```
function keys = dtmfrun(xx,L,fs)
%DTMFRUN    keys = dtmfrun(xx,L,fs)
%    returns the list of key numbers corresponding
%      to the DTMF waveform, xx.
%      L = filter length
%     fs = sampling freq

freqs = [697,770,852,941,1209,1336,1477,1633];  % list of centre frequencies

hh = dtmfdesign( freqs,L,fs );
%   hh = MATRIX of all the filters. Each column contains the impulse
%        response of one BPF (bandpass filter)

dtmf.keys = ...
['1','2','3','A';
'4','5','6','B';
'7','8','9','C';
'*','0','#','D'];
dtmf.colTones = [1209,1336,1477,1633];
dtmf.rowTones = [697;770;852;941];

[nstart,nstop] = dtmfcut(xx,fs);   %<--Find the start and end points of each
 tone

%%%% add your lines below to complete the code

% a culmination of sorts - Stephen's code begins here
% objective: return list of keys pressed in order
% length of nstart is number of distinct waveforms, so I'll use that to
% check how many keys there are

% keys is a list of strings
keys = strings;

for i=1:1:length(nstart)
    % sample in question is xx(nstart(i):nstop(i))
    % check row by cycling through row tones
    toneRow = 1;
    for j=1:1:length(dtmf.rowTones)
        isRow = dtmfscore(xx(nstart(i):nstop(i)), hh(:, j));
        if isRow == 1
            toneRow = j;
        end
    end
    % repeat for columns
    toneCol = 1;
    for k=1:1:length(dtmf.colTones)
        isCol = dtmfscore(xx(nstart(i):nstop(i)), hh(:, k+4));
        if isCol == 1
            toneCol = k;
        end
    end
```

```
    % we have row and column, so find key now and add to list keys
    keys(i) = dtmf.keys(toneRow, toneCol);
end
```

Published with MATLAB® R2023a