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:
- Project 1 due tonight at 11pm; late submission accepted until Friday 11pm sharp with 10% penalty
- Due to Feb Break, attendance at discussion next week is optional but you are responsible for the contents of the exercise to be posted. Attend any Wednesday sections (10:10-4:25) if you like. See Syllabus for locations.
An algorithm is an idea. To use an algorithm you must choose a programming language and implement the algorithm.
An algorithm is an idea. To use an algorithm you must choose a programming language and implement the algorithm.
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!
A stick of unit length is split into two pieces. The breakpoint is randomly selected. On average, how long is the shorter piece?

A: .000001
B: .25
C: .333333
D: .499999
E: none of the above
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
```
% one trial of the experiment
breakPt = rand;
shortPiece = min(breakPt, 1-breakPt);

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\n’, aveLength)
Example: “Accumulate” a solution

```matlab
% 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
Remember to initialize

% Average 10 numbers from user input

n= 10; % number of data values
total= 0; % current sum (initialized to zero)
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\n', ave)
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
Example: “Accumulate” a solution

% Average 10 numbers from user input

n= 10;       % number of data values
total= 0;    % current sum (initialized to zero)
for k= 1:1: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\n', ave)
Monte Carlo Approximation of $\pi$

Throw $N$ darts

Sq. area = $N = L \times L$

Circle area = $N_{in} = \pi L^2 / 4$
Monte Carlo Approximation of $\pi$

Throw $N$ darts

Sq. area = $N = L \times L$

Circle area = $N_{in}$

$$= \pi \frac{L^2}{4}$$

$$\pi = 4 \frac{N_{in}}{N}$$
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 $L$-by-$L$ board

\[ N = \_ \_ \_ ; \]
\[ \text{for } k = 1 : N \]
\[ \text{end} \]
\[ \text{myPi} = 4 \times \text{hits} / N ; \]
Monte Carlo $\pi$ with $N$ darts on L-by-L board

\begin{verbatim}
N=__; for k = 1:N
    % Throw kth dart
    % Count it if it is in the circle
end
myPi = 4*hits/N;
\end{verbatim}
Monte Carlo $\pi$ with $N$ darts on $L$-by-$L$ board

\begin{verbatim}
N=__;
for k = 1:N
    % Throw kth dart
    x = rand*L - L/2;
    y = rand*L - L/2;
    % Count it if it is in the circle
    if sqrt(x^2+y^2) <= L/2
        hits = hits + 1;
    end
end
myPi = 4*hits/N;
\end{verbatim}
Monte Carlo $\pi$ with $N$ darts on $L$-by-$L$ board

\[ N=__; L=__; \text{ hits}= 0; \]

\begin{verbatim}
for k = 1:N
    \% Throw kth dart
    x = rand*L - L/2;
y = rand*L - L/2;
    \% Count it if it is in the circle
    if sqrt(x^2+y^2) <= L/2
        hits = hits + 1;
    end
end
\end{verbatim}

\[ \text{myPi} = 4*\text{hits}/N; \]
Syntax of the \texttt{for} loop

\begin{verbatim}
for <var>= <start value>:<incr>:<end bound>
    \textit{statements to be executed repeatedly}
end
\end{verbatim}

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