Full Name: ______________________________

EEL 4750 / EEE 5502 (Fall 2018) – Code #03 **Due Date:** **Oct. 4, 2018**

Question #1: *(1 pts)* How many hours did you spend on this homework?

Question #2: *(7 pts)* *Correlation and the Discrete-Time Fourier Transform*
Consider the expression for correlation (also known as cross-correlation) between $x[-n]$ and $y[n]$,

$$c[n] = x[-n] * y[n]$$

In this assignment, we discuss the properties of correlation in the discrete-time Fourier transform (DTFT) domain. For the questions below, let $X(\omega)$ and $Y(\omega)$ be the discrete-time Fourier transform of signals $x[n]$ and $y[n]$, respectively.

(a) Show that the discrete-time Fourier transform of $x[n] * y[n]$ is equal to $X(\omega)Y(\omega)$.

(b) Show that if $x[n]$ is real (i.e., there are no imaginary numbers), then the real part of $X(\omega)$ is even (i.e., $X(\omega) = X(-\omega)$). Hint: consider the real part of the DTFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

under these conditions.

(c) Show that if $x[n]$ is real (i.e., there are no imaginary numbers), then the imaginary part of $X(\omega)$ is odd (i.e., $X(\omega) = -X(-\omega)$).

(d) Use the previous two results to show that if $x[n]$ is real (i.e., there are no imaginary numbers), then the DTFT of $x[-n]$ is $X^*(\omega)$. You may start from the time-reversal property in the transform table on the course website.

(e) Use the previous result to show that the discrete-time Fourier transform of $c[n]$ is $X^*(\omega)Y(\omega)$.

Question #3: *(6 pts)* *The Speed of Fourier*
Finding segments in large datasets (e.g., finding a particular face in numerous pictures) can be overly time consuming with raw convolution. Fourier methods can speed this up, particularly with the Fast Fourier transform (FFT). Note that we treat the FFT as a "black box" in this assignment.

(a) Create a random signal $x[n]$ with the MATLAB code

```
x = cos(4*pi/N*(1:N)) + randn(1, N);
```

where `N` is the length of the signal. Let `N = 1000000`. Plot a random signal with `plot(x)`.

(b) Create a vector corresponding to the impulse response of a 10000-point running average filter

$$h[n] = \frac{1}{M}\sum_{k=0}^{M}\delta[n-k]\ .$$

Let `M = 10000`. Use `conv` to filter $x[n]$ and get an output $y_1[n]$. Plot $x[n]$ and $y_1[n]$ with

```
plot(nx, x); hold on; plot(ny1, y1); hold off;
```

Determine what `nx` and `ny1` should be from your coding assignment #2 results.

(c) How does `y1` compare (how is it similar, how is it different?) with `x` in your plots?

(d) Now use the Fast Fourier transform function `fft` to perform the same operation, based on your result from Question#2(a). Specifically,

```
h2 = [h; zeros(1,N-M)];
y2 = ifft(fft(x).*fft(h2));
```

Note that we add zeros to `h` (creating `h2`) to make `h2` and `x` the same size. Plot $x[n]$, $y_1[n]$, and $y_2[n]$ with

```
plot(nx, x); hold on; plot(ny1, y1); plot(nx,y2) hold off;
```

Note that `y2` and `y1` will not be exactly the same. We will discuss why in class when we get to the discrete Fourier transform.

Question #4: *(7 pts)* *Name that tune!*

Included with this lab is 39 mp4 files by Peter Rudenko from the free music archive (http://freemusicarchive.org/). To load these files into MATLAB use the `audioread` command. For example,

```
[y, Fs] = audioread(['rudenko_01.mp4']);
```

If you want to use a variable to control the file number, you can use the commands:

```
ii = 1;
[y, Fs] = audioread(['rudenko_' num2str(ii, '%02i') '.mp4']);
```

The output of the `audioread` function provides `y` (the audio signal) and `Fs` (the sampling rate of the audio signal). The sampling rate is the number of values (i.e., samples) per second in the signal.

To visually see the audio signal with correct axis labels, type the commands

```
t = 1/Fs:1/Fs:length(y)/Fs;
plot(t, y);
xlabel('Time [sec]')
ylabel('Amplitude')
```

The variable `t` describes the time-axis of the audio.

Also included is a p-file (an obfuscated m-file) function

```
[findme, Fs] = get_tune(ufid);
```

Note that to run this function, all 39 Rudenko files must be in the current directory. The input to this function `ufid` will be a **string** with your UFID. The output of the function `findme` is an audio segment from the one of the 39 audio files. The second output `Fs` is the sampling rate of the audio (this should be equal for all files).

Replace `ufid` with your UFID (in single quotes) and run this function to extract an audio segment that is unique to you. To play the audio segment, use the command:

```
sound(findme, Fs)
```

Write a MATLAB script that utilizes the capabilities of correlation from the coding assignment #2 and capabilities of the FFT that we learned in the previous questions to determine which music file contains your unique audio segment `findme`.

Note that even when using the `fft` function, your script may take a few minutes to fully execute. Submit your .m file used to locate the unique audio segment and provide the name of the matching music file.