Exercise A: FFT, Spectral Leakage, Zero Padding

Using a software program like MATLAB, Octave, etc., implement the following tasks.
For each part, you need to prepare and send the source code, the image (.png, .bmp, .pdf, …) of the required time-domain or frequency-domain plots, and your comments on the result of each part, if needed. Name the files according to the numbers of corresponding parts, and send all of them as a .zip file to the address below:

xamiri@fi.muni.cz

Deadline is April 10th.

1) Generate two separate 64 length buffers of the two sinusoids:

$$x_1(t) = 100\sin(2\pi 101.56t) \qquad\qquad x_2(t) = 2\sin(2\pi 156.25t)$$

Use a sampling frequency of 1 kHz.

2) Evaluate the FFT of each of these signals using a rectangular window, and determine which frequency bins have the highest peaks.

3) Repeat the FFT evaluations done in part 2, but using a 1024 point FFT. (Applying a 1024 point FFT to a 64 length data buffer has the effect of appending (1024-64) zeros to the end of the data, which increases the number of points at which the spectrum is evaluated.)

4) The Hamming window is defined as:

$$w(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N}\right) \qquad for\ n = 0,1,2,\dots,N-1$$

Generate and plot (stem) the Hamming window.

5) Now repeat parts 2 and 3, but applying first a Hamming window to the 64 length data buffers before evaluating the FFTs. Observe the outputs of the FFTs.

6) Now add these two sine waves together, and apply both 64-length and 1024-length FFTs, with both rectangular and Hamming windows. Observe how "spectral leakage" can mask a small signal in a large one.

7) Now add Gaussian noise to the combined signals, with a variance of 10. Compare the results of evaluating the 1024 point FFT on a 64-length data buffer (i.e. with zero padding), with a 1024 point FFT applied to a 1024-length data buffer (i.e. generate more data).

Hint: in MATLAB, to generate the noise, try: noise = sqrt (variance) * randn();

8) Try using the same FFT analysis done above but using "junglemono.wav" sound file as the source. The file contains two chunks: sampled data ("data"), and sample rate ("fs") in Hz.