Lab 2

1. Plot on the same figure the signals $sin(2\pi ft)$ and $cos(2\pi ft)$, with $f = 0.3$ and $t \in [0, 10]$.

2. Load the audio file ‘Kalimba.mp3’ in the Matlab workspace. Only load samples between 1 and 200000 (to avoid out of memory error)

   a. Play it through the computer’s audio device
   b. Change the sampling frequency to half the correct value, and play again. How will the sound be changed?
   c. Amplify the sound by multiplying the data by 4. Play the sound and observe the difference.
   d. Swap the left and right channels (it’s a stereo file) and play the sound again.

Lab 3

1. Load the `Lena` image (use `imread()`), convert it to double, convert it to grayscale, scale the values to the [0, 1] range, and display the image (use `imshow()`).

2. Construct a new image based on the `Lena`, but in which each pixel value is set as a linear combination of the original pixels around it, as in the following equation:

$$\begin{aligned} y[i,j] = \frac{1}{9}x[i-1,j-1] + \frac{1}{9}x[i-1,j] + \frac{1}{9}x[i-1,j+1] \\ +\frac{1}{9}x[i,j-1] + \frac{1}{9}x[i,j] + \frac{1}{9}x[i,j+1] \\ +\frac{1}{9}x[i+1,j-1] + \frac{1}{9}x[i+1,j] + \frac{1}{9}x[i+1,j+1] \end{aligned}$$

   Ignore the first and last row/column, if needed.

   Display the resulting image in a new window. How did it change?

3. Repeat Exercise 2 but change the values of the coefficients to

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Lab 4

1. Load the `Lena` image (use `imread()`), convert it to a grayscale image, convert it to `double` type, adapt the values to the $[0, 1]$ range, and display it (use `imshow()`).
2. Create a video sequence by scrolling the Lena image circularly to the right, by 3 pixels at every frame. Display the video at 25fps.

Code template for creating a video sequence in Matlab:

```
height = ...; % desired height
width  = ...; % desired width
NoF    = ...; % desired number of frames
% an array of size height x width x 1 x NoF:
video  = zeros(height, width, 1, NoF);
for i = 1:NoF
    video(:,:,:,i) = ... the frame number i ... ;
end

% Play the sequence
implay(video);
```

Lab 5

1. Create a function `mysys1()` that implements the following system H_1:

$$y[n] = H_1\{x[n]\} = \frac{1}{4}x[n] - \frac{1}{2}x[n-1] + \frac{1}{4}x[n-2]$$

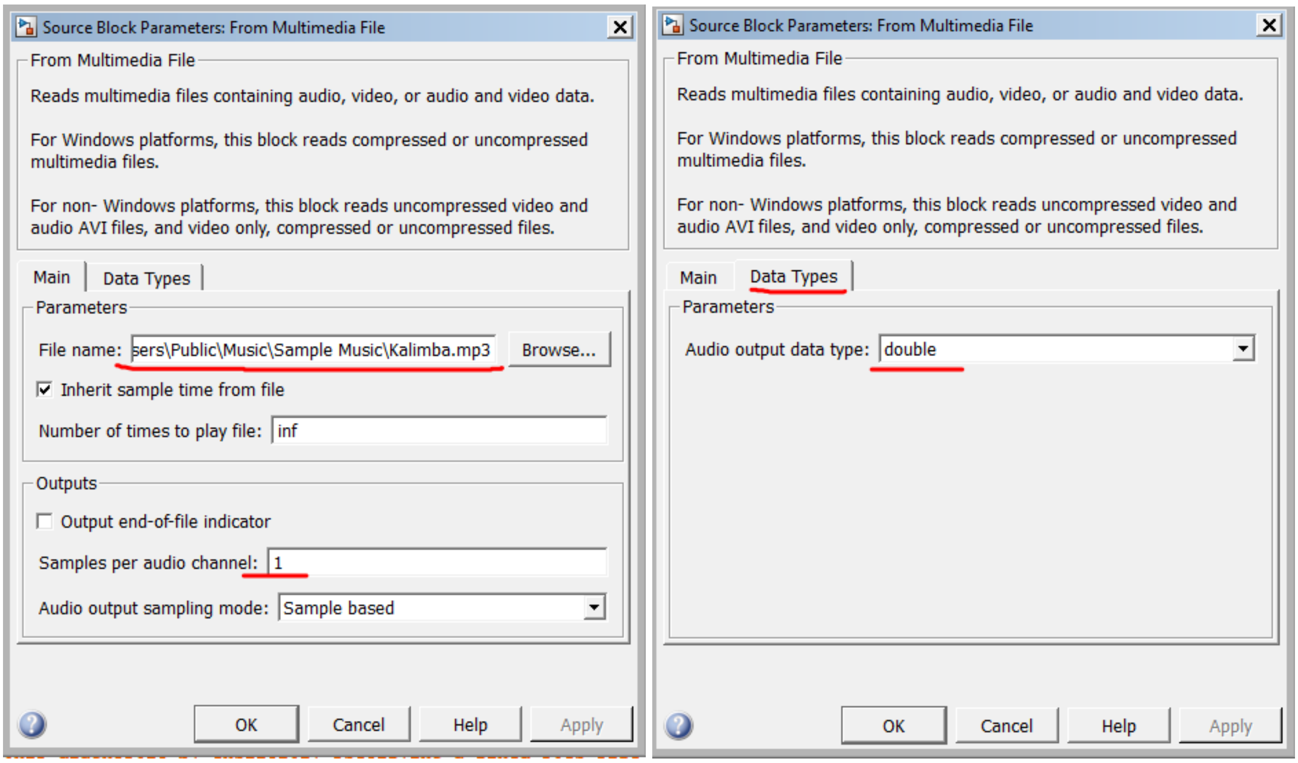
   - the function takes 1 input argument `x` and outputs 1 result vector `y`
2. In a separate script, test the linearity of this system in the following way:
   - generate two random vectors `x` and `y` and two random numbers `a` and `b`
   - apply the function `mysys1()` to `a*x`, `b*y`, and `a*x + b*y`, and check if the results verify the linearity equation
   - the check shall be repeated for 5 times, with 5 different randomly generated data
   - is the system linear?

Lab 6

1. Create a Simulink model to implement the following system H_1:

$$y[n] = H_1\{x[n]\} = \frac{1}{4}(x[n] + x[n-1] + x[n-2] + x[n-3])$$

   - the system should be implemented as a Subsystem block with one input and one output signal
2. Apply the system to the audio data (mp3 file) loaded with `FromMultimediaFile` block and play the resulting output using `Buffer` and `ToAudioDevice` blocks.
   - make sure you set the properties of the `FromMultimediaFile` block as shown below:



Lab 7

1. Create a Simulink model to implement the following system H_1:

$$y[n] = H_1\{x[n]\} = 0.8y[n-1] + 0.25x[n] + 0.1x[n-1]$$

   - the system should be implemented as a Subsystem block with one input and one output signal
2. Test linearity of this system as follows:
   - create three copies of the system inside the model (copy/paste)
   - use two random input vectors `x` and `y` (use two *Random* blocks)
   - apply input signals `x`, `y` and `x+y` to the three copies of the system
   - add the outputs of the systems which have `x` and `y` as inputs, then subtract the output of the system which has `x + y` as input
   - show the resulting signal. Is the system linear?

Lab 9

1. Generate a 100 samples long signal `x` defined as $x[n] = 0.7\cos(2\pi f_1 n) + 1.2\sin(2\pi f_2 n)$, with $f_1 = 0.05$ and $f_1 = 0.1$.
   a. Plot the signal in the top half of a figure (use `subplot()`).
   b. Compute the Fourier series coefficients with `fft()` and plot their magnitude in the lower half of the figure.
2. Take the Fourier series coefficients of the above signal `x`, and keep only the coefficients of the DC + first two sinusoidal components. Generate the signal from the Fourier coefficients with `ifft()` and plot it. What is the resulting signal?

Lab 11

1. Use the Filter Design tool in Matlab (call `fdatool` in command line) to design a Low-Pass filter of order 5, IIR, with cutoff frequency 0.1. Export the coefficients to the Matlab workspace.

2. Generate a signal composed of 30 values of 1 followed by 30 values of 0. Filter the signal with the designed filter (use `filter()`). Plot a figure with 2 subfigures showing the original signal and the filtered signal.

Lab 12

1. Use the Filter Design tool in Matlab (`fdatool`) to design an oscillator with frequency 0.05. Implement it in Simulink, visualize & play the output signal.
   - design a system of order 2 with 2 conjugate poles placed **on the unit circle** at the correct frequency, and 2 zeros at low & high frequencies
   - implement the system in Simulink. You can **omit the input signal** (not necessary, an oscillator has no input)
   - set a non-zero initial condition in the system, to start-up the oscillator
   - play the resulting output using `Buffer` and `ToAudioDevice` blocks. What frequency do you hear when running the simulation?