DIGITAL SIGNAL PROCESSING: COMPLETE SYSTEM

$x(t)$ =Continuous-time (analog) signal.
EX: Audio (from microphone) signal.
Goal: *Digitally* filter this signal $x(t)$.

$\tilde{x}(t)$ =Lowpass-filtered version of $x(t)$.
How? Analog filter; use EECS 215 ideas.
Why? Remove frequencies $> \frac{1}{2}\left(\substack{\text{SAMPLING}\\ \text{FREQUENCY}}\right)$
$\rightarrow$ ensures there will be no aliasing.

$x[n]$ =Discrete-time (sampled) signal.
How? $x[n] = \tilde{x}(t = n\Delta); \Delta = \substack{\text{SAMPLING}\\ \text{INTERVAL}}.$
Why? We can now use EECS 206 ideas.
Note: Can recover $\tilde{x}(t)$ from $x[n]$ exactly, due to the anti-alias filter.

$\hat{x}[n]$ =Quantized version of $x[n]$.
How? Round $x[n]$ to nearest of 2^B levels. B=#bits used to represent numbers.
Why? To send/store bits, not numbers.
Note: Can't recover $x[n]$ from $\hat{x}[n]$, but the error is usually negligible.

$y[n]$ =Filtered version of input $\hat{x}[n]$.
How? $y[n] = h[n] * \hat{x}[n] = \sum_i h[i]\hat{x}[n-i]$.
$h[n]$ =impulse response of digital filter.
Why? Lowpass filter→remove some noise.
Notch filter→remove 60 Hz "hum."

$y(t)$ =Interpolated $y[n]$=analog output.
How? Use zero-order hold (constant interpolation)
OR: Linear interpolation (between samples $y[n]$)
OR: Exact formula (*Sampling* handout).