

Tonfrequenzen und Notenbeispiele:

Noten	Frequenz (Hz)	Alle meine Entchen...
C0	264	C0_4 D0_4 E0_4 F0_4 G0_2 G0_2
D0	297	A0_4 A0_4 A0_4 A0_4 G0_2
E0	330	A0_4 A0_4 A0_4 A0_4 G0_2
F0	352	F0_4 F0_4 F0_4 F0_4 E0_2 E0_2
G0	396	G0_4 G0_4 G0_4 G0_4 C0_1
A0	440	
H0	495	Jingle Bells
C1	528	E0_4 E0_4 E0_2
D1	587	E0_4 E0_4 E0_2
E1	659	E0_4 G0_4 C0_4 D0_4
F1	698	E0_1
G1	784	F0_4 F0_4 F0_4 F0_4
A1	880	F0_4 E0_4 E0_4 E0_8 E0_8
H1	988	E0_4 D0_4 D0_4 E0_4
		D0_2 G0_2

Um Töne zu erzeugen verwenden wir die Systemfunktion Beep die uns von Windows bereitgestellt wird (sie dient vor allem zur Erzeugung von Warntönen).

Zum Beispiel für eine "C0" 1/8 Note verwenden wir die Parameter:

Beep(264, 1000/8); usw.

Aufgabenstellung:

» Entwickeln Sie einen Noteninterpreter, der die in der Tabelle angegebenen Noten versteht und entsprechend abspielt.

» Erstellen Sie ein T-Diagramm für den zukünftigen Noteninterpreter.

» Stellen Sie eine Grammatik für Ihre Eingabesprache auf. Eine mögliche Schreibweise für eine Viertelnote E0 ist „E0_4“.

» Geben Sie auch ein Syntaxdiagramm für Ihre Notensprache an.

» Erstellen Sie einen Kellerautomaten für Ihre Notensprache und simulieren Sie diesen für einige kleine Beispiele.

» Entwickeln Sie nun den Parser mit S-Attributen in VCC und erstellen Sie den Noteninterpreter (Compiler).

» Testen Sie den Interpreter mit Beispieldateien in T-Diag.