PC als Instrument, HC06LS9.C

Um jetzt noch aufeinander abgestimmte Töne unserer üblichen Instrumentenstimmung verwenden zu können, genügt eine Frequenztabelle. (Natürlich könnte man jede Frequenz dieser Tabelle bei Bedarf jeweils neu berechnen, eine Tabelle spart aber Rechenzeit.)

```
FLOAT freq_tab[] =
{
  130.8,  /*        C          0  */
  138.6,  /*   Db        C#    1  */
  146.8,  /*        D          2  */
  155.6,  /*   Eb        D#    3  */
  164.8,  /*        E          4  */
  174.6,  /*        F          5  */
  185.0,  /*   Gb        F#    6  */
  196.0,  /*        G          7  */
  207.7,  /*   Ab        G#    8  */
  220.0,  /*        A          9  */
  233.1,  /*   Hb        A#    10 */
  246.9,  /*        H          11 */
  261.7,  /*        c          0 */
  277.2,  /*   db        c#    */
  293.7,  /*        d          */
  311.1,  /*   eb        d#    */
  329.6,  /*        e          */
  349.2,  /*        f          */
  370.0,  /*   gb        f#    */
  392.0,  /*        g          */
  415.3,  /*   ab        g#    */
  440.0,  /*        a          */
  466.2,  /*   hb        a#    */
  493.9,  /*        h          */
  523.3,  /*        c1         */
  0.0,
  0.0
};
```

Das folgende Programm zeigt wie man diese Tabelle dazu benutzt, eine chromatische, eine Dur- und eine Moll-Tonleiter zu spielen.

```
/* HC06LS9.C */
/*
 * Verschiedene Tonleitern im PC
 * =============================
 */
#include <stdio.h>
#include <conio.h>
#include <mylib.h>
#ifndef MYLIB
#include "..\\source\\spk.c"
#endif

VOID main(VOID)
{
  FLOAT *frequenz;
  INT tonnummer;
  UINT tondauer=200;

  clrscr();

  printf("Chromatische Tonleiter C1..C..c, "
         "Start mit Taste\n\n");
  getch();
  frequenz = &freq_tab[0];
  do
  {
    soundf1(*frequenz,tondauer);
    frequenz++;
  }
  while (*frequenz>1.0);
  printf("\n\n");

  printf("\nDUR-Tonleiter C1..C..c, "
         "Start mit Taste\n\n");
  getch();
  frequenz = &freq_tab[0];
  tonnummer = 0;
  do
  {
    soundf1(*frequenz,tondauer);
    switch (tonnummer)
    {
    case 0: case 2: case 5: case 7: case 9:
      frequenz++; tonnummer++;
    default:
      frequenz++; tonnummer++;
    }
    tonnummer %= 12;
  }
  while (*frequenz>1.0);
  printf("\n\n");

  printf("\nMOLL-Tonleiter C1..C..c, "
         "Start mit Taste\n\n");
  getch();
  frequenz = &freq_tab[0];
  tonnummer = 0;
  do
  {
    soundf1(*frequenz,tondauer);

    switch (tonnummer)
    {
    case 0: case 3: case 5: case 8: case 10:
      frequenz++; tonnummer++;
    default:
      frequenz++; tonnummer++;
    }
    tonnummer %= 12;
  }
  while (*frequenz>1.0);
  printf("\n\n");

  getch();
}
```

Das wärs zunächst über den Lautsprecher; wir werden ihn aber noch mehrmals antreffen, etwa beim Schreiben einer BIOS oder DOS-Erweiterung. Auch für den Timer werden wir eine eigene C++-Klasse entwerfen, die genau weiß, welche Einstellung der Timer hat, denn derzeit kann man die Register des Timers nicht zurücklesen.

In der nächsten Folge werden Hardware-Interrupts und einfache Interrupt-Service-Routinen geschrieben. □

Real programmers don't use LISP. Only effeminate programmers use more parentheses than actual code.

Real programmers disdain structured programming. Structured programming is for compulsive, prematurely toilet-trained neurotics who wear neckties and carefully line up sharpened pencils on an otherwise uncluttered desk.

Real programmers have no use for managers. Managers are a necessary evil. Managers are for dealing with personnel bozos, bean counters, senior planners and other mental defectives.

Real programmers scorn floating point arithmetic. The decimal point was invented for pansy bedwetters who are unable to "think big."

Real programmers like vending machine popcorn. Coders pop it in the microwave oven. **Real programmers** use the heat given off by the cpu. They can tell what job is running just by listening to the rate of popping.

Real programmers don't drive clapped-out Mavericks. They prefer BMWs, Lincolns or pick-up trucks with floor shifts. Fast motorcycles are highly regarded.

Real programmers don't believe in schedules. Planners make up schedules. Managers "firm up" schedules. Frightened coders strive to meet schedules. **Real programmers** ignore schedules.

Real programmers don't like the team programming concept. Unless, of course, they are the Chief Programmer.

Real programmers know every nuance of every instruction and use them all in every real program. Puppy architects won't allow execute instructions to address another execute as the target instruction. **Real programmers** despise such petty restrictions.

Real programmers don't bring brown bag lunches to work. If the vending machine sells it, they eat it. If the vending machine doesn't sell it, they don't eat it. Vending machines don't sell quiche.