

```
40 PRINT "RIGHTx (Xx,4)= ";RIGHTx(Xx,4)
```

Další funkce MID_x (X_x,I,J) je univerzální, protože umožňuje vybrat z textové proměnné libovolnou část od I-tého znaku v délce J znaků. Je pochopitelné, že oba parametry I a J musí být menší než délka textové proměnné LEN (X_x). Zapomeneme-li parametr J, bude výsledkem funkce MID_x(X_x,I) text od I-tého znaku do konce.

```
50 PRINT "MIDx (Xx,5,2)= ";MIDx(Xx,5,2): PRINT
```

Jako parametry v uvedených funkcích mohou být uvedeny i proměnné počítadel programových cyklů:

```
60 PRINT "K"; TAB(8); "LEFT*(X*,K)"; TAB(22); "RIGHT*(X*,K)"
70 FOR K=1 TO LEN(X*)
80 PRINT K, LEFT*(X*,K); TAB(30-K); RIGHT*(X*,K)
90 NEXT K: PRINT: PRINT "K"; "MID*(X*,9-K,(K+K)/2)"
100 FOR K=1 TO LEN(X*): PRINT K, MID*(X*,9-K,(K+K)/2)
110 NEXT K: END
```

Pokud ste pečlivě dopsali programové řádky až do konce, můžete program spustit a přesvědčit se o tom, jak jednotlivé funkce pracují se zadaným textem.

Poznámka: Kdybychom zadali jako parametr číslo 0 (LEFT*(X*,0)), ohlásil by po čítač chybu: + + +Fnc. param. at ln + + +
Zkuste, co se stane, bude-li parametr > LEN (X*).

Pomocí funkce MID*(X*,I,J) můžeme například zjistit, kolikrát se určité písmeno vyskytuje v nějakém textu (větě a pod.):

```
10 GCLEAR: PRINT "Analyza vyskytu pismen v textu"
20 DISP "Zadej vetu textu": INPUT A*: PRINT AT(0,3); A*
30 DISP "Zadej pismeno": INPUT B*: C=0: REM Pocitadlo C
40 FOR I=1 TO LEN(A*)
50 IF MID*(A*,I,1)=B* THEN C=C+1
60 NEXT I: PRINT "Pismeno "; B*; " se vyskytuje "; C; " krat."
70 DISP "Jine pismeno (A/N) ?": INPUT D*: IF D*="A" THEN 30
80 DISP "Jiny text (A/N) ?": INPUT D*: IF D*="A" THEN 20
90 END
```

Funkce STR*(C) přemění číslo na text (řetězec číslic), s nímž může pracovat pomocí všech funkcí pro práci s texty. Asi nemá smysl takto pracovat s čísly v semilogaritmické tvaru. Funkce STR* umožňuje spojování čísel na čísla >999999 tam, kde nelze používat semilogaritmická čísla (např. výkazy mezd za podnik, plnění plánu ap.). Příklad takové aplikace si uvedeme později.

```
10 GCLEAR:PRINT"Vypocet casti rodneho cisla"
15 DISP"Zadej rok sveho narozeni":INPUT R
20 IF R<1900 OR R>1988 THEN PRINT"To snad ne?":GOTO 15
25 R%=RIGHT$(STR$(R),2)
30 DISP"Zadej mesic narozeni":INPUT M
35 IF M=0 OR M>12 THEN 30
40 DISP"Zadej den narozeni":INPUT D
45 IF D=0 OR D>31 THEN 40
50 DISP"Jsi zena (A/N)":INPUT D$:IF D$="A"THEN M=M+50
55 M%=STR$(M):IF M<10 THEN M%="0"+RIGHT$(M%,1)
60 D%=STR$(D):IF D<10 THEN D%="0"+RIGHT$(D%,1)
65 PRINT"Tvoje rodne cislo zacina: ";
70 PRINT R%+RIGHT$(M%,2)+RIGHT$(D%,2)+"/....":END
```

Pro kontrolu správnosti zadaných dat narození byl použit logický operátor OR (nebo), jehož význam je ze zápisu programu zřejmý

Funkce VAL(C%) přemění text nebo výraz uložený v C% na číslo a umožňuje s ním pracovat pomocí všech aritmetických operátorů i funkcí.

```
10 GCLEAR: PRINT"Soucet cislic velkeho cisla"
15 DISP"Zadej vicemistne cislo": INPUT A$:N=LEN(A%)
20 B=0:FOR I=1 TO N:B=B+VAL(MID$(A%,I,1)):NEXT I
25 PRINT"Soucet vseh cislic zadaneho cisla ";A%;" je: ";B
30 STOP
50 DISP"Zadej funkci promenne X:":INPUT F%
60 PRINT"Zadana funkce: ";F%
70 DISP"Zadej hodnotu X:":INPUT X:PRINT"Zadano X=";X
80 PRINT"Hodnota funkce ";F%;" pro X=";X;" je: ";VAL(F%)
90 END
```

Tento program vznikl spojením dvou programů. Na řádce 50 můžeme zadat nějakou funkci, obsahující proměnnou X (např. X^2 , $SQR(X)$, X^2 , $SIN(X)$ aj.). Pro zadanou hodnotu na řádce 70 program vypočte hodnotu funkce na řádce 80.

Cvičení:

1. Sestavte program, kterým se zadaný text vytiskne pozpátku, např. ze slova 'PEC' udělá 'CEP'
2. Upravte program výpočtu rodného čísla tak, aby je sestavil z normálně zapsaného data narození: dd.mm.rrrr

TEMA: Data v programu, práce s magnetofonem

Lekce: 10

NOVE POJMY: Číselná a textová data v programu, čtení dat,
----- ukazovátko dat, záznam programu na magnetofon,
verifikace záznamu, čtení programu z magnetofonu

NOVE PŘIKAZY: DATA seznam konstant
----- READ seznam proměnných
RESTORE n kde n=číslo řádku DATA
SAVE nn,název kde nn=číslo nahrávky <99
CHECK nn název=titul programu
LOAD nn max. 8 znaků
NEW

Přikazy DATA a READ tvoří dvojici. Zatímco příkaz DATA slouží pro umístění konstant různého typu, tvořících součást programu, příkaz READ slouží ke čtení těchto konstant a jejich přiřazení jednotlivým proměnným v seznamu. V seznamu konstant za DATA jsou jednotlivé konstanty různého typu odděleny čárkami ',', podobně jsou odděleny proměnné za příkazem READ. Konstanty jsou do proměnných přiřazovány tak, jak jdou za sebou na jednom nebo více řádcích v programu, podmínkou je, aby si typ konstanty a proměnné navzájem odpovídaly (číslo lze přiřadit do textové proměnné, ale naopak ne) - jinak nahlásí BASIC chybu: + + + Syntax err ... + + + Pokud je konstant v DATA méně než zadaných proměnných v READ, počítač ohlásí chybu: + + + Data exhaust ... + + + (chybějící data).

Funkci dvojice příkazů READ-DATA si ukážeme na následujícím programu:

```
11 GCLEAR: PRINT "Čtení dat příkazem READ"
22 DATA LEDEN,31,UNOR,28,BREZEN,31,DUBEN,30
33 DATA KVETEN,31,CERVEN,30,CERVENEC,31,SRPEN,31
44 DATA ZARI,30,RIJEN,31,LISTOPAD,30,PROSINEC,31
55 DIM M$(12),D(12):REM Protože je konstant >11
66 FOR I=1 TO 12:READ M$(I),D(I):PRINT I;M$(I);D(I):NEXT I
77 DISP "Vlož číslo měsíce,":INPUT X:IF X=0 OR X>12 THEN 77
88 PRINT X;". měsíc je ";M$(X);" a má";D(X);" dnů."
99 DISP "Jste jednou (A/N)?":INPUT D$:IF D$="A" THEN 77
100 END
```

Pokud má textová proměnná obsahovat čárku, dvojtečku nebo mezeru zleva či zprava, musí být uvedena v uvozovkách, jinak lze uvozovky vynechat, jak bylo uvedeno v programu. Vynechání konstanty v řádku DATA má za následek přiřazení nulové hodnoty do číselné proměnné nebo 'vynezerování' textové proměnné.

Kdybychom nyní spustili program příkazem GOTO 66, nahlásil by chybu. Víte jakou? Pokud ne, tak si to zkuste a zdůvodněte. Proč nenahlásí chybu, když jej spustíme příkazem RUN 66 ?

Někdy potřebujeme data uvedená v programu používat víckrát. Pro tento případ je v souboru povelů jazyka BASIC příkaz RESTORE nn (obnovit), který slouží jako ukazovátko na řádek se správnými daty pro navazující příkaz READ. Pokud není za RESTORE uvedeno číslo řádku, nastaví se ukazovátko na první řádek DATA v programu. Ukážeme si to nejprve na našem programu, který spustíme takto:

```
RESTORE 33:GOTO 66
```

Program 'vynechal' DATA na řádku 22, proto je chybně přiřadil, a protože mu další DATA chyběla, nahlásil chybu. Pokud program odstartujeme RESTORE:GOTO 66 bude všechno v pořádku.

Náš poslední program si nyní zkusíme nahrát na kazetu v magnetofonu. Předpokládejme, že je počítač s magnetofonem propojen kabelem (v počítači je to konektor 'o_o' na zadní stěně a v magnetofonu konektor IN/OUT nebo RADIO). Nejprve si připravíme do dialogového řádku příkaz

```
SAVE 1,DataRead
```

na magnetofonu stlačíme současně ZAZNAM+VPRED nebo RECORD+START a potom odešleme připravený příkaz klávesou EOL. Pokud má magnetofon odposlech, je slyšet nejprve pískání (asi 2 sec) a potom chraplavý zvuk nahrávaného programu. Nahrávka končí opět pískáním a výpisem OK do dialogového řádku. Magnetofon nezastavujeme, ale stiskneme klávesu 'RCL', která nám do dialogového řádku vrátí příkaz záznamu. Ten potvrdíme 'EOL' a nahrajeme program ještě jednou. Po ukončení nahrávky vrátíme kazetu na 'začátek' (tam, kde jsme začínali nahrávat). Připravíme si příkaz pro ověření nahrávky

```
CHECK 1
```


a spustíme magnetofon pouze pro čtení (VPRED nebo START). Pokud se ozve pískání z magnetofonu, mělo by se ozvat současně i z reproduktoru počítače (pokud tomu tak není, postačuje v mnoha případech zastavit a znovu spustit magnetofon). Současně s chraplavým zvukem z magnetofonu by se měla vypsát tzv. hlavička programu:

```
01/> DataRead
```

a po bezchybném ověření nahrávky by se mělo vypsát OK. Pokud byla zjištěna chyba, tak počítač vypíše zprávu

```
+ + + File error + + +
```

Zkusíme pomocí 'RCL' znovu vyvolat 'CHECK 1', odešleme jej 'EOL' a podobně zkontrolujeme i druhou nahrávku. Pokud je nahrávka OK, vrátíme kazetu v magnetofonu před naši první nahrávku, vymažeme program z paměti počítače pomocí příkazu

```
NEW          (pozor, je to nebezpečný příkaz).
```

Přesvědčíme se o vymazání programu například příkazem LIST.

Nyní si zkusíme náš program nahrát pomocí příkazu LOAD (naplnit) s parametrem čísla programu:

```
LOAD 1
```

Spustíme magnetofon pro čtení (jako při CHECK) a čekáme na pískání, potom na 01/> DATAREAD a závěrečné OK, po němž magnetofon zastavíme (nebo RCL, EOL atd).

Zpočátku se Vám to bude zdát trochu komplikované, ale je to pouze otázka cviku. Používáte-li nové a dobré kazety, postačuje obvykle dvojitá nahrávka programu a není nutné program kontrolovat pomocí CHECK, který ověřuje pouze správnost kontrolního součtu nahrávky a neověřuje totožnost dat v paměti počítače.

Cvičení:

1. Nahrejte si nejzajímavější programy z kursu.

Příklady řešení úloh

=====

LEKCE/Cvičení

```
2/2 PRINT 10*(5+6)                výsledek: 110
    PRINT 15*(5+7*(10-8))          285
    PRINT 3+4*(3/(4-15))           1.90909
    DEG:PRINT SIN(45)*COS(70)      .241845
    RAD:PRINT SIN(45)*COS(70)      .538896
    PRINT 7/((5*6)-7*(8-7))        .304348
    PRINT LOG(128*(15+78))/(15/34+2)/4*.2 .192215

3/1 10 GCLEAR:A=4:B=8:C=12
    20 PRINT"Strany trojuhelnika"
    30 PRINT"A=","B=","C="
    40 PRINT A,B,C
    50 PRINT"Obvod trojuhelnika: A+B+C=";A+B+C
    60 END

3/2 10 GCLEAR:PRINT"Vypocet prepony pravouhlehého trojuhelnika"
    20 A=6:B=8:PRINT"A=","B=","C=SQR(A*A+B*B)"
    30 PRINTA,B,SQR(A*A+B*B)
    40 END

4/1 10 GCLEAR:DISP"Zadej polomer koule R:":INPUT R
    20 PRINT"Zadano R= ",R
    30 PRINT"Obvod koule: ",2*3.14*R
    40 PRINT"Povrch koule: ",4*3.14*R^2
    50 PRINT"Objem koule: ",4/3*3.14*R^3
    60 END

4/2 10 GCLEAR:DISP"Zadej 4 cisla, oddelana carkou"
    20 INPUTA,B,C,D:PRINT"Zadano: ";A;B;C;D
    30 PRINT"Prumer z tecto cisel je: ";(A+B+C+D)/4
    40 END

4/3 10 GCLEAR:PRINT"Vypočet objemu a povrchu kvadru"
    20 DISP"Zadej stranu A:":INPUT A:PRINT "A=";A
    30 DISP"Zadej B:":INPUT B:PRINT "B=";B
    40 DISP"Zadej C:":INPUT C:PRINT "C=";C
    50 PRINT"Objem kvadru=";A*B*C
    60 PRINT"Povrch kvadru=";2*(A*B+A*C+B*C)
    70 END
```

4/4 Počítač pípne a čeká na vstup další proměnné. Po jejím zadání pokračuje v práci na programu.

```
5/1 10 GCLEAR:S=0:C=0
    20 PRINT"Kolik celych cisel nutno secest do 100?"
    30 C=C+1:S=S+C:PRINT,C;S
    40 IF S<100 THEN 30
    50 PRINT"Soucet cisel od 1 do ";C;" je:";S
    60 END
```

```
5/2 10 GCLEAR:PRINT"Prumer znamek z 10 predmetu":I=1:S=0
    20 DISP"Zadej znamku z";I;" predmetu":INPUT A
    30 IF A<5 THEN 50
    40 PRINT"Tento zak neprospel":END
    50 S=S+A:I=I+1:IF I<11 THEN 20
    60 P=S/10:PRINT"Zak ma prumer:";P;
    70 PRINT", prospel ";
    80 IF P<=1.5 THEN PRINT"s vyznamenanim":END
    90 IF P<=2 THEN PRINT"velmi dobre":END
```

```
6/1 10 GCLEAR:PRINT"Vypocet faktorialu cisla N"
    20 S=1:I=0:DISP"Zadej cislo N:":INPUT N
    30 I=I+1:S=S*I:IF I<N THEN 30
    40 PRINT"Faktorial cisla ";N;" je:";S
    50 END
```

```
6/2 10 GCLEAR:PRINT"Nejvetsi zadane cislo"
    20 M=-999999:DISP"Zadej pocet cisel":INPUT N
    30 FOR I=1 TO N:DISP"Zadej";I;" cislo":INPUT B
    40 IF B>=M THEN M=B
    50 PRINT B:NEXT I
    60 PRINT"Nejvetsi cislo je:";M
    70 END
```

```
6/3 10 GCLEAR:PRINT"Nejvetsi a nejmensi cislo"
    20 M=-999999:L=999999:DISP"Zadej..."
    30 ...
    35 IF B<=L THEN L=B
    40 IF B>=M THEN M=B
    50 ...
    60 ...
    65 PRINT"Nejmensi cislo je:";L
    70 END
```

```
6/4 10 GCLEAR:PRINT"Mala nasobilka:"
    20 FOR R=1 TO 10
    30 FOR S=1 TO 10:PRINT AT2*R,4*S;S*R:NEXT S
    40 NEXT R:END

7/1 10 GCLEAR:PRINT"Pocet hodu kostkou na soucet >=100"
    20 S=0:C=0
    30 X=INT(6*RND(1)+1):S=S+X:C=C+1
    40 IF S<=100 THEN 30
    50 PRINT"Tentokrat jsme hazeli";C;" krat"
    60 GOTO 20

7/2 10 GCLEAR:FOR I=1 TO 10
    20 PRINT 5*RND(1)-10
    30 NEXT I:END

7/3 10 GCLEAR:PRINT"Pocitam. Stopuj cas do 'STOP'"
    20 FOR I=1 TO 10000:NEXT I:REM Asi 20 sec
    30 STOP
    40 N=10000:PRINT"Pocitam. Stopuj cas do 'OK'"
    50 N=N-1:IF N>0 THEN 50:REM Asi 90 sec
    60 END

8/1 5 GCLEAR:PRINT"Seznam narozeni, trideni podle veku"
    10 DISP"Zadej pocet polozek:":INPUT K:N=K
    15 FOR I=1 TO K:DISP"Zadej";I;". jmeno":INPUT A$(I)
    20 PRINT A$(I);DISP"Zadej jeho rok narozeni":INPUT B(I)
    25 PRINT B(I):NEXT I:PRINT"Konec seznamu"
    30 N=N-1:P=0:FOR I =1 TO N
    35 IF B(I)<=B(I+1) THEN 50
    40 P=P+1:C=B(I):B(I)=B(I+1):B(I+1)=C:REM Presun roku
    45 C$=A$(I):A$(I)=A$(I+1):A$(I+1)=C$:rem Presun jmena
    50 NEXT I:IF P>0 THEN 30
    55 PRINT"Setrideno podle veku:"
    60 FOR I=1 TO K:PRINT,B(I);A$(I):NEXT I
    65 END

8/2 10 GCLEAR:PRINT"Hadani krajskeho mesta"
    15 DATA10,PR,CB,PL,UL,HK,BR,OV,BRA,BB,KO
    20 READ N:FOR I=1 TO N:READ A$(I):PRINT A$(I):NEXT I
    25 M=INT(N*RND(1)+1):C=1
    30 PRINTAT15,0"Vybral jsem si zkratku, hadej kterou?"
    35 INPUT H$:IF H$<>A$(M) THEN C=C+1:PRINT H$;" ne!":GOTO35
    40 PRINT"Uhodl jsi na";C;". pokus"
    45 DISP"Znovu (A/N)?":INPUT D$:IF D$="A" THEN RUN
    50 END
```

```
9/1 10 GCLEAR:PRINT"Obraceni textu"
20 DISP"Zadej text bez carek":INPUT T$:N=LEN(T$):DIM A$(N)
30 FOR I=1 TO N:A$(I)=MID$(T$,I,1):NEXT I
40 FOR I=N TO 1 STEP-1:PRINT A$(I);:NEXT I
50 END

      nebo jinak:
30 FOR I=0 TO N-1:A$=A$+MID$(T$,N-I,1):NEXT I
40 PRINT A$:END

      nebo takto:
30 FOR I=N TO 1 STEP-1:A$=A$+MID$(T$,I,1):NEXT I
40 PRINT A$:END

9/2 10 GCLEAR:PRINT"Rodne cislo z data narozeni"
15 DISP"Zadejte datum narozeni (den mesic rok)"
20 INPUT Z$:N=LEN(Z$):PRINT,"Zadano:";Z$
25 FOR I=1 TO N:A$=MID$(Z$,I,1)
30 IF A$<"0" OR A$>"9" THEN D=I:I=N
35 NEXT I:D$=LEFT$(Z$,D-1):Z$=RIGHT$(Z$,N-D)
40 FOR I=1 TO N-D:A$=MID$(Z$,I,1)
45 IF A$<"0" OR A$>"9" THEN M=I:I=N-D
50 NEXT I:M$=LEFT$(Z$,M-1):R$=RIGHT$(Z$,2)
55 DISP"Jste zena (A/N)?":INPUT X$
60 IF X$="A" THEN M=VAL(M$)+50:M$=RIGHT$(STR$(M),2)
65 IF LEN(D$)<2 THEN D$="0"+D$
70 IF LEN(M$)<2 THEN M$="0"+M$
75 PRINT R$+M$+D$+"/....":END
```

Základní prvky jazyka BASIC-G:

=====

Konstanty: -číselné: -reálná čísla z intervalu $(-10^{38}, +10^{38})$
místo desetinné čárky je desetinná tečka;
-šestnáctková celá čísla ('0000 až 'FFFF);
-textové (řetězcové) jsou tvořeny znaky z ASCII kódu
v uvozovkách;

-zobrazení na 6 platných míst (zaokrouhleně):
-v přímém tvaru: (.01, 999999)
psaní znaménka + je nepovinné;
-v exponenciálním tvaru: $(-1E-38, +1E+38)$
obecný tvar: $(-x.xxxxxE-yy; x.xxxxxE+yy)$
-nevýznamné nuly jsou vynechávány;

Proměnné: -číselné, označené velkým písmenem/písmenem a znakem;
A,BX,C2,D9 mimo zkratk: AT,IF,ON,OR,TO;
-textové, označené podobně, odlišené znakem x (dolar);
Fx,G3x,HHx,I6x
-indexované: označené indexem v kulaté závorce:
.jedorozměrné: K(4),LM(J),P6x(0),Qx(I)
.dvojdroměrné: R(1,2),S2x(I,J)
.trojdroměrné: TT(X,Y,Z),Ux(0,4,8)

Pole: -číselných nebo textových proměnných (podle obsahu);
-jedorozměrná, dvojdroměrná nebo trojdroměrná;
-pokud je rozměr pole <11, rezervace je automatická,
jinak musí být předem rezervováno pomocí příkazu DIM;
-prvky pole se počítají od 0: DIM A(5)-pole 6 prvků;

Výrazy: -aritmetické: $A+B, A-B, A*B$ (násobení), $A/B, A^B$ (mocnina);
-logické: NOT A (negace), A AND B (součin), A OR B (součet);
-relační: $A>B, A<B, A=B, A\leq B, A\geq B, A\neq B$ (nerovnost);
-textové: $Ax=Bx+Cx+Dx$ -spojení textových proměnných;

Funkce: -standardní: SIN, COS, TAN, ATN, LOG, EXP, SQR, ABS, INT, SGN;
BIT, FRE, INP, INKEY, PEEK, RND, SPC, TAB, USR
ADR, APEEK, AT, INK
-uživatelské: DEF FNC A(X)
-textové: ASC, LEN, VAL - výsledkem je číslo;
(řetězcové) CHRx, HEXx, LEFTx, RIGHTx, MIDx, STRx;

Příkazy: -editační: AUTO, LIST, LLIST, NEW
-řídící: CLEAR, CONT, CHECK, DLOAD, DSAVE, LOAD, RUN, SAVE
-standardní: BEEP, DATA/READ, DEG, DIM, DISP, END, FOR/NEXT,
GCLEAR, GOSUB/RETURN, GOTO, IF, LET, INPUT, NULL
ON, PAUSE, PEN, PRINT, RAD, REM, RESTORE, STOP, ?
-grafické: AXES, BMOVE, BPLOT, FILL, LABEL, MOVE, PLOT, SCALE
-systémové: CODE, OUT, POKE, ROM
-vstup/výstupní: OUTPUT, ENTER, CONTROL, STATUS, WAIT

Priorita (pořadí provádění operátorů a funkcí ve výrazech):

1. kulaté závorky (...) a funkce
2. umocňování: A^B , 10^5
3. aritmetická negace (unární minus): $-A$, $-.234$
4. násobení a dělení: $A*B$, $56*.25$, A/B , $2/5$
5. sečítání a odečítání: $A+B$, $1+1$, $A-B$, $.34-.12$
6. relační operátory: $A<B$, $A>B$, $A=B$, $A\leq B$, $A\geq B$, $A\neq B$
7. binární logická negace: NOT A
8. binární logický součin: $A \text{ AND } B$
9. binární logický součet: $A \text{ OR } B$

Příkazový řádek má délku 80 znaků, zobrazených 48 lze posouvat;
-může mít více příkazů, musí být odděleny dvojtečkou;
-nezačíná-li číslem, příkazy se provedou po stisku EOL
-začíná-li číslem, po EOL je uložen do paměti programu

Obrazovka monitoru počítače umožňuje tato zobrazení:

-znakový režim: 25 řádků po 48 znacích,
znak v rastru 6*8 bodů;
-grafický režim: 243*288 bodů

-jednotlivé zobrazované body jsou uloženy ve videopa-
měti od adresy 'C000 (šestnáctkové), a mohou být zo-
brazovány plným nebo polovičním jasnem a mohou blikat.

