
```

% ENGR 451 - Signals and Systems Laboratory

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

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

% test time
test_lab1('times(x, y)')
test_lab1('times(2, x)')
test_lab1('times(x, -2)')

% 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, -1))')
test_lab1('minus(plus(times(shift(flip(x), 3), shift(y, 2)), flip(y)), x)')

% test stem
set(gcf, 'Position', [200 200 400 200])
stem(flip(1+(x-shift(y, 2).*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(2, x): sequence O.K.
times(x, -2): 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, -1)): sequence O.K.
minus(plus(times(shift(flip(x), 3), shift(y, 2)), flip(y)), x): sequence O.K.

```

```

--- sequence.m -----

```

```

classdef sequence
    properties
        data
        offset
    end

    methods
        function s = sequence(dat, off)
            % SEQUENCE Sequence object
            % S = SEQUENCE(DATA, OFFSET) creates sequence S
            % using DATA and OFFSET
            %
            % Samson Huang 17 Feb 2023
            % Jared Richardson 17 Feb 2023
            % Nadim Damlaaj 17 Feb 2023
            s.data = dat;
            s.offset = off;
        end

        function y = flip(x)
            % FLIP Flip a Matlab sequence structure x so y = x[-n]
            dat = x.data;
            off = x.offset;
            len = length(dat)-1;

            % Flipping array and changing the offset
            flipoff = (off + len) * -1;
            fliparr = x.data(end:-1:1);
            y = sequence(fliparr, flipoff);
        end

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

```

```

    y = sequence(dat, off+n0);
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') && isa(y, 'double')) % only x is a sequence
        % do something;
        zdat = x.data + y;
        z = sequence(zdat, x.offset);
    elseif (isa(y, 'sequence') && isa(x, 'double')) % only y is a sequence
        % do something else
        zdat = y.data + x;
        z = sequence(zdat, y.offset);
    else % both x and y are sequences
        % do something
        xdat = x.data;
        ydat = y.data;
        xoff = x.offset;
        yoff = y.offset;
        xlen = length(xdat);
        ylen = length(ydat);

        % Get the array of whole size, min and max of arr
        temparr1 = [1:xlen] + xoff;
        minarr1 = min(temparr1);
        maxarr1 = max(temparr1);
        temparr2 = [1:ylen] + yoff;
        minarr2 = min(temparr2);
        maxarr2 = max(temparr2);
        fullsize = min(min(temparr1), min(temparr2)) : max(max(temparr1),
max(temparr2));

        % Convert into new array with the new size
        newx = zeros(1,length(fullsize));
        newy = zeros(1,length(fullsize));

        % Finding indeces and replacing the values
        newx((fullsize >= minarr1) & (fullsize <= maxarr1)) = xdat;
        newy((fullsize >= minarr2) & (fullsize <= maxarr2)) = ydat;

        zdat = newx + newy;
        zoff = min([xoff yoff]);
        z = sequence(zdat, zoff);
    end
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') && isa(y, 'double'))
        zdat = x.data - y;
        z = sequence(zdat, x.offset);
    end
end

```

```

elseif (isa(y, 'sequence') && isa(x, 'double'))
    zdat = x - y.data;
    z = sequence(zdat, y.offset);
else
    xdat = x.data;
    ydat = y.data;
    xoff = x.offset;
    yoff = y.offset;
    xlen = length(xdat);
    ylen = length(ydat);

    % Get the array of whole size, min and max of arr
    temparr1 = [1:xlen] + xoff;
    minarr1 = min(temparr1);
    maxarr1 = max(temparr1);
    temparr2 = [1:ylen] + yoff;
    minarr2 = min(temparr2);
    maxarr2 = max(temparr2);
    fullsize = min(min(temparr1), min(temparr2)) : max(max(temparr1),
max(temparr2));

    % Convert into new array with the new size
    newx = zeros(1,length(fullsize));
    newy = zeros(1,length(fullsize));

    % Finding indeces and replacing the values
    newx((fullsize >= minarr1) & (fullsize <= maxarr1)) = xdat;
    newy((fullsize >= minarr2) & (fullsize <= maxarr2)) = ydat;
    zdat = newx - newy;

    zoff = min([xoff yoff]);
    z = sequence(zdat, zoff);
end
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') && isa(y, 'double'))
        zdat = x.data .* y;
        z = sequence(zdat, x.offset);
    elseif (isa(y, 'sequence') && isa(x, 'double'))
        zdat = y.data .* x;
        z = sequence(zdat, y.offset);
    else
        xdat = x.data;
        ydat = y.data;
        xoff = x.offset;
        yoff = y.offset;
        xlen = length(xdat);
        ylen = length(ydat);

        % Get the array of whole size, min and max of arr
        temparr1 = [1:xlen] + xoff;

```

```

    minarr1 = min(temparr1);
    maxarr1 = max(temparr1);
    temparr2 = [1:ylen] + yoff;
    minarr2 = min(temparr2);
    maxarr2 = max(temparr2);
    fullsize = min(min(temparr1), min(temparr2)) : max(max(temparr1),
max(temparr2));

    % Convert into new array with the new size
    newx = zeros(1,length(fullsize));
    newy = zeros(1,length(fullsize));

    % Finding indeces and replacing the values
    newx((fullsize >= minarr1) & (fullsize <= maxarr1)) = xdat;
    newy((fullsize >= minarr2) & (fullsize <= maxarr2)) = ydat;

    zdat = newx .* newy;
    zoff = min([xoff yoff]);
    z = sequence(zdat, zoff);
end
end

function stem(x)
% STEM Display a Matlab sequence x using a stem plot.
dat = x.data;
off = x.offset;

len = length(dat);
offArr = [0:len-1]+off;

stem(offArr, dat);
end
end
end

```

