Previous Lecture:
- Finite vs. infinite; discrete vs. continuous
- `plot`, `fill`
- Intro to 1-d array and vectorized code

Today’s Lecture:
- 1-d array—vector
- Probability and random numbers
- Simulation using random numbers, vectors

Announcements:
- Project 3 due Friday 3/1 at 11pm
- Discussion this week in Upson B7 lab
- Prelim 1 on Mar 7th at 7:30pm. Email Randy Hess (rbh27) now if you have an exam conflict (specify conflicting course and instructor contact info)

Example
- Write a program fragment that calculates the cumulative sums of a given vector `v`.
- The cumulative sums should be stored in a vector of the same length as `v`.

```
1, 3, 5, 0   v
1, 4, 9, 9   cumulative sums of v
```

Example
- Write a function `evalPoly` to evaluate an \( n \) th order polynomial of \( x \):
  \[
a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n
\]
- Input parameter `coef` has length \( n+1 \), contains the coefficients of the polynomial
- `coef(1)` is the coefficient for the term \( x^0 \)
- Input parameter `x`
- Return the value of the polynomial evaluated at `x`
- No Matlab predefined function other than `length`
Random numbers

- **Pseudorandom** numbers in programming
- Function `rand(...)` generates random real numbers in the interval (0,1). All numbers in the interval (0,1) are equally likely to occur—uniform probability distribution.

Examples:
- `rand(1)` one random # in (0,1)
- `6*rand(1)` one random # in (0,6)
- `6*rand(1)+1` one random # in (1,7)

Simulate a fair 6-sided die

Which expression(s) below will give a random integer in [1..6] with equal likelihood?

A. `round(rand*6)`
B. `ceil(rand*6)`
C. Both expressions above

Possible outcomes from rolling a fair 6-sided die

Keep tally on repeated rolls of a fair die

Repeat the following:

```matlab
% roll the die
% increment correct "bin"
```
function count = rollDie(rolls)
FACES= 6; % # faces on die
count= zeros(1,FACES); % bins to store counts
% Count outcomes of rolling a FAIR die
for k= 1:rolls
  % Roll the die
  face= ceil(rand*FACES);
  % Increment the appropriate bin
  count(face)= count(face) + 1;
end
% Show histogram of outcome

% Count outcomes of rolling a FAIR die
count= zeros(1,6);
for k= 1:100
  face= ceil(rand*6);
  if face==1
    count(1)= count(1) + 1;
  elseif face==2
    count(2)= count(2) + 1;
    % increment count(2)
  elseif face==5
    count(5)= count(5) + 1;
  else
    count(6)= count(6) + 1;
  end
end

function [x, y] = RandomWalk2D(N)
% 2D random walk in 2N-1 by 2N-1 grid.
% Walk randomly from (0,0) to an edge.
% Vectors x, y represent the path.

function [x, y] = RandomWalk2D(N)
k=0; xc=0; yc=0;
while not at an edge
  % Choose random dir, update xc, yc
  % Record new location in x, y
end
function \([x, y] = \text{RandomWalk2D}(N)\)

\[
\begin{align*}
k &= 0; \\
x_c &= 0; \\
y_c &= 0;
\end{align*}
\]

\[
\text{while } |x_c| < N \text{ and } |y_c| < N \text{ do}
\begin{align*}
% \text{Choose random dir, update } x_c, y_c \\
% \text{Record new location in } x, y
\end{align*}
\]

\[
\text{end}
\]

% Standing at \((x_c, y_c)\)  
% Randomly select a step
\[
\begin{align*}
r &= \text{rand}(1); \\
\text{if } r < 0.25 & \quad \text{then } y_c = y_c + 1; \quad \% \text{north} \\
\text{elseif } r < 0.5 & \quad \text{then } x_c = x_c + 1; \quad \% \text{east} \\
\text{elseif } r < 0.75 & \quad \text{then } y_c = y_c - 1; \quad \% \text{south} \\
\text{else} & \quad \text{then } x_c = x_c - 1; \quad \% \text{west}
\end{align*}
\]

Another representation for the random step

- Observe that each update has the form
  \[
  \begin{align*}
  x_c &= x_c + \Delta x \\
  y_c &= y_c + \Delta y
  \end{align*}
  \]
  no matter which direction is taken.
- So let’s get rid of the if statement!
- Need to create two “change vectors” \(\Delta x\) and \(\Delta y\)

\[
\begin{align*}
\Delta x &= \boxed{\ldots} \\
\Delta y &= \boxed{\ldots}
\end{align*}
\]

RandomWalk2D_v2.m