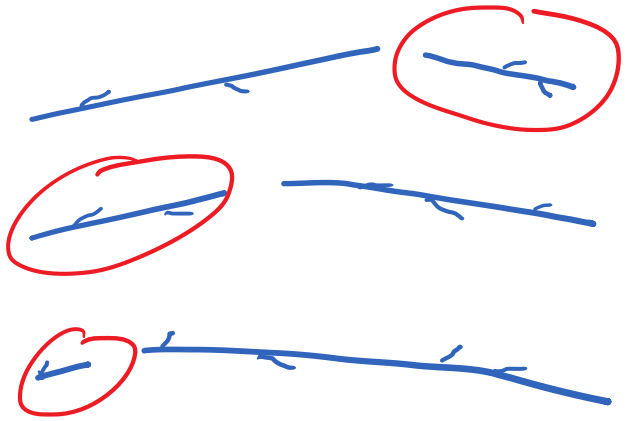
Lecture 5: Definite iteration

- Previous lecture:
  - Logical operators (`&&`, `||`, `~`) and "short-circuiting"
  - Nested `if`-statements
  - Top-down design
- Today:
  - Iteration using `for`
  - (at home) Watch MatTV episode "Troubleshooting for-loops"
- Announcements:
  - P1 due tonight, 11pm EST
    - Late submissions accepted tomorrow with 5% penalty
  - Read *Insight* §2.2 (or MatTV episode on while-loop) and *Insight* §3.2 before next lecture

Question

A 1 meter-long stick is split into two pieces. The breakpoint is randomly selected. On average, how long is the shorter piece?



Thought experiment? → analysis

Physical experiment?

Computational experiment! → simulation

} Need to repeat many trials!

Question

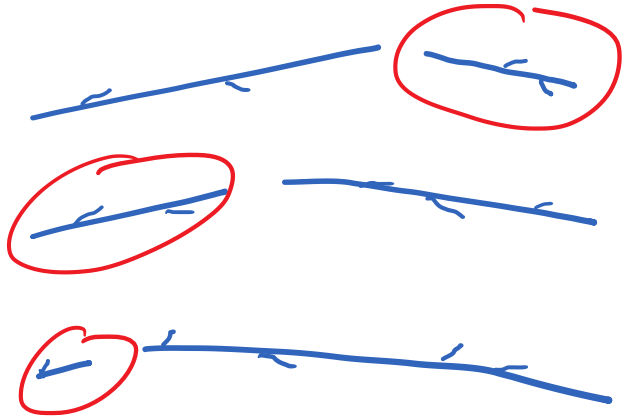
A 1 meter-long stick is split into two pieces. The breakpoint is randomly selected (equally likely anywhere along the stick). On average, how long is the *shorter* piece?

A: ¼ m

B: 1/3 m

C: ½ m

D: other



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
```

More shortcuts: `min()`

```
% 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.

badStickExp.m

Algorithm (bottom-up development)

Repeat many times:

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

Take average

Print result

initialization

loop

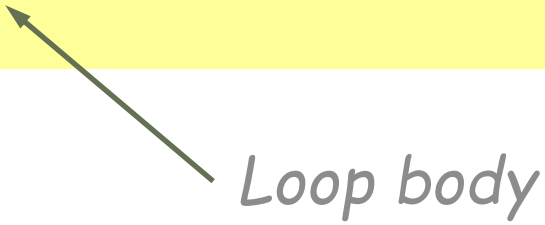
loop body

```
n= 10000;   % number of trials
total= 0;   % accumulated length so far

for k = 1:1:n   % Repeat many times
    % one trial of the experiment
    breakPt= rand();
    shortPiece= min(breakPt, 1-breakPt);
    total= total + shortPiece;
end

avgLength= total/n;   % Take average
fprintf('Average length is %f\n', ...
        avgLength)   % Print result
```

See `stickExp.m` , `showForLoop.m`

Syntax of the `for` loop

```
for <var>= <start value>:<incr>:<end bound>
        statements to be executed repeatedly
end
```



Loop header specifies all the values that the index variable will take on, one for each pass of the loop.

E.g, `k= 3:1:7` means `k` will take on the values 3, 4, 5, 6, 7, one at a time.

`for` loop examples

```
for k = 2:0.5:3
    disp(k)
end
```

`k` takes on the values 2, 2.5, 3

Non-integer increment is OK

```
for k = 1:4
    disp(k)
end
```

`k` takes on the values 1, 2, 3, 4

Default increment is 1

```
for k = 0:-2:-6
    disp(k)
end
```

`k` takes on the values 0, -2, -4, -6

"Increment" may be negative

```
for k = 0:-2:-7
    disp(k)
end
```

`k` takes on the values 0, -2, -4, -6

Colon expression specifies *bounds*

```
for k = 5:2:1
    disp(k)
end
```

The set of values for k is the empty set: the loop body won't execute

Pattern for doing something *n* times

```
n= ______
for k= 1:n


    % code to do
    % that something


end
```

Definite iteration

Accumulation Pattern

```
% 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
avg= total/n;  % average of n numbers
fprintf('Average is %f\n', avg)
```

Accumulator variable

Example: “Accumulate” a solution

```
% Average 10 numbers from user input
clear       % clear workspace
n= 10;      % number of data values

for k = 1:n
  % read and process input value
    num= input('Enter a number: ');
    total= total + num;
end
avg= total/n;  % average of n numbers
fprintf('Average is %f\n', avg)
```

How many passes through the loop will be completed?

A: 0

B: 1

C: 9

D: 10

E: 11

Remember to initialize

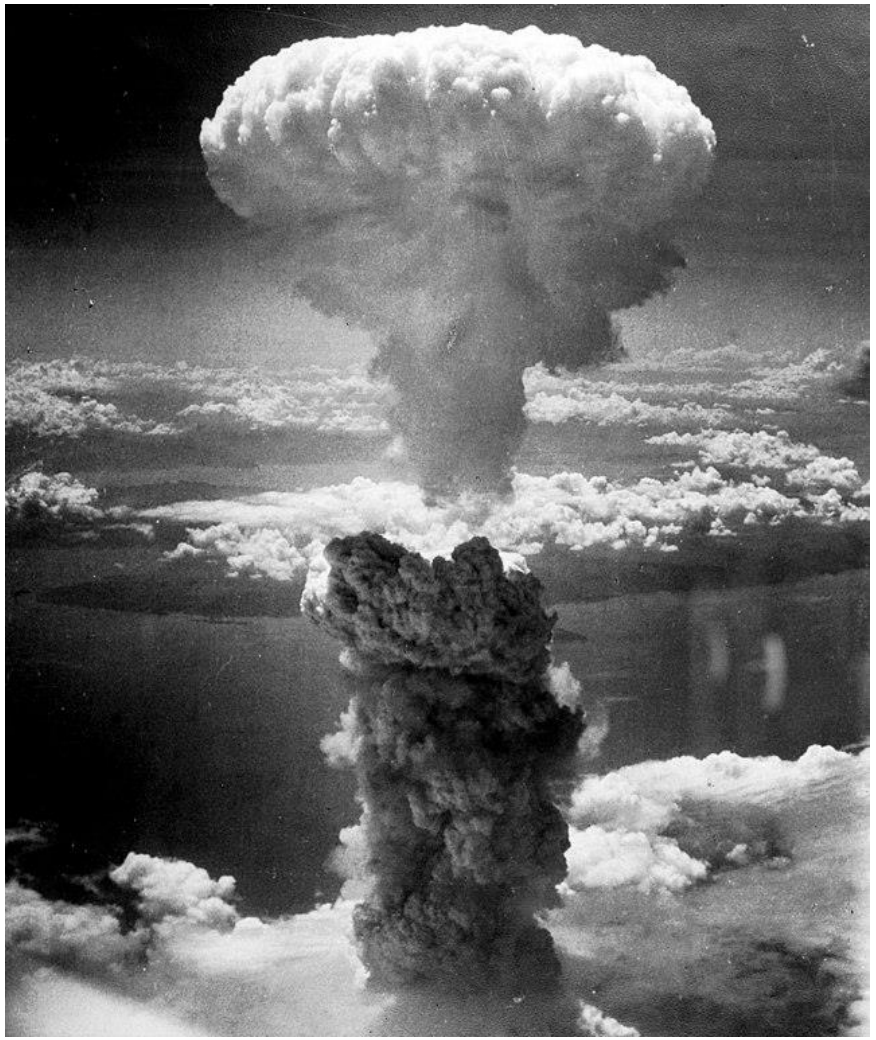
```
% Average 10 numbers from user input


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
avg= total/n;  % average of n numbers
fprintf('Average is %f\n', avg)
```

Important Features of Iteration

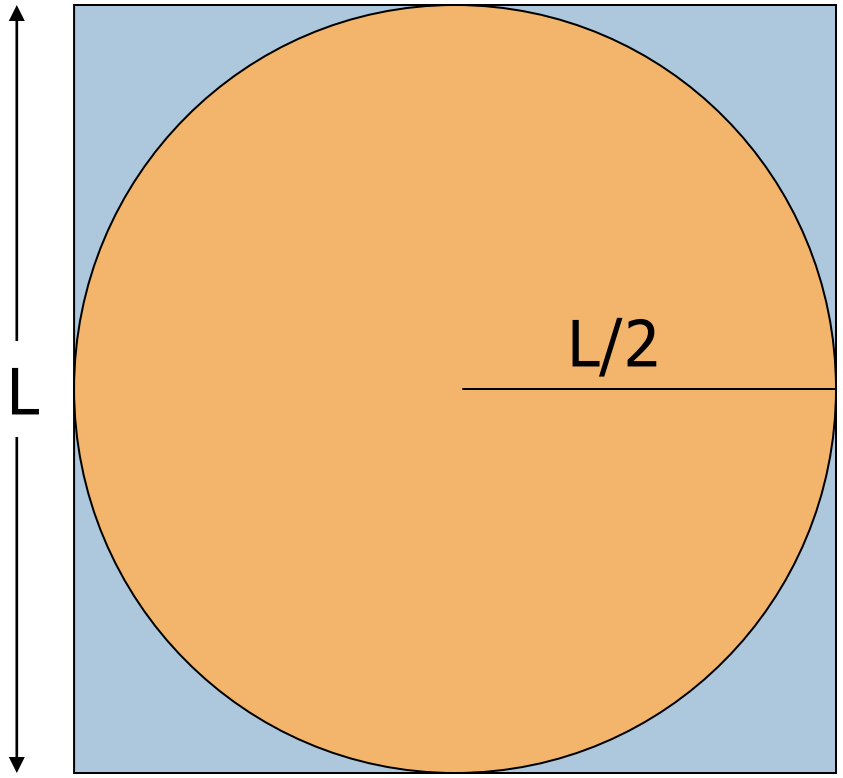
- A task can be accomplished if some steps are repeated; these steps form the loop body
- Need a starting point
- Need to know when to stop
- Need to keep track of (and measure) progress—update

Monte Carlo methods

1. Derive a relationship between some *desired quantity* and a *probability*
2. Use simulation to estimate the probability
   - Computer-generated random numbers
3. Approximate desired quantity based on prob. estimate



Monte Carlo Approximation of π



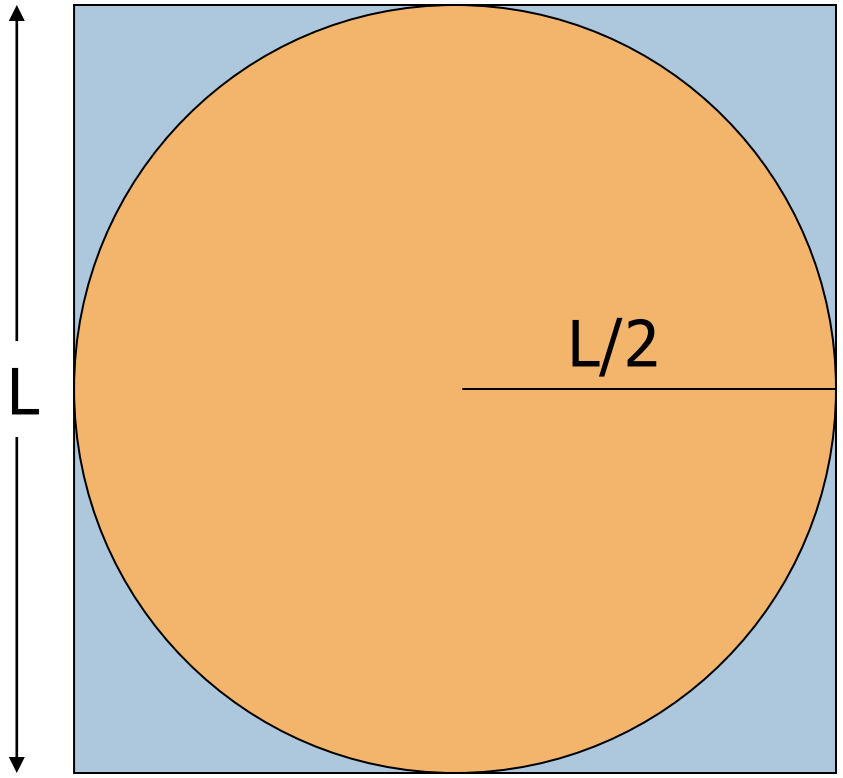
Throw N darts

Sq. area = $L \times L$

Circle area = $\pi L^2/4$

Prob. landing in circle
= (circle area)/(sq. area)
= $\pi/4$
≅ N_{in}/N

Monte Carlo Approximation of π



Throw N darts

$$\pi \cong 4\, N_{in} / N$$

Monte Carlo Approximation of π

For each of N trials

Throw a dart

If it lands in circle

add 1 to total # of hits

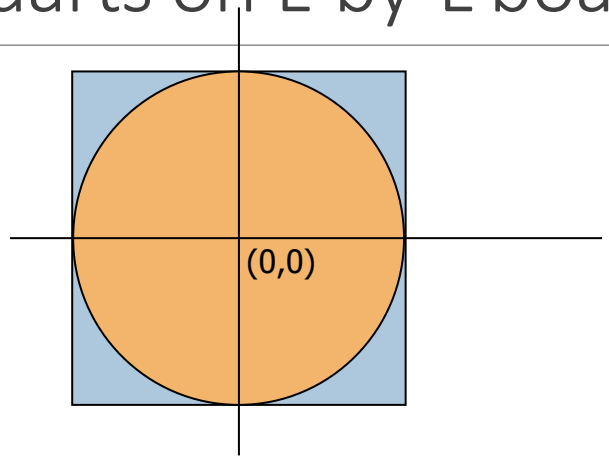
Pi is 4*hits/N

Monte Carlo Approximation of π with N darts on L-by-L board

```
N=__;
for k = 1:N




end
myPi= 4*hits/N;
```

Monte Carlo Approximation of π with N darts on L-by-L board

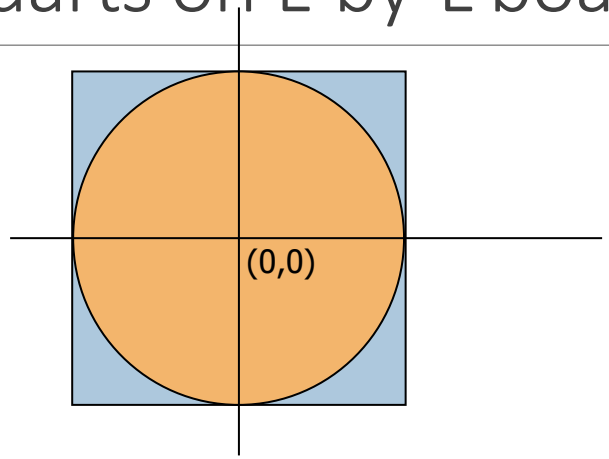
```
N=__;
for k = 1:N
    % Throw kth dart



    % Count it if it is in the circle



end
myPi = 4*hits/N;
```



See **mcPi.m**

Monte Carlo Approximation of π with N darts on L-by-L board

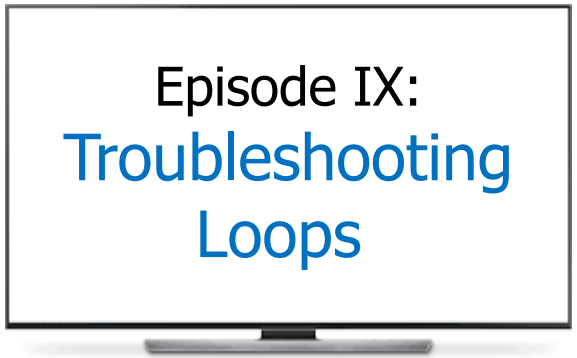
```
N=__; L=__; hits= ???
for k = 1:N
    % Throw kth dart
    x= rand()*L - L/2;
    y= rand()*L - L/2;
    % Count it if it is in the circle
    if sqrt(x^2 + y^2) <= L/2
        hits= hits + 1;
    end
end
myPi= 4*hits/N;
```



What will be displayed when you run the following script?

```
for k = 4:6
    disp(k)
    k= 9;
    disp(k)
end
```

Watch MatTV to find out!



Wrap-up review

```
% What will be printed?

for k= 1:2:6
    fprintf('%d ', k)
end
printf('\n')
```

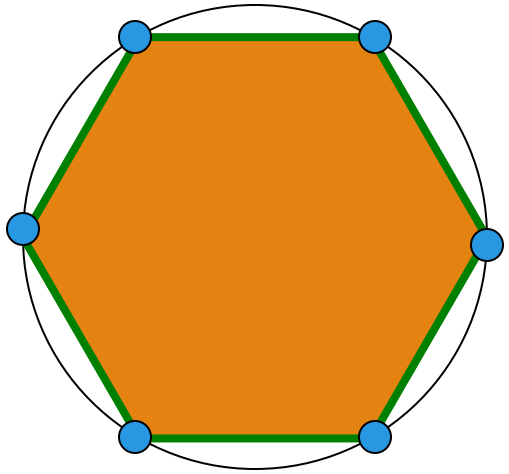
A: 1 2 3 4 5 6

B: 1 3 5 6

C: 1 3 5

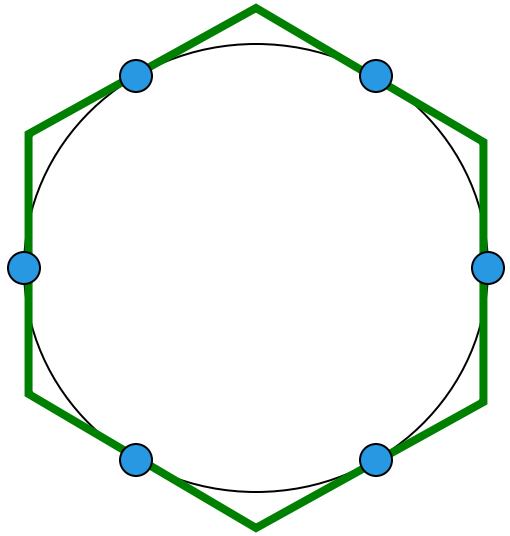
D: *error (incorrect bounds)*

Example: *n*-gon → circle



Inscribed hexagon

$(n/2)\sin(2\pi/n)$



Circumscribed hexagon

$n\tan(\pi/n)$

As *n* approaches infinity, the inscribed and circumscribed areas approach the area of a circle.

When will |OuterA – InnerA| <= .000001?