Fredagspython: Lydbehandling i Python

Denne uka skal dere få prøve dere på å jobbe med lyd i Python. Det krever også bruk av løkker, som så vidt ble nevnt forrige fredagspython, og som er pensum for neste uke. Vi bruker løkkene for å gå gjennom listene. Det står i læreboka på side 344-345.

I Python kan vi jobbe med .wav-filer. Vi kan laste dem inn, og så må vi gjøre dem om fra noe som heter NumPy-array, til vanlige Python-lister. NumPy er en pakke som har mange spennende vitenskapelige anvendelser.

Vi må laste inn to pakker: scipy og numpy

Dere kan skrive:

```
pip install scipy
```

Da får dere både scipy og numpy. I fila kan vi da skrive:

```
import scipy.io.wavfile
import numpy
```

For å laste inn filene kan vi skrive:

```
innhold = scipy.io.wavfile.read("filnavn.wav")
samplerate = innhold[0]
data = innhold[1].tolist()
```

Innholdet har to deler: samplerate, som ligger på indeks 0, og selve dataene som ligger på indeks 1. .tolist() gjør om fra NumPy-array til ei liste vi kan jobbe med.

For å lagre lydfiler kan vi skrive følgende:

```
scipy.io.wavfile.write("filnavn.wav",samplerate,numpy.asarray(outdata,
dtype="int16"))
```

outdata er dataen, som "data" i eksempelet over.

Manipulering av data

I boka står det at vi kan skrive følgende kode for å gjøre alle verdier som er lavere enn 0 til -30 000, og alle verdier som er høyere eller like 0 til 30 000. Her bruker vi en for løkke med range(), som sier at vi skal gjøre samme sjekk for hvert element i lyd-lista. Vi lager ei ny liste (outputdata) til å holde på de nye verdiene våre:

```
outputdata = []

for i in range(len(data)):
    if (data[i]) > 0:
        outputdata.append(30000)
```

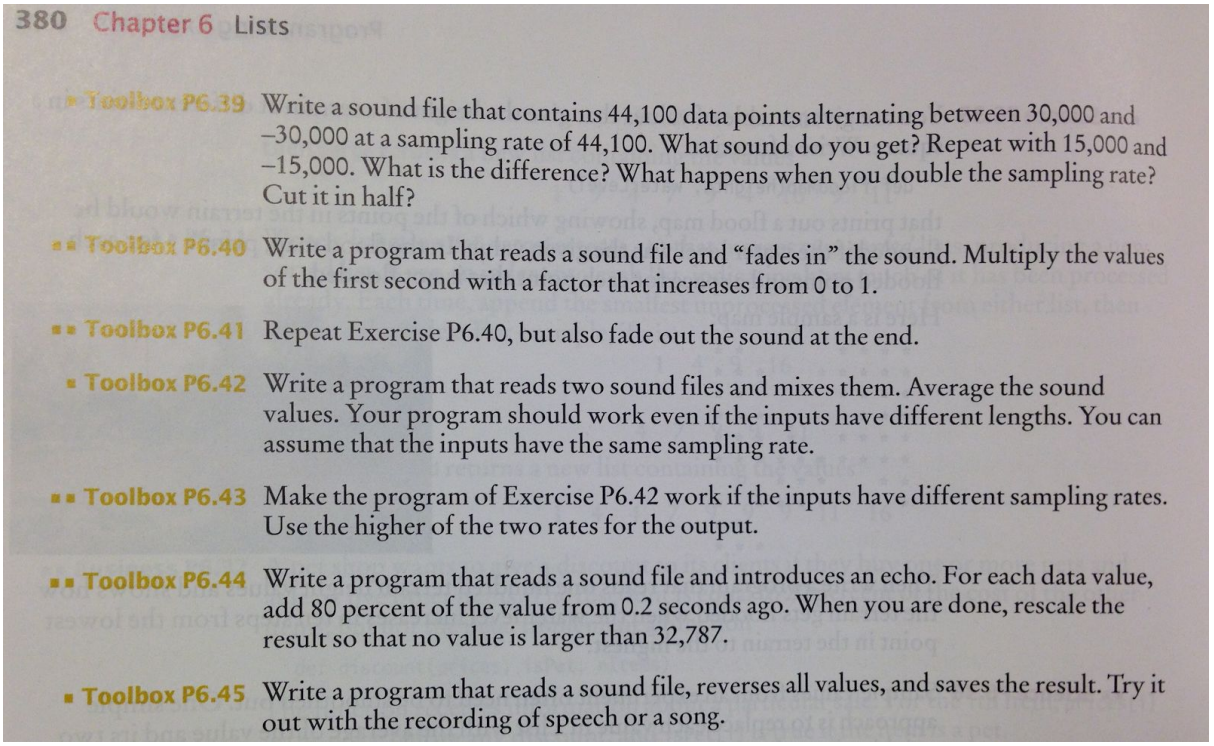
```
        else:
            outputdata.append(-30000)
```

Dette gir en rar effekt på lydfilen, men vi kan fortsatt høre hva den var. Husk å ikke ha for høy lyd på øretelefonene.

BBC har en stor mengde lydfiler tilgjengelig som dere kan øve på.

http://bbcsfx.acropolis.org.uk/

Oppgaver fra boka:



Toolbox P6.39 Write a sound file that contains 44,100 data points alternating between 30,000 and −30,000 at a sampling rate of 44,100. What sound do you get? Repeat with 15,000 and −15,000. What is the difference? What happens when you double the sampling rate? Cut it in half?

Toolbox P6.40 Write a program that reads a sound file and "fades in" the sound. Multiply the values of the first second with a factor that increases from 0 to 1.

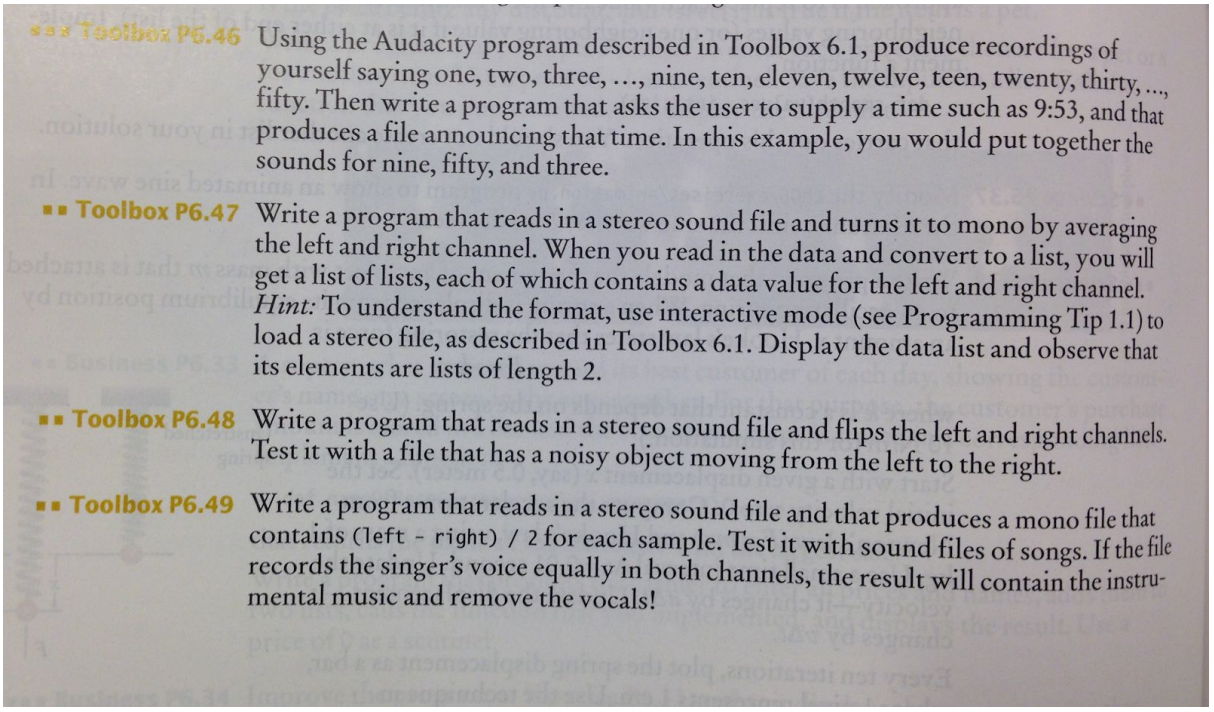
Toolbox P6.41 Repeat Exercise P6.40, but also fade out the sound at the end.

Toolbox P6.42 Write a program that reads two sound files and mixes them. Average the sound values. Your program should work even if the inputs have different lengths. You can assume that the inputs have the same sampling rate.

Toolbox P6.43 Make the program of Exercise P6.42 work if the inputs have different sampling rates. Use the higher of the two rates for the output.

Toolbox P6.44 Write a program that reads a sound file and introduces an echo. For each data value, add 80 percent of the value from 0.2 seconds ago. When you are done, rescale the result so that no value is larger than 32,787.

Toolbox P6.45 Write a program that reads a sound file, reverses all values, and saves the result. Try it out with the recording of speech or a song.



■■■ Toolbox P6.46 Using the Audacity program described in Toolbox 6.1, produce recordings of yourself saying one, two, three, ..., nine, ten, eleven, twelve, teen, twenty, thirty, ..., fifty. Then write a program that asks the user to supply a time such as 9:53, and that produces a file announcing that time. In this example, you would put together the sounds for nine, fifty, and three.

■■ Toolbox P6.47 Write a program that reads in a stereo sound file and turns it to mono by averaging the left and right channel. When you read in the data and convert to a list, you will get a list of lists, each of which contains a data value for the left and right channel. *Hint:* To understand the format, use interactive mode (see Programming Tip 1.1) to load a stereo file, as described in Toolbox 6.1. Display the data list and observe that its elements are lists of length 2.

■■ Toolbox P6.48 Write a program that reads in a stereo sound file and flips the left and right channels. Test it with a file that has a noisy object moving from the left to the right.

■■ Toolbox P6.49 Write a program that reads in a stereo sound file and that produces a mono file that contains `(left - right) / 2` for each sample. Test it with sound files of songs. If the file records the singer's voice equally in both channels, the result will contain the instrumental music and remove the vocals!

Ekstra oppgave: Tale(?)
Prøv å lage en prosedyre som ber brukeren skrive en setning eller et ord, og som så klipper sammen filer og lagrer setningen som en lydfil. Prøv å klippe sammen lyder. For eksempel kan en bruker skrive ordet *tak*, og så setter programmet sammen t, a og k. For å få tak i lydene kan du prøve å klippe egen stemme ved å bruke programmet Audacity. Du kan også bruke lydfiler som vi har klippet for dere. Noen av lydene vil dere kunne høre, mens andre vil høres veldig rare ut. Vær obs på at det i virkeligheten selvsagt er mye mer komplisert, dette er bare for å få et innblikk i noen av problemene. Hvis det blir vanskelig med lyder, kan du endre programmet slik at det passer med de lydfilene dere får til å bruke.

Tenk over:
Hvilke lyder er vanskelige å få til å fungere? Hvilke andre problemer får du når du skal konvertere fra tekst til tale?

For de som har lyst til å lese mer anbefaler vi kapittel 7-11 i *Speech and Language Processing*, av Jurafsky og Martin, 2009.