Topics: iteration using for, plotting, random number

Reading (ML): Sec 4.2–4.5 (exclude 4.2.2), Sec 2.10, 2.11

Polynomial, revisited

Write a program fragment to evaluate an \( n \)th order polynomial of \( x \):

\[
a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n
\]

A given vector \( p \) has length \( n + 1 \) and contains the coefficients of the polynomial where \( p(1) \) is the coefficient for the term \( x^0 \). Your program should request a user input value for variable \( x \) and then evaluate the polynomial. Use a while loop. Do not use MATLAB predefined functions other than length and input.

General form of the for Loop

```
for index = expression
  Statements to execute
  Also called loop body
end
```

Expression usually takes the form of a vector. E.g., \( 1:\text{m} \).

Two patterns for doing something \( n \) times

```
for i = 1:n
  \% do something
  \% ...
end

i = 1;
while i <= n
  \% do something
  \% ...
  i = i + 1;
end
```

Polynomial, re-revisited

Consider the program fragment for evaluating the polynomial. Can you replace the while loop with a for loop?

Plotting

It is very easy to make plots using MATLAB. An x-y plot can be generated using the built-in function plot. The command

```
plot(a,b,'-', c,d, '*')
```

will generate a plot with two graphs, one showing the data contained in vectors \( a \) (in x-direction) and \( b \) (in y-direction) as a line and the other showing the data in vectors \( c \) and \( d \) as asterisks. Refer to Sec 2.11 in the Chapman text for the many formatting options. If you omit the formatting option ('-' and '*' above), the default on most system is to show the data as a line.
A random example. I mean a random walk example!

Write a program that performs a “random walk.” In a random walk, possible moves are left, right, up, or down (in a Cartesian plane). Prompt the user for the number of steps and the starting point.

% Perform n steps of random walk starting from (x0,y0)

disp('Do a random walk!')
n = input('How many steps? ');
x0 = input('From what x-coordinate? ');
y0 = input('From what y-coordinate? ');

% possible movements: ( xdir(i), ydir(i) )
xdir = [ 1 -1 0 0 ];
ydir = [ 0 0 1 -1 ];

x = [x0 zeros(1,n)]; % trajectory in x direction
y = [y0 zeros(1,n)]; % trajectory in y direction

% Perform walk, each step is based on a random integer

% Show the walk, starting point, ending point
plot(x,y,x(1),y(1),'*',x(end),y(end),'o')
title([num2str(n) ' steps of random walk from * to o'])

Building vectors through concatenation

% Add vectors a and b
n = length(a);
c = zeros(1,n); % unnecessary statement but improves performance
k = 1; while ( k <= n)
c(k) = a(k) + b(k);
k = k + 1;
end