Previous Lecture:
- Nesting if-statements
- Logical operators short-circuit
- Top-down design

Today's Lecture:
- Iteration using for
- Watch MatTV episode "Troubleshooting for-loops"

Announcements:
- Discussion this week in the classrooms as listed in Student Center
- CS1112 "Partner Search Mixer" on Thursday, 9/8, 5:30-6:30pm, Gates Hall atrium in front of G01, sponsored by WICC, ACSU, URMC, and the CS Dept
- Last call to register your clickers—use the link on the course website

Question
A stick of unit length is split into two pieces. The breakpoint is randomly selected. On average, how long is the shorter piece?

Physical experiment? •
Thought experiment? → analysis
Computational experiment! → simulation•

*Need to repeat many trials!

Simulation:
use code to imitate the physical experiment

% one trial of the experiment
breakPt= rand;
if  breakPt<0.5
    shortPiece= breakPt;
else
    shortPiece= 1-breakPt;
end

Want to do many trials, add up the lengths of the short pieces, and then divide by the number of trials to get the average length.

Repeat n times

% one trial of the experiment
breakPt= rand;
shortPiece= min(breakPt, 1-breakPt);

Take average
Print result
n = 10000;  % number of trials
total = 0;  % accumulated length so far
for k = 1:n
    % one trial of the experiment
    breakPt = rand;
    shortPiece = min(breakPt, 1-breakPt);
    total = total + shortPiece;
end
aveLength = total/n;
fprintf('Average length is %f
', aveLength)

Example: "Accumulate" a solution

% Average 10 numbers from user input
n = 10;  % number of data values
for k = 1:n
    % read and process input value
    num = input('Enter a number: ');
    total = total + num;
end
ave = total/n;  % average of n numbers
fprintf('Average is %f
', ave)

How many passes through the loop will be completed?

A: 0
B: 1
C: 9
D: 10
E: 11

Important Features of Iteration
- A task can be accomplished if some steps are repeated; these steps form the loop body
- Need a starting point
- Need to know when to stop
- Need to keep track of (and measure) progress—update

Monte Carlo Approximation of \( \pi \)

For each of \( N \) trials
- Throw a dart
- If it lands in circle
  - add 1 to total # of hits

\( \pi \) is \( 4 \times \text{hits}/N \)

Monte Carlo \( \pi \) with \( N \) darts on \( L \times L \) board

\( N=\_\_\; \)
for \( k = 1: N \)
    % Throw kth dart
    % Count it if it is in the circle
end
myPi = 4*hits/N;
Syntax of the for loop

\[
\text{for } \text{<var>=<start value>:<incr>:<end bound>}
\]

\[
\text{statements to be executed repeatedly}
\]

\[
\text{end}
\]

Loop header specifies all the values that the index variable will take on, one for each pass of the loop.
E.g. \text{k=3:1:7} means \text{k} will take on the values 3, 4, 5, 6, 7, one at a time.

Pattern for doing something \text{n} times

\[
\text{n=_____}
\]

\[
\text{for k=1:n}
\]

\[
\text{\% code to do}
\]

\[
\text{\% that something}
\]

\[
\text{end}
\]

\text{Definite iteration}

\text{for loop examples}

\[
\text{for k=2:0.5:3}
\]

\[
\text{disp(k)}
\]

\text{k takes on the values \ldots}
\text{Non-integer increment is OK}

\[
\text{for k=1:4}
\]

\[
\text{disp(k)}
\]

\text{k takes on the values \ldots}
\text{Default increment is 1}

\[
\text{for k=0:-2:-6}
\]

\[
\text{disp(k)}
\]

\text{k takes on the values \ldots}
\text{“Increment” may be negative}

\[
\text{for k=0:-2:-7}
\]

\[
\text{disp(k)}
\]

\text{k takes on the values \ldots}
\text{Colon expression specifies bounds}

\[
\text{for k=5:2:1}
\]

\[
\text{disp(k)}
\]

\text{end}

\text{end}

\text{end}

% What will be printed?
\text{for k=1:2:6}
\text{fprintf(‘%d ’, k)}
\text{end}

\text{\% code to do}
\text{\% that something}
\text{end}

A: 123456
B: 1356
C: 135
D: error (incorrect bounds)

% What will be printed?
\text{for k=10:-1:14}
\text{fprintf(‘%d ’, k)}
\text{end}
\text{fprintf(‘!’)}

A: error (incorrect bounds)
B: 10 (then error)
C: 10!
D: 14!
E: !

What will be displayed when you run the following script?

\text{for k = 4:6}
\text{disp(k)}
\text{k=9;}
\text{disp(k)}
\text{end}

A: 4
B: 4
C: 9
D: 14
E: 1

Something else …