Jak začít s jazykem Tiny BASIC

Pro úplné začátečníky bývají první kroky někdy velmi těžké. Abyste nemuseli tápat jak začít, najdete zde pár ukázek a rad při prvních krocích s mikropočítačem TinyBasRV.
(Poznámka: V textu se vyskytují uvozovky dolní i horní. Programovací jazyk Tiny BASIC má ale pouze jeden typ – a to horní uvozovky. Při zadávání příkazů u TinyBasRV lze používat pouze horní uvozovky.)

Příkaz nebo řádek programu

Počítače orientované na různé verze jazyka BASIC mají obvykle po startu spuštěný interpreter jazyka, který plní současně úkoly „operačního systému“, příkazového řádku a překladače jazyka. Podobně je to i u TinyBasRV. Po spuštění počítač čeká na zadání příkazu (nebo více příkazů), které má vykonat.
Uživatel se může rozhodnout, zda počítači zadá přímo povel „proveď ihned určitou akci“, nebo mu bude formou psaní programu „dělat seznam“, co vše bude počítač provádět, až mu dá pokyn ke startu programu.
Oba režimy (přímý příkaz i psaní programu) pracují s příkazy, kterým počítač rozumí. Je jich celkem 28 a najdete je podrobně popsané v příručce TinyBasLang a na jejím konci je souhrnná tabulka s přehledem příkazů.
Příkazy jazyka Tiny BASIC se počítači TinyBasRV nezadávají postupným vypisováním pomocí jednotlivých znaků. Příkaz se počítači předá pomocí stisknutí levé klávesy Ctrl a pak současným stisknutím klávesy podle tabulky na konci v příručce TinyBasLang.
Typickou první ukázkou bývá v učebnicích programování úkol, který donutí počítač, aby pozdravil. U TinyBasRV toho můžeme dosáhnout 2 způsoby:
1) Zadat přímo příkaz pro vypsání pozdravu

    PRINT „Ahoj“

a pak stisknout Enter. Příkaz PRINT se zadá jako kombinace Ctrl+P a pak se mezi horní uvozovky napíše text. Počítač na dalším řádku vypíše požadovaný pozdrav. Nevýhodou je, že takto musíte postupovat kdykoliv, kdybyste chtěli, aby počítač pozdravil.
2) Druhou možností je napsat vypsání pozdravu jako program a ten si pak spustit. Můžete si představit, že počítač je sluha, který čeká se zápisníkem dlouhým 128 řádků. Řádky v zápisníku má počítač očíslované 0 až 127. Na každý řádek si může zapsat vámi zadaný povel. Až mu nadiktujete své povely, můžete mu nařídit „a teď to podle seznamu vykonej“. Některé řádky mohou být volné, jinde mohou být až 3 příkazy na řádku.
V našem případě můžeme počítači zadat např.:

    1 PRINT „Ahoj“

Po stisknutí Enter se zdánlivě nestane nic. Počítač si ale uložil náš příkaz do svého zápisníku na řádek 1. A bude čekat na další příkazy. Pokud nyní zadáme příkaz RUN (Ctrl+R) a stiskneme Enter, počítač se vrhne na plnění úkolů v zápisníku. Na řádku 1 najde příkaz, který provede. Výsledkem bude stejný výstup jako u bodu 1). Rozdíl je ale v tom, že nyní si můžeme pozdrav pomocí příkazu RUN vyvolat kdykoliv chceme.

Víc příkazů na řádku

Protože má mikropočítač řádek pro příkazy dlouhý 32 znaků, bylo by škoda toho nevyužít. Na řádek si tak může zapsat až 3 příkazy i s parametry oddělené pomocí znaku **;** . Omezeni jste pouze počtem 32 znaků. Můžete zkusit napsat na jeden řádek:

    INPUT A,B ; PRINT „Plocha: “ , A*B

A zkuste, co to udělá po spuštění pomocí Enter. Pokud jste se nespletli v žádném znaku při zadávání příkazů, máte jednoduchý nástroj na výpočet plochy obdélníka.

S výhodou toho můžete využít, pokud potřebujete někam na obrazovku displeje zapsat konkrétní znak. Můžete použít kombinaci příkazů

```
CURSOR 400 ; PUTCH 143
```

Uvedené hodnoty parametrů u příkazů vypíší uprostřed obrazovky plný čtvereček.

Jak počítač zpracovává řádky programu

Jestliže se rozhodnete tvořit program, který bude počítač později vykonávat, můžete mu začít „diktovat“ na řádky do zápisníku, co bude provádět. Vždy můžete zapisovat na libovolný řádek od 0 do 127. Když budete mít program dokončený, spustíte ho povelem RUN.
Počítač začne postupně procházet jednotlivé řádky od čísla 0 a plnit příkazy, které na nich najde. Prázdný řádek přeskočí a přesune se k dalšímu řádku o 1 vyššímu. Jakmile dojde na řádek 127 a měl by přejít na neexistující řádek 128, vykonávání programu ukončí.
Vyzkoušejte např. program:

```
1 INPUT A
5 PRINT „Objem: “ , A*A*A
```

A máte jednoduchý program pro výpočet objemu krychle. Zkuste si jej spustit.
Vykonávání programu můžete také ukončit na libovolném řádku pomocí příkazu END. Doplňte program o řádek:

```
3 END
```

Nyní po spuštění vás počítač vyzve k zadání hodnoty A, ale objem krychle už nespočítá. Je to kvůli řádku 3, na němž narazil na ukončovací příkaz END a na řádek 5 už nedojde.

Jako radu můžete vzít doporučení na speciální využití řádku 0. Je dobré se naučit zapsat na řádek 0 příkaz REM a za něj doplnit krátký popis, co dělá program. Po čase si již nemusíte pamatovat, co jste programem chtěli vytvořit a poznámka na začátku programu vám může osvěžit paměť.

Tvorba programu – nekonečná a počítaná smyčka

Počítače usnadňují práci díky tomu, že jim popíšete, co mají opakovat a nemusíte jim každý krok předepisovat zvlášť. Kdybychom chtěli vypsat sudá čísla, nebudeme předepisovat:

```
1 PRINT 2
2 PRINT 4
3 PRINT 6
…
```

Necháme počítač počítat po 2 (od toho je přece počítač) a hodnoty budeme vypisovat na řádky pod sebe. Např. takto:

```
1 LET A=0
2 LET A=A+2 ; PRINT A ; PUTCH 10
3 GOTO 2
```

Program po spuštění začne velkou rychlostí vypisovat sudá čísla. Na první řádku nastavíme hodnotu A na 0. Druhý řádek zvedne hodnotu A o 2, vypíše ji a pomocí příkazu PUTCH se přesune kurzor na další řádek. Protože příkaz GOTO na řádku 3 vrací program znovu na řádek 2, vznikla nekonečná smyčka, která nechá program běžet dokud počítač nevypnete nebo neresetujete. Program je možné také přerušit příkazem END zadaným na klávesnici.
Kdybyste chtěli vypsat pouze omezený počet sudých čísel (např. do 20) upravte řádek 3 na novou verzi:

```
3 IF A<20 THEN GOTO 2
```

Nyní když počítač spustíte, vypíše sudá čísla jenom do 20. Podmíněný příkaz IF vyhodnocuje splnění podmínky a vrací program na řádek 2 pouze pro čísla menší než 20. A pokud jste nevymazali řádek 5 z minulého příkladu, vypíše jako bonus navíc i objem krychle o straně 20. Řádek 5 můžete jednoduše vymazat zadáním prázdného řádku 5:

```
5
```

Příkazy, které se vícekrát za sebou opakují, se obvykle označují jako smyčka programu. Příkaz IF je možné zadat jak na začátek smyčky – např.:

```
5 IF podmínka THEN GOTO 21
        …
        příkazy smyčky
        …
20 GOTO 5
```

tak na konci smyčky, jak bylo uvedeno u příkladu s výpisem sudých čísel. Na první pohled se může zdát, že je jedno, kam se vyhodnocování podmínky umístí. V praxi to také často vede ke stejnému chování programu. Rozdíl je ale především v tom, že smyčka s vyhodnocovacím příkazem na konci se musí vždy alespoň jedenkrát provést. Je-li test na začátku a podmínka je při prvním vstupu do smyčky splněna, nemusí smyčka proběhnout ani jednou.
Pomocí příkazu IF/THEN a GOTO lze i v takto jednoduchém jazyku vytvořit programové konstrukce, na které mají vyšší jazyky speciální příkazy.

Potřebujete li provádět během programu opakovaně stejné operace s různými parametry, můžete využít tzv. podprogram. Jako ukázku podprogramu si můžeme předvést něco, co po svém zavolání zajistí několik pípnutí reproduktoru podle hodnoty v proměnné A.

```
40 BEEP 1000,200 ; DELAY 200
41 LET A=A-1; IF A>0 THEN GOTO 40
42 RETURN
```

V programu nyní můžu kdykoliv, když potřebuji signalizovat zvukově několik pípnutí, vložit řádek:

```
23 LET A=5 ; GOSUB 40
```

Na daném řádku si v proměnné A nastavím, kolikrát se má pípnout a pak volám podprogram na řádku 40. Ten provede patřičný počet pípnutí a pomocí návratového příkazu RETURN na řádku 42 se mi program vrátí na řádek o 1 číslo větší, než odkud byl volán. V našem případě by program pokračoval řádkem 24.

Kde je program uložený (příkazy FILE a DISK)

Mikropočítač si uchovává zadaný program ve své sériové EEPROM paměti. Protože jeden zápisník je pro správného sluhu málo, má i náš počítač možnost vybavit se více zápisníky. Je jasné, že program na péči o zahradu je jiný, než program na úklid domácnosti, nebo na údržbu auta.
Podle typu instalované paměti lze do ní uložit 1 až 16 programů. Typy pamětí a počet programů souhrnně ukazuje tabulka v návodu TinBasHW.
Mezi programy lze přepínat pomocí příkazu FILE. Např.

```
FILE 3
```

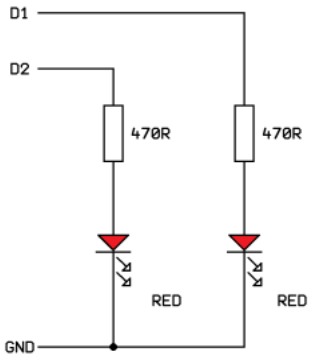
přepne na soubor s programem 3 (dá našemu sluhovi do ruky zápisník číslo 3). Tento program (zápisník v ruce) zůstane aktuální až do nového příkazu FILE. Po startu počítače je aktuální vždy program 0. Pokud si nejste jisti, který program je právě přepnutý, zadejte příkaz FILE bez parametru a počítač vám jeho číslo odpoví.
Aby to nebylo tak jednoduché, může mít náš sluha 2 knihovničky a v každé z nich více zápisníků. Na desku počítače je totiž možné osadit 2 sériové EEPROM paměti. Aby se mezi nimi dalo rozlišovat, je jedna označená „h“ (jako hard disk) a je pevně připájená na desku. Druhá má označení

„f“ (jako flash nebo floppy disk). Jestliže je použita patice, lze ji ve vypnutém stavu vyjmout a vložit místo ní novou.
Mezi EEPROM pamětmi se přepíná příkazem DISK – např.:

```
DISK f
```

Opět v případě nejistoty můžete zadat příkaz bez parametru a počítač odpoví, který disk je aktivní. Po startu je to vždy disk h.

Rozhraní s vnějším světem – výstražné světlo
Abychom si vyzkoušeli práci s výstupy a jednoduchým ovládáním vnějších periferií, můžete zkusit připojit na digitální výstupy D1 a D2 přes rezistory dvě červené LED podle následujícího schéma:

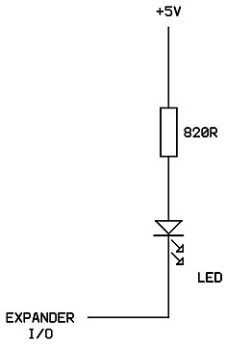


Vložíme do paměti jednoduchý program:

```
1 DOUT 1,1;  DOUT 0,2
2 BEEP  2000,100;  BEEP  800,200;  DELAY 200
3 DOUT 0,1;  DOUT 1,2
4 BEEP  2000,100;  BEEP  800,200;  DELAY 200
5 GOTO  1
```

Získáme tím jednoduché výstražné zařízení k železničnímu přejezdu i se zvukovou signalizací. Na řádcích 1 a 3 vždy zapneme jednu LED (a druhou naopak vypneme). Řádky 2 a 4 jsou stejné a dělají zvukovou signalizaci a krátkou prodlevu. Řádek 5 pak program zacyklí.

Expander
Máte-li dostupnou rozšiřující desku s expanderem PCF8575, můžete ovládat větší množství LED. Lze tak vytvořit např. „běžící šipku“ nebo jiný světelný efekt. Na výstupy expanderu připojte požadovaný počet LED (max. 16 ks) vždy podle obrázku:



LED, které chcete rozsvítit, můžete zapnout např. příkazem (pro 8 LED):

```
I2CW b11110100
```

Binární číslice v parametru určují, co svítit bude – na pozici je 0, a co naopak zhasne – 1. Budete-li chtít běžící světlo, můžete jej např pro 8 LED vytvořit takto:

```
1 LET  A=-2
2 I2CW  A ;  DELAY 200
3 LET  A=A*2+1
4 IF  A < -300 THEN  LET  A=-2
5 GOTO  2
```

Zde se využívá kouzlo počítačové matematiky. Číslo -2 je tvořeno samými 1 a jednou 0. Operace A*2+1 pak tuto 0 posouvá o 1 pozici výše.