```
function [stockMod] = simStock_Rand01(...
                        stockStruct, ...
                        currentDay)

    % This function will simulate the
    % random behavior of a stock price.

    % In this particular function, there
    % will be a 50% chance of the price
    % going either up or down.  The amount
    % by which it goes up or down will be
    % determined by a Gaussian (bell-curve)
    % distribution, causing larger
    % changes to be less likely.

    % The trading volume will be held
    % fixed at its initial value.

    % The high and low prices for the day
    % will also be held fixed in this
    % particular function.

    % The only that is changed in this
    % function is the closing price.

    % All values in the stock data struct
    % that are not changed will be set
    % to "-1" in their corresponding
    % matrices, denoting an empty value.

    % Set parameters by which to determine
    % the new stock price.  These are the
    % limits of the likely percent change
    % that a stock might see in one day of
    % trading.
    percChangeUpperLimit = 0.5;
    percChangeLowerLimit = 0.001;

    % Determine whether the stock price
    % will go up or down.
    if(rand(1,1) > 0.5)
        movDir = 1;
    else
        movDir = -1;
    end

    % Determine how much the stock
    % price will change.  Choose a
    % percentage change based on a
    % random Gaussian distribution and 2
    % limiting thresholds.  Modify
```

```
% The Gaussian distribution so that
% the 0-sigma (mean) value
% corresponds to the lower
% threshold and the 3-sigma
% value corresponds to the
% upper threshold.  Also, only
% Accept positive values from
% the distribution.

% First, check to see that the
% current price of the stock
% is not zero or negative.  If
% it is, then assume the company
% is not in business and return
% the new price as zero.
if(stockStruct.currentPrice <= 0)
    newPrice = 0;
else
    % Get a Gaussian distributed
    % random number.  Mean is at
    % 0 and standard deviation is 1.
    gaussVal = randn(1,1);
    % Shift random value so that the
    % distribution is now centered
    % at the percChangeLowerLimit.
    gaussVal = gaussVal + percChangeLowerLimit;
    % Stretch/condense the distribution
    % so that the 3-sigma value
    % corresponds to the upper percent
    % change limit.  The current 3-sigma
    % value is 3.
    gaussVal = (gaussVal*(0.5/3));
    % Take the absolute value of the
    % Gaussian random number.
    % This will be the percent change
    % of the stock price on the given day.
    % THE PERCENT CHANGE WILL BE A PERCENT
    % OF THE ORIGINAL PRICE!  OTHERWISE
    % THE STOCK PRICE WILL ALWAYS
    % TEND TO ZERO.  The original price
    % is given by "stockStruct.close(1)".
    percChange = abs(gaussVal);
    priceChange = (movDir ...
        * stockStruct.close(1) ...
        * percChange);
    newPrice = stockStruct.currentPrice ...
        + priceChange;
end

% If the new price is less than or
% equal to zero, then assume the
```

```
    % company has gone out of business
    % and the stock is now worthless.
    if(newPrice <= 0)
        newPrice = 0;
    end

    % Update the stock data.
    newDataIndex = (length(stockStruct.year) + 1);
    stockStruct.currentPrice = newPrice;
    stockStruct.high(newDataIndex) = -1;
    stockStruct.low(newDataIndex) = -1;
    stockStruct.close(newDataIndex) = newPrice;
    stockStruct.volume(newDataIndex) = -1;

    % Return updated stock struct.
    stockMod = stockStruct;

    return;

end
```