ENGR 451 - Chapter 2 Laboratory

Matlab tutorial

```
clear
x = sequence([1 2 3 4 5], 1);
y = sequence([5 3 1 -1 3 -2 2 3], -1);

% test plus
test_lab1('plus(x, y)')
test_lab1('plus(y, x)')
test_lab1('plus(1, x)')
test_lab1('plus(x, 1)')

y = sequence([5 3 1 0 3 -2 2 3], -4);
test_lab1('plus(x, y)')
test_lab1('plus(y, x)')

% test minustract
test_lab1('minus(x, y)')
test_lab1('minus(y, x)')
test_lab1('minus(1, x)')
test_lab1('minus(x, 1)')

% test timesiplication
test_lab1('times(x, y)')
test_lab1('times(3, x)')
test_lab1('times(x, 3)')

% test flip
test_lab1('flip(x)')

% test shift
test_lab1('shift(y, 2)')

%combinations
test_lab1('flip(minus(shift(plus(x, 2), 4), y))')
test_lab1('plus(flip(plus(x, y)), shift(y, -5))')
test_lab1('minus(plus(times(shift(flip(x), 4), shift(y, 3)), flip(y)),
 x)')

% test stem
set(clf, 'Position', [200 200 400 200])
stem(flip(2+(x-shift(y, -4).*y-3)))
title('y[n]');

% Program Listings
fprintf('\n\n')
disp('--- sequence.m -------------------------------')
type sequence
```

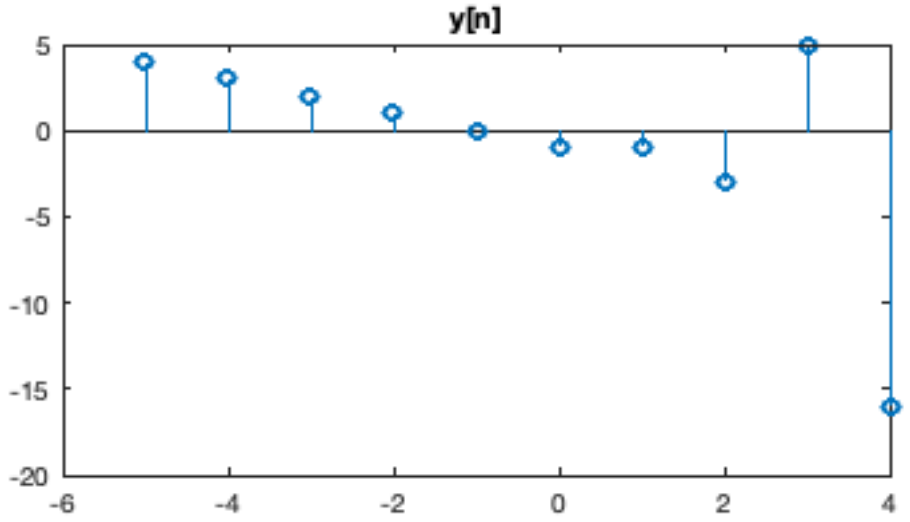
```
plus(x, y): sequence O.K.
```

```
plus(y, x): sequence O.K.
plus(1, x): sequence O.K.
plus(x, 1): sequence O.K.
plus(x, y): sequence O.K.
plus(y, x): sequence O.K.
minus(x, y): sequence O.K.
minus(y, x): sequence O.K.
minus(1, x): sequence O.K.
minus(x, 1): sequence O.K.
times(x, y): sequence O.K.
times(3, x): sequence O.K.
times(x, 3): sequence O.K.
flip(x): sequence O.K.
shift(y, 2): sequence O.K.
flip(minus(shift(plus(x, 2), 4), y)): sequence O.K.
plus(flip(plus(x, y)), shift(y, -5)): sequence O.K.
minus(plus(times(shift(flip(x), 4), shift(y, 3)), flip(y)), x):
 sequence O.K.


--- sequence.m ------------------------------

classdef sequence
 properties
  data
  offset
 end

 methods
  function s = sequence(data, offset)
   % SEQUENCE   Sequence object
   %            S = SEQUENCE(DATA, OFFSET) creates sequence S
   %            using DATA and OFFSET
   %
   %            Kevin Baltazar Reyes  13 Feb 2019
   s.data = data;
   s.offset = offset;
  end

  function display(s)
   var = inputname(1);
   if (isempty(var))
    disp('ans =');
   else
    disp([var '=']);
   end
   switch length(s.data)
    case 0
     disp('    data: []')
    case 1
     disp(['    data: ', num2str(s.data)])
    otherwise
     disp(['    data: [' num2str(s.data) ']'])
   end
```

```
  disp(['  offset: ' num2str(s.offset)])
 end

 function y = flip(x)
  % FLIP Flip a Matlab sequence structure, x, so y = x[-n]
           tempData = x.data(end:-1:1);    %start with the end of the
sequence then count down 1 each time

           tempOffest = -(x.offset+length(x.data) - 1);
           y = sequence(tempData,tempOffest);
 end

 function y = shift(x, n0)
  % SHIFT Shift a Matlab sequence structure, x, by integer amount n0
so that y[n] = x[n - n0]

           sameDataX=x.data;  %data sequence remains the same, we are
only shifting the offset
           newOffset=(x.offset+n0);   %new offset = previous offset +
value you are shifting
           y=sequence(sameDataX,newOffset);
       end

       function x = trim(x)
           %takes zeros off from each side of sequence

           while x.data(end)==0
               x.data(end)=[];
           end

           while x.data(1)==0
               x.data(1)=[];
               x.offset=x.offset+1;
           end

       end


 function z = plus(x, y)
  % PLUS  Add x and y. Either x and y will both be sequence
structures, or one of them may be a number.

           if isa(x,'sequence') == 0     %checks if x is a constant
               z=sequence(y.data+x,y.offset);  %if x is a constant,
add constant x to every data pt in y sequence, leave offest untouched
               z=trim(z);
               return;
           end

           if isa(y,'sequence') == 0    %same as above but instead

               z=sequence(x.data+y,x.offset);
               z=trim(z);
               return;
```

```
            end

            lx=length(x.data);  %length of data in sequence x
            ly=length(y.data);  %length of data in sequence y

            ody=y.offset-x.offset;  %difference between sequence
offsets IF Y HAS GREATER OFFSET THAN X
            odx=x.offset-y.offset;  %%difference between sequence
offsets IF X HAS GREATER OFFSET THAN Y

            x.data=[zeros(1,odx) x.data zeros(1,ody-(lx-ly))];  %add
zeros to the beginning & end of sequence x as a "filler". You cannot
perform operations between x & y if there is no data at a given
index. The zeros are put in to fill these empty spots.
            y.data=[zeros(1,ody) y.data zeros(1,odx-(ly-lx))];

            off=min(x.offset,y.offset);     %minimum offset between x
& y
            z=sequence(x.data+y.data,off);  %create sequence z as a
result of adding sequence x & y together

            z=trim(z);

 end

 function z = minus(x, y)
  % MINUS Subtract x and y. Either x and y will both be sequence
structures, or one of them may be a number.

            if isa(x,'sequence')==0
                z=sequence(x-y.data,y.offset);
                z=trim(z);
                return;
            end
            if isa(y,'sequence')==0
                z=sequence(x.data-y,x.offset);
                z=trim(z);
                return;
            end

            Lx=length(x.data);
            Ly=length(y.data);

            ody=y.offset-x.offset;
            odx=x.offset-y.offset;

            x.data=[zeros(1,odx) x.data zeros(1,ody-(Lx-Ly))];
            y.data=[zeros(1,ody) y.data zeros(1,odx-(Ly-Lx))];

            off=min(x.offset,y.offset);
            z=sequence(x.data-y.data,off);

            z=trim(z);
```

```
  end

  function z = times(x, y)
   % TIMES Multiply x and y (i.e. .*) Either x and y will both be
 sequence structures, or one of them may be a number.
            if isa(x,'sequence')==0
                z=sequence(y.data*x,y.offset);
                return;
            end
            if isa(y,'sequence')==0
                z=sequence(x.data*y,x.offset);
                return;
            end

            Lx=length(x.data);
            Ly=length(y.data);

            ody=y.offset-x.offset;
            odx=x.offset-y.offset;

            x.data=[zeros(1,odx) x.data zeros(1,ody-(Lx-Ly))];
            y.data=[zeros(1,ody) y.data zeros(1,odx-(Ly-Lx))];

            off=min(x.offset,y.offset);
            z=sequence(x.data.*y.data,off);

            z=trim(z);

  end

  function stem(x)
   % STEM Display a Matlab sequence, x, using a stem plot.
              stem( x.offset : length(x.data )+x.offset-1,x.data);
  end
 end
end
```



Published with MATLAB® R2018b