SOLUTION 1. For brevity, the discussion bellow follows MATLAB's syntax. The corresponding Python commands are very similar (please see the provided code on Moodle).

1. Let N be the length of the vector `s` containing the samples. The sampling time is `Ts = 1/fs`. The command to generate the vector `t` is

   ```
   t = linspace(0, (N-1)*Ts, N);
   ```

   Equivalently, we can do:

   ```
   t = (0:Ts:(N-1)*Ts);
   ```

2. From the discussion in the class, we know that the points of the DFT are spaced $1/T_p$ Hz apart, where $T_p = NT_s$. For computational efficiency, we choose N to be an integer power of 2. If `s_f=fft(s,N)`, then the components of `s_f` are associated to the frequencies

   $$\underbrace{0, \frac{1}{T_p}, \ldots, \frac{1}{2T_s} - \frac{1}{T_p}}_{\texttt{N/2 components}}, \underbrace{-\frac{1}{2T_s}, -\frac{1}{2T_s} + \frac{1}{T_p}, \ldots, -\frac{1}{T_p}}_{\texttt{N/2 components}}.$$

   If we plot `fftshift(s_f)` then the components are associated to the frequencies $-\frac{1}{2T_s}, -\frac{1}{2T_s} + \frac{1}{T_p}, \ldots, -\frac{1}{T_p}, 0, \frac{1}{T_p}, \ldots, \frac{1}{2T_s} - \frac{1}{T_p}$. Thus one possibility to create the vector `f` is

   ```
   f = linspace(-1/(2*Ts), 1/(2*Ts)-1/Tp, N);
   ```

   Another possibility is

   ```
   f = (-1/(2*Ts):1/Tp:1/(2*Ts)-1/Tp);
   ```

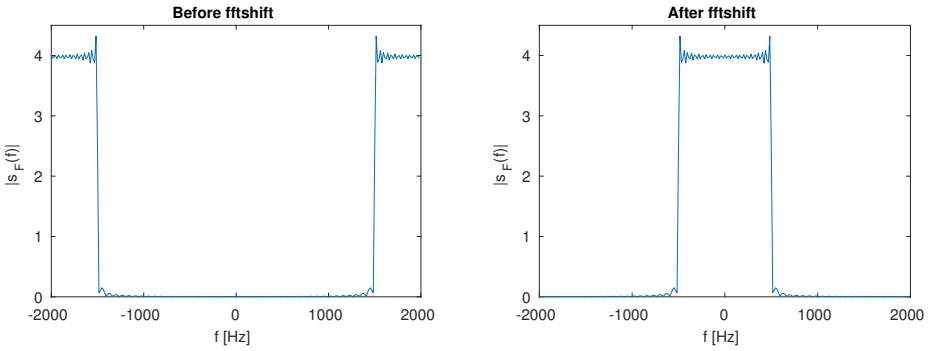

Figure 1: Effect of the `fftshift` command.

3. *Note.* Since `title` is the name of a built-in MATLAB function, you cannot use it as the name of a function argument. In the following code, we called the argument `plottitle` instead.

   ```
   function sol_tfplot(s, fs, name, plottitle)
   % SOL_TFPLOT Time and frequency plot
   %    SOL_TFPLOT(S, FS, NAME, TITLE) displays a figure window with two
   %    subplots.  Above, the signal S is plotted in time domain; below,
   %    the signal is plotted in frequency domain. NAME is the "name" of the
   %    signal, e.g., if NAME is 's', then the labels on the y-axes will be
   %    's(t)' and '|s_F(f)|', respectively.  TITLE is the title that will
   %    appear above the two plots.
   ```

```
% Note: Since TITLE is the name of a built-in Matlab function, you cannot
% use it as the name of a function argument. In the following code, we
% called the argument PLOTTITLE instead.

if ~isscalar(fs)
    error('Fs must be scalar');
end

% Compute the time and frequency axes
Ts = 1/fs;
t = linspace(0, (length(s)-1)*Ts, length(s));

NFFT = 2^nextpow2(length(s)); % for computational efficiency
Tp = NFFT*Ts;
f = linspace(-1/(2*Ts), 1/(2*Ts)-1/Tp, NFFT);

% Compute the FFT
s_f = fft(s,NFFT);
% Correct the scaling and the frequency axis
% Use fftshift to move the negative frequencies to the left
s_f = Ts * fftshift(s_f);

figure;
% First plot: time
subplot(2,1,1); plot(t, s);
xlabel('t [s]'); ylabel(sprintf('%s(t)', name));
title(plottitle);

% Second plot: frequency
subplot(2,1,2); plot(f, abs(s_f));
xlabel('f [Hz]'); ylabel(sprintf('|%s_F(f)|', name));
```

SOLUTION 2. Please see the provided MATLAB/Python routines, which are self-explanatory. For the AM demodulator we also provide an alternative solution which uses a filter.