Lab 2

Kevin Le

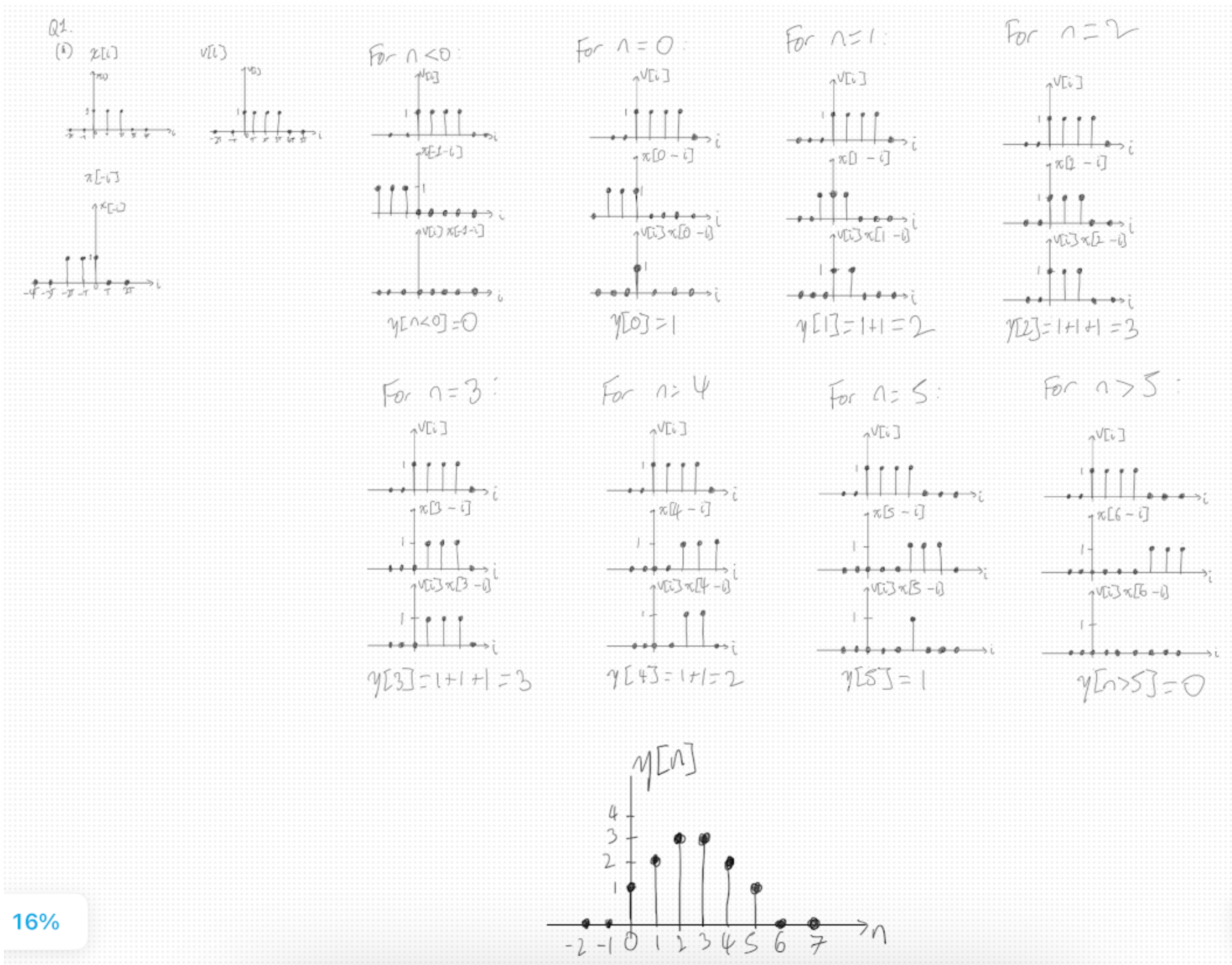
McMaster University

ELECENG 3TP3

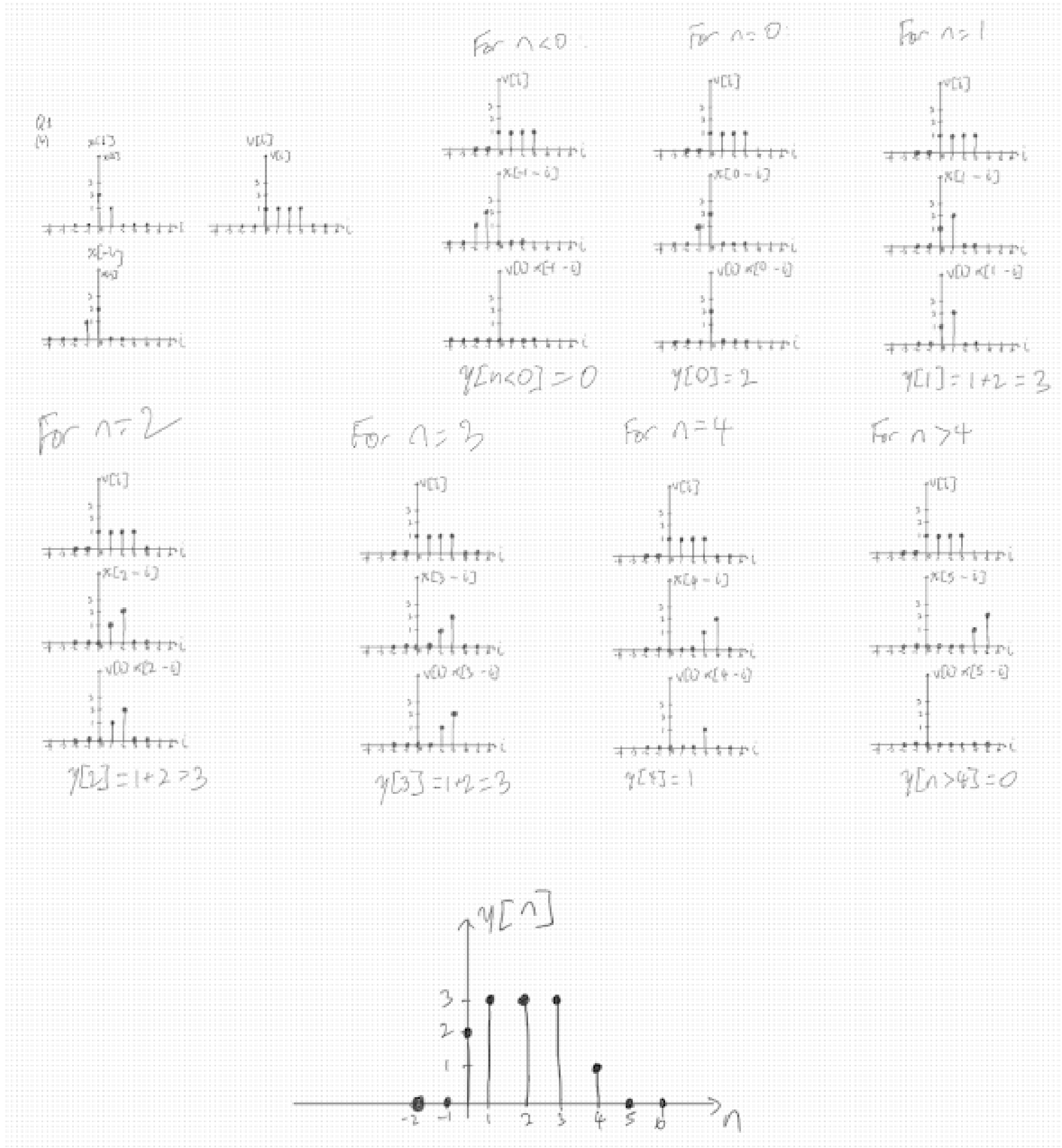
October 10th, 2023

1. Textbook Question
Manual Computation:

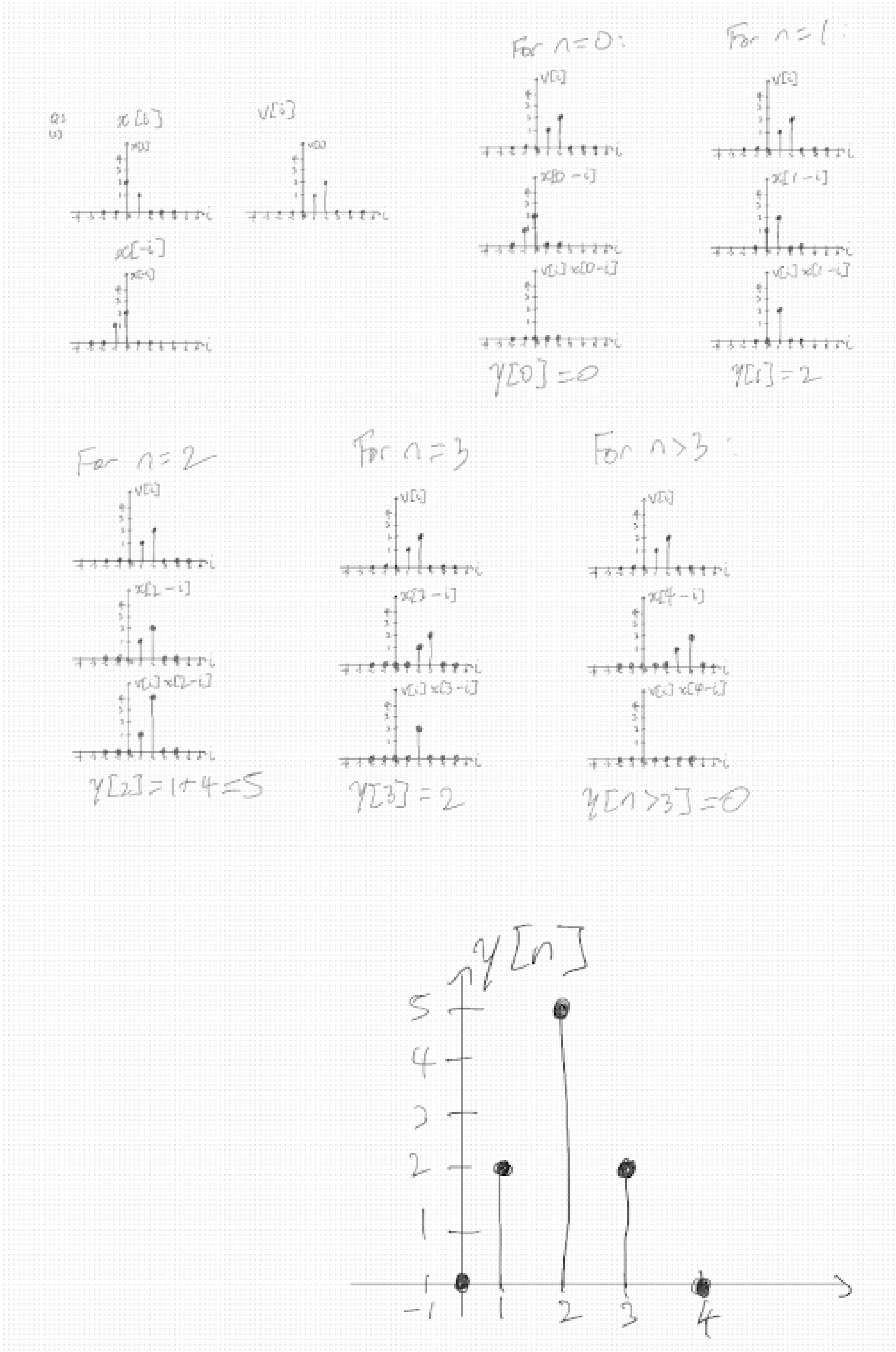
Part A:



Part B:


For n<0
For n=0
For n=1
y[n<0] = 0
y[0] = 2
y[1] = 1+2 = 3
For n=2
For n=3
For n=4
For n>4
y[2] = 1+2 = 3
y[3] = 1+2 = 3
y[4] = 1
y[n>4] = 0
y[n]
n

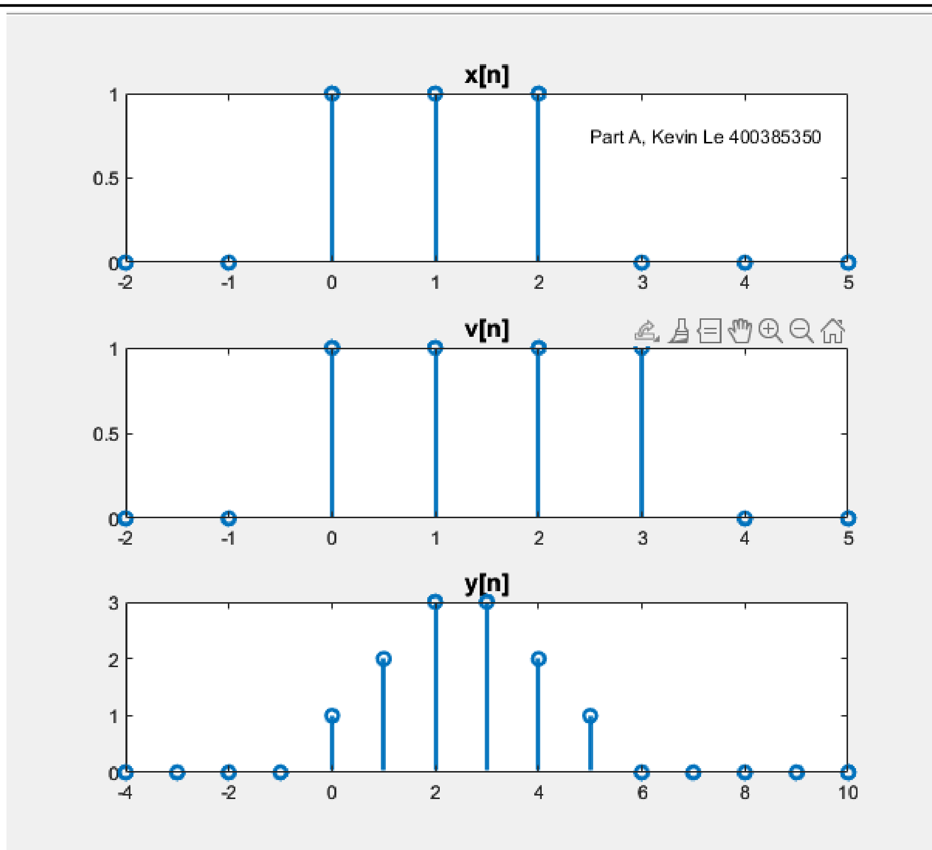
Part C:


For n=0:
For n=1:
x[i]
v[i]
x[-i]
y[0]=0
y[1]=2
For n=2
For n=3
For n>3:
y[2]=1+4=5
y[3]=2
y[n>3]=0
y[n]

MATLAB Code and Plot:

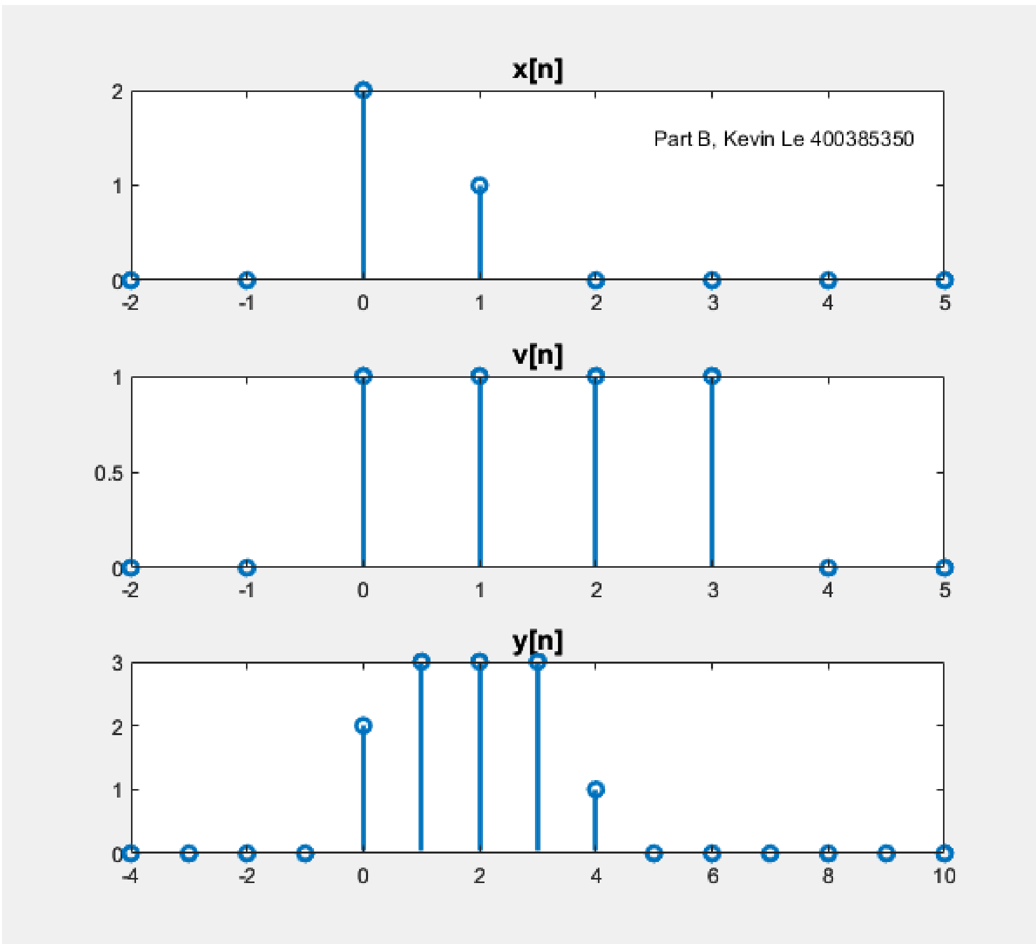
Part A:

```
%Part A
f = SimpleFunctions();
n = -2:5;
n_y = -4:10; %because length(y) = length(x)+length(v) - 1
%representing x[n] and v[n] with unitstep functions
x = f.unitstep(n) - f.unitstep(n-3);
v = f.unitstep(n) - f.unitstep(n-4);
y = conv(x,v);
%plotting x[n]
subplot(3,1,1);
stem(n,x,'LineWidth',2);
title('x[n]','FontSize',12);
text(2.5,0.75,'Part A, Kevin Le 400385350','FontSize',9);
%plotting v[n]
subplot(3,1,2);
stem(n,v,'LineWidth',2);
title('v[n]','FontSize',12);
%plotting y[n]
subplot(3,1,3);
stem(n_y,y,'LineWidth',2);
title('y[n]','FontSize',12);
```



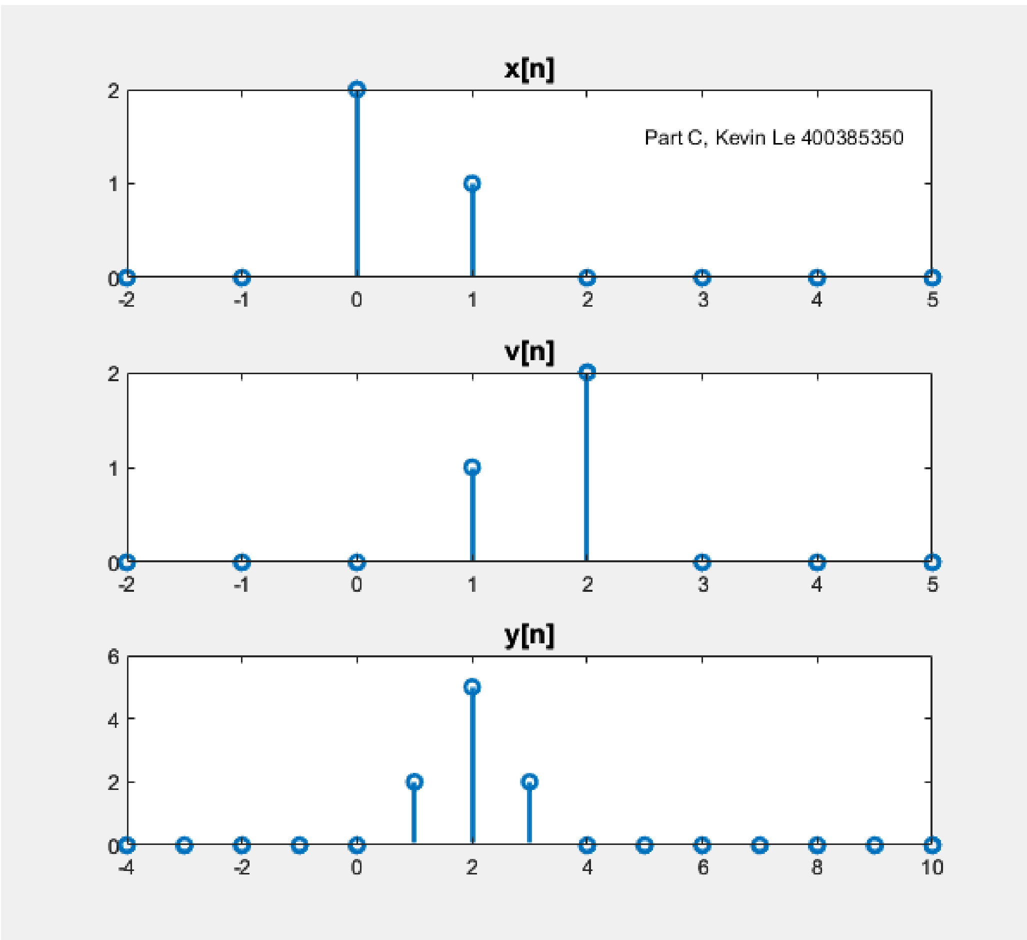
Part B:

```
%Part B
f = SimpleFunctions();
n = -2:5;
n_y = -4:10; %because length(y) = length(x)+length(v) - 1
%representing x[n] and v[n] with impulse and unitstep functions
x = 2*f.delta(n) + f.delta(n-1);
v = f.unitstep(n) - f.unitstep(n-4);
y = conv(x,v);
%plotting x[n]
subplot(3,1,1);
stem(n,x,'LineWidth',2);
title('x[n]','FontSize',12);
text(2.5,1.5,'Part B, Kevin Le 400385350','FontSize',9);
%plotting v[n]
subplot(3,1,2);
stem(n,v,'LineWidth',2);
title('v[n]','FontSize',12);
%plotting y[n]
subplot(3,1,3);
stem(n_y,y,'LineWidth',2);
title('y[n]','FontSize',12);
```



Part C:

```
%Part C
f = SimpleFunctions();
n = -2:5;
n_y = -4:10; %because length(y) = length(x)+length(v) - 1
%representing x[n] and v[n] with impulse functions
x = 2*f.delta(n) + f.delta(n-1);
v = f.delta(n-1) + 2*f.delta(n-2);
y = conv(x,v);
%plotting x[n]
subplot(3,1,1);
stem(n,x,'LineWidth',2);
title('x[n]','FontSize',12);
text(2.5,1.5,'Part C, Kevin Le 400385350','FontSize',9);
%plotting v[n]
subplot(3,1,2);
stem(n,v,'LineWidth',2);
title('v[n]','FontSize',12);
%plotting y[n]
subplot(3,1,3);
stem(n_y,y,'LineWidth',2);
title('y[n]','FontSize',12);
```



2. Add echo to audio file

```
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal); % number of samples in the signal
T = 1/Fs; % sampling period in seconds
t = [0:L-1]*T; % time vector in seconds
Te = 200; % in milliseconds
alpha = 1/2; % reduced amplitude factor
% convert echo delay from msec to sec,
% then divide by sampling period to find number of delayed samples
delay = floor(Te/1000/T); % must floor because signals are discrete
% 'echo' equals 'signal' shifted to the right by 'delay' steps,
% with zeros filled into the empty space at the beginning of the vector
echo = zeros(L+delay,1); % size of echo becomes L+delay
echo(delay+1:end) = alpha*signal(1:end); % signal times reduced amplitude factor
alpha
%size of signal must change because although signal have stopped, there
%will still be signal from echo. fills the end of signal with zeros so
%signal size equals echo size
signal_withzeros = zeros(L+delay,1);
signal_withzeros(1:L) = signal(1:end);
signalplusecho = signal_withzeros + echo;
% rescale
signalplusecho = signalplusecho/max(abs(signalplusecho));
%write a new wav file
audiowrite('speechwithecho.wav', signalplusecho, Fs);
```

3. Add echo to audio file using convolution

```
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal); % number of samples in the signal
T = 1/Fs; % sampling period in seconds
t = [0:L-1]*T; % time vector in seconds
Te = 200; % in milliseconds
alpha = 1/2; % reduced amplitude factor
% convert echo delay from msec to sec,
% then divide by sampling period to find number of delayed samples
delay = floor(Te/1000/T); % must floor because signals are discrete
f = SimpleFunctions();
%delta(t) for the original signal and alpha*delta(t-delay*T) for the echo
impulse = f.delta(t) + alpha*f.delta(t-delay*T);
signalplusecho = conv(signal,impulse); %convolution gives signal plus echo
%shorten to only length portion with sound
signalplusecho = signalplusecho(1:L+delay);
% rescale
signalplusecho = signalplusecho/max(abs(signalplusecho));
%write a new wav file
audiowrite('speechwithechoconvolved.wav', signalplusecho, Fs);
```

Explain your choice of impulse response and include that in your writeup.

Knowing that any signal convolved with delta(t) is just that signal, any signal convolved with delta(t-n) is that signal shifted to the right by n, and any signal convolved with a*delta(t) is that signal amplified by a. So convolving the signal with delta(t)+alpha*delta(t-Te) creates a signal with the original signal plus the original signal amplified by alpha with a Te second delay.

4. Experiment with different values of Te when alpha is equal to 1. How small does Te have to be before the quality of the speech is acceptable? Does your answer change when the value of alpha is decreased?

With alpha equal to 1, I think Te at 60ms or smaller gives an acceptable quality of speech although the quality still isn't the greatest.
When I change the alpha value from 1 to ⅓, I find that the quality at Te = 60ms is better, and even Te at 100ms sounds acceptable.
When I change the alpha value to 1/10, I find that the quality at Te=60ms and Te=100ms sound great. It is harder to notice the echo compared to alpha equal 1 and 1/3.

5. Add reverberation to audio file using convolution

```
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal); % number of samples in the signal
T = 1/Fs; % sampling period in seconds
t = [0:L-1]*T; % time vector in seconds
Te = 200; % in milliseconds
alpha = 1/2; % reduced amplitude factor
Ne = 5; % number of echos
% extending time vector because current time vector may not be long enough
% for Ne number of echos
t = [0:T:Ne*Te/1000];
% convert echo delay from msec to sec,
% then divide by sampling period to find number of delayed samples
delay = floor(Te/1000/T); % must floor because signals are discrete
f = SimpleFunctions();
impulse = f.delta(t); % original signal
for i=1:Ne % Ne number of impulses for Ne number of echos
% exponentially decreasing in amplitude with alpha^i and Te ms apart
impulse = impulse + (alpha.^i)*f.delta(t-i*delay*T);
end
signalplusecho = conv(signal,impulse); %convolution gives signal plus echo
%shorten to only length portion with sound
signalplusecho = signalplusecho(1:L+delay*Ne);
% rescale
signalplusecho = signalplusecho/max(abs(signalplusecho));
%write a new wav file
audiowrite('speechwithreverberation.wav', signalplusecho, Fs);
```

6. Experiment with different values of Te when alpha is equal to 1. How small does Te have to be before the quality of the speech is acceptable? Does your answer change when the value of alpha is decreased?

With reverberation, with alpha set to 1 and Ne set to 5, I find that Te at 6ms or smaller gives an acceptable quality of speech but the sound quality is still terrible.
When I change alpha from 1 to ⅓, the quality is much better at Te=6ms. Even at Te=100ms, the quality is still acceptable but the echo is still noticeable.
When I change alpha to 1/10, the quality is great at Te=100ms and the echo is hardly noticeable.

Even when I leave alpha=1/10 and Te=100ms and change Ne to 15, the sound quality is still great and echo is hardly noticeable. There is almost no difference between Ne=5 and Ne=15 at this setting.
When I change alpha back to ⅓ and Te=100ms, Ne=5 vs Ne=15 also makes almost no difference in sound.
At alpha=1 and Te=6ms, the difference between Ne=5 and Ne=15 starts to become noticeable. at Ne=15, the sound gets a lot more distorted and words become harder to hear.

Files Included:

my_speech_clip.wav: original sound signal

speechwithecho.wav: original sound signal with an added alpha=½ and Te=200ms echo

speechwithechoconvolved.wav: original sound signal with an added alpha=½ and Te=200ms echo, created using convolution

speechwithreverberation.wav: original sound signal with 5 added echos with alpha,i=(½)^i and Te,i=i*200ms, where i ranges from 1 to 5, created using convolution