```
function [audioSignals] = preProcess(exampleID)
    %create audio signals cell array
    audioSignals = cell(10, 1);

    % Find the duration of the longest audio
    maxRows = 0;
    for i = 0:9
        [y, ~] = audioread(sprintf("samples/%d_16_%d.wav", i, exampleID));
        [rows, ~] = size(y);
        if(maxRows < rows)
            maxRows = rows;
        end
    end

    for i = 0:9
        % y is the audio signal
        % Fs is the sampling frequency
        [y, Fs] = audioread(sprintf("samples/%d_16_%d.wav", i, exampleID));

        % Normalize the signal based on the maximum amplitude
        min_y = min(y);
        max_y = max(y);
        y = (y - min_y) / (max_y - min_y);
        y = 2 * y - 1;
        % [rows, cols] are the original dimensions of y (there are #rows
samples)
        [rows, ~] = size(y);

        % Split the audio signal in small frames in order to extract certain
features
        % Frame size in seconds
        frameSize = 0.001;
        % Get number of samples per frame
        frameSamples = round(frameSize * Fs);
        % Calculate number of frames
        numFrames = floor(rows / frameSamples);

        % frameEnergy is an array with the energy value of every frame
        frameEnergy = getFrameEnergy(y, frameSamples, numFrames);

        energyThreshold = 0.1;

        % Find first index of the first frame with energy above threshold
        startFrame = find(frameEnergy > energyThreshold, 1);
        % Get the TIME index of the first frame with energy above threshold
        startSample = (startFrame - 1) * frameSamples + 1;

        % Trim y in order to remove low energy values
        y = y(startSample:end);
        % Add silence to the end of y
        concatY = zeros(maxRows - length(y), 1);
        y = [y; concatY];

        audioSignals{i + 1} = y;
    end
end
```