```
function idx = Check_2dB( driverOverall_dBA, passengerOverall_dBA, handles, frontOverall_dBA )
% S7.1.3: Identify first four valid tests within 2dBA.
%
% (a) For each valid test sound file identified in S7.1.2, determine a maximum
% overall SPL value, in decibels. Each SPL value will be reported to the
% nearest tenth of a decibel.
%
% (b) Compare the first four left-side SPL values from S7.1.3(a) of this
% paragraph, and determine the range by taking the difference between the
% largest and smallest of the four values. In the same manner, determine the
% range of SPL values for the first four right-side and the first four front-
% center sound files. ****** If the range for the left side, right side, and
% front-center are all less than or equal to 2.0 dB, then ****** the twelve
% sound files associated with the first four valid tests will be used for the
% one-third octave band evaluations in S7.1.5. and S7.1.6. If the range of
% the SPL values for the left side are not within 2 dBA, or for the right
% side are not within 2 dBA, or for the front-center of the vehicle are not
% within 2 dBA, an iterative process will be used to consider sound files
% from additional sequential tests until the range for all three microphone
% locations are within 2 dBA for the same sequence number recordings for all
% three locations

% S7.2 Stationary vehicle in reverse gear.  Test the vehicle per S7.1.1
% through S7.1.5 except that the rear plane of the vehicle is placed on the
% PP' line, no center microphone is used, and the vehicle's transmission gear
% selector is placed in the 'Reverse' position.  The minimum sound level
% requirements for the Reverse test condition are contained in S5.1.2,
% Table 2, for 4-band compliance and in S5.2, Table 6, for 2-band compliance.

sampleSize = 4; % Start by looking at groups of four (may increase within the loop)
while sampleSize > 3
    if sampleSize > length ( driverOverall_dBA )
       set( handles.display2dBError, 'Visible', 'on' );
       set( handles.display2dBError, 'String', 'No combination of 4 samples are within 2dB' );
       set( handles.display2dBError, 'ForegroundColor', [1, 0, 0 ] );
       set( handles.pushbuttonCompute, 'Enable', 'on' );
       break

    end
    idxNumberSpace = 1 : sampleSize; % Gives list of indeces to be considered:
    %                                  1-4 gives only one option for idx (1 2 3 4 )
    %                                  1-5 = options 5 options ( 1 2 3 4 )
    %                                                          ( 1 2 3 5 )
    %                                                          ( 1 2 4 5 )
    %                                                          ( 1 3 4 5 )
    %                                                          ( 2 3 4 5 ), etc
    idx = nchoosek( idxNumberSpace, 4 );
    % Each row gives one possible combination of 4 samples from the (equal or) larger sampleSize

    % Check Driver Side
```

```
    for ink = 1 : size( idx, 1 )
        %this is using 1 decimal precision, don't need to round to 1 decimal again
        checkDriver( ink ) = max( driverOverall_dBA( idx( ink, : ) ) ) - min( driverOverall_dBA( idx( ink, : ) ) );
    end
    boolSort = checkDriver <= 2;
    idx = idx( checkDriver <= 2, : );
    if isempty( idx )
        sampleSize = sampleSize + 1;

    else % Driver OK, Check Passenger Side
        for ink = 1 : size( idx, 1 ) % Just compare the ones that are within 2 dB for driver side
            checkPassenger( ink ) = max( passengerOverall_dBA( idx( ink, : ) ) ) - min( passengerOverall_dBA( idx( ink, : ) ) );
        end
        idx = idx( checkPassenger <= 2, : ); % This now includes the idxs that meet requirement for both sides
        if isempty( idx )
            sampleSize = sampleSize + 1;

        elseif exist( 'frontOverall_dBA', 'var' ) % Passenger OK, Check Front Side
            for ink = 1 : size( idx, 1 ) % Just compare the ones that are within 2 dB for driver and passenger side
                checkFront( ink ) = max( frontOverall_dBA( idx( ink, : ) ) ) - min( frontOverall_dBA( idx( ink, : ) ) );
            end
            idx = idx( checkFront <= 2, : ); % This now includes the idxs that meet requirement for all sides
            if isempty( idx )
                sampleSize = sampleSize + 1;

            else % Front Side OK, idx meet tolerance for all sides
                % If idx is just one row, then unambiguous, if more than one
                % need to take the one with the lowest idx sum (or the first one
                % with the lowest idx sum)
                idxSum = sum( idx, 2 );
                [ ~, idxMin ] = min( idxSum );
                idx = idx( min( idxMin ), : );
                sampleSize = 0;
            end

        else % Passenger OK, No Front Side, idx meet tolerance for all sides
            % If idx is just one row, then unambiguous, if more than one
            % need to take the one with the lowest idx sum (or the first one
            % with the lowest idx sum)
            idxSum = sum( idx, 2 );
            [ ~, idxMin ] = min( idxSum );
            idx = idx( min( idxMin ), : );
            sampleSize = 0;
        end
    end
end
```