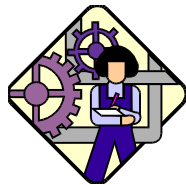


Building your own Functions



CS112 Scientific Computation
Department of Computer Science
Wellesley College

Built-in MATLAB functions

math: `sum, prod, mean, std, abs, sqrt, sin, cos,`
`abs, exp, min, max`

logical: `any, all, and, or, not`

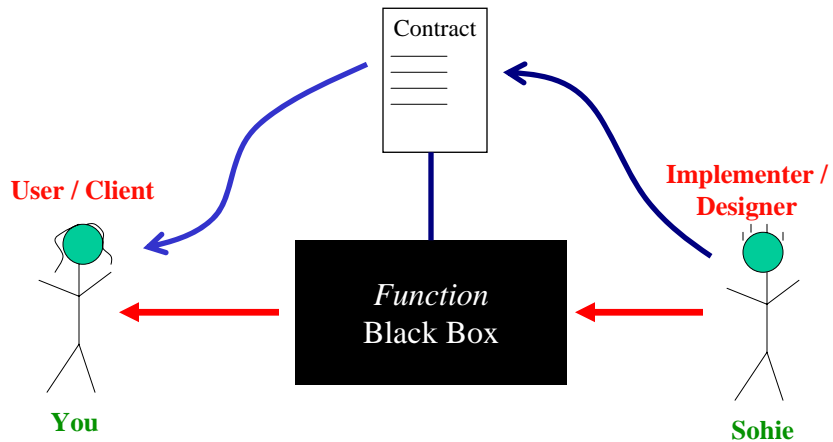
creation: `linspace, colon, ones, zeros`

dimensions: `size, length`

graph/display: `plot, figure, subplot, xlabel, ylabel,`
`title, axis, legend, imshow, imtool`

input/output: `input, disp`

Big idea: Abstraction

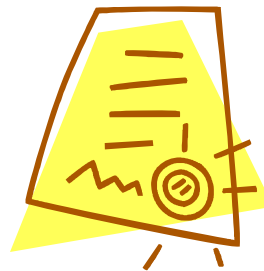


Functions 8-3

The other side of the contract

We've been using functions built by others - let's try writing a few of our own

Implement a **myMean** function that returns a single value representing the average value of a *vector or matrix*



Functions 8-4

Rules of the road

```
function avg = myMean (data)
% avg = myMean(data)
% returns the average of all of the values
% in data, which may be a vector or matrix
dims = size(data);
if (min(dims) == 1)
    avg = sum(data)/length(data);
else
    avg = sum(sum(data))/prod(dims);
end
```

Annotations:

- keyword → **function**
- output parameter → **avg**
- function name (stored in file myMean.m) → **myMean**
- input parameter → **(data)**
- local variable* → **dims**
- assign → **avg = sum(data)/length(data);**
- output → **else**
- assign → **avg = sum(sum(data))/prod(dims);**
- end → **end**
- help comments contract → **% avg = myMean(data)**
% returns the average of all of the values
% in data, which may be a vector or matrix
- function body → **dims = size(data);**
if (min(dims) == 1)
avg = sum(data)/length(data);
else
avg = sum(sum(data))/prod(dims);
end

* Local variables only exist during the execution of the function

Functions 8-5

Calling the new myMean function

```
>> nums = [3 9 6 2 8];
>> meanVal = myMean(nums)
meanVal =
    5.6000
>> nums = [3 4 7; 2 8 6]
nums =
     3     4     7
     2     8     6
>> meanVal = myMean(nums)
meanVal =
    5.0000
```



Functions 8-6

Executing a function

MATLAB workspace

Execution land

```
>>
```

Functions 8-7

Create `nums` vector

MATLAB workspace

`nums`

| | | | | |
|---|---|---|---|---|
| 3 | 9 | 6 | 2 | 8 |
|---|---|---|---|---|

Execution land

```
>> nums = [3 9 6 2 8];
```

Functions 8-8

Invoke myMean function

MATLAB workspace

nums 3 9 6 2 8

myMean local workspace

Execution land

```
>> nums = [3 9 6 2 8];  
>> meanVal = myMean(nums)
```

```
function avg = myMean(data)  
    dims = size(data);  
    if (min(dims) == 1)  
        avg = sum(data)/length(data);  
    else  
        avg = sum(sum(data))/prod(dims);  
    end
```

Functions 8-9

Create variable for input parameter data and copy value of nums to data

MATLAB workspace

nums 3 9 6 2 8

myMean local workspace

data 3 9 6 2 8

Execution land

```
>> nums = [3 9 6 2 8];  
>> meanVal = myMean(nums)
```

```
function avg = myMean(data)  
    dims = size(data);  
    if (min(dims) == 1)  
        avg = sum(data)/length(data);  
    else  
        avg = sum(sum(data))/prod(dims);  
    end
```

Functions 8-10

Execute function body

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MATLAB workspace nums <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> | 3 | 9 | 6 | 2 | 8 | myMean local workspace data <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> dims <table border="1"><tr><td>1</td><td>5</td></tr></table> | 3 | 9 | 6 | 2 | 8 | 1 | 5 |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | |
| 1 | 5 | | | | | | | | | | | | |
| Execution land » nums = [3 9 6 2 8]; » meanVal = myMean(nums) | <pre>function avg = myMean(data) dims = size(data); if (min(dims) == 1) avg = sum(data)/length(data); else avg = sum(sum(data))/prod(dims); end</pre> | | | | | | | | | | | | |

Functions 8-11

Is min(dims) == 1?

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MATLAB workspace nums <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> | 3 | 9 | 6 | 2 | 8 | myMean local workspace data <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> dims <table border="1"><tr><td>1</td><td>5</td></tr></table> | 3 | 9 | 6 | 2 | 8 | 1 | 5 |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | |
| 1 | 5 | | | | | | | | | | | | |
| Execution land » nums = [3 9 6 2 8]; » meanVal = myMean(nums) | <pre>function avg = myMean(data) dims = size(data); if (min(dims) == 1) avg = sum(data)/length(data); else avg = sum(sum(data))/prod(dims); end</pre> | | | | | | | | | | | | |

Functions 8-12

Yes, so we do 'then' clause

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|------|
| MATLAB workspace nums <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> | 3 | 9 | 6 | 2 | 8 | myMean local workspace data <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> dims <table border="1"><tr><td>1</td><td>5</td></tr></table> avg <table border="1"><tr><td>5.60</td></tr></table> | 3 | 9 | 6 | 2 | 8 | 1 | 5 | 5.60 |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | | |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | | |
| 1 | 5 | | | | | | | | | | | | | |
| 5.60 | | | | | | | | | | | | | | |

| | |
|--|---|
| Execution land <pre>>> nums = [3 9 6 2 8]; >> meanVal = myMean(nums)</pre> | <pre>function avg = myMean(data) dims = size(data); if (min(dims) == 1) avg = sum(data)/length(data); else avg = sum(sum(data))/prod(dims); end</pre> |
|--|---|

Functions 8-13

Return value stored in output variable

| | | | | | | | | | | | | | | | |
|--|---|---|---|---|---|--|---|---|---|---|---|---|---|---|------|
| MATLAB workspace nums <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> meanVal <table border="1"><tr><td> </td></tr></table> | 3 | 9 | 6 | 2 | 8 | | myMean local workspace data <table border="1"><tr><td>3</td><td>9</td><td>6</td><td>2</td><td>8</td></tr></table> dims <table border="1"><tr><td>1</td><td>5</td></tr></table> avg <table border="1"><tr><td>5.60</td></tr></table> | 3 | 9 | 6 | 2 | 8 | 1 | 5 | 5.60 |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| 3 | 9 | 6 | 2 | 8 | | | | | | | | | | | |
| 1 | 5 | | | | | | | | | | | | | | |
| 5.60 | | | | | | | | | | | | | | | |

| | |
|--|---|
| Execution land <pre>>> nums = [3 9 6 2 8]; >> meanVal = myMean(nums)</pre> | <pre>function avg = myMean(data) dims = size(data); if (min(dims) == 1) avg = sum(data)/length(data); else avg = sum(sum(data))/prod(dims); end</pre> |
|--|---|

Functions 8-14

And the local workspace goes away

MATLAB workspace

```
nums [3 9 6 2 8]
meanVal 5.60
```

Execution land

```
>> nums = [3 9 6 2 8];
>> meanVal = myMean(nums)
meanVal =
    5.6000
```

Functions 8-15

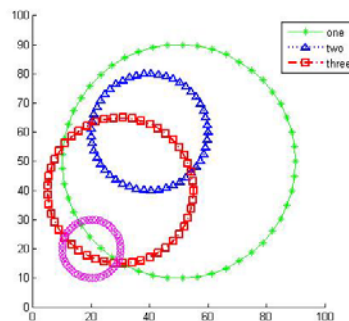
Try another one?

Write a function to draw a circle

Think first about the contract:

use inputs to control appearance:
radius, location, color, markers,
line style, line width

Call the new function `drawCircle`
and store it in an M-File named
`drawCircle.m`



Functions 8-16

Getting started

```
function drawCircle (radius, xcenter, ycenter, properties, width)
% drawCircle(radius, xcenter, ycenter, properties, width)
% draws circle with the specified radius, centered on location
% (xcenter, ycenter) with the specified properties and width
```

← don't forget
contract
comments!

Functions 8-17

Filling in the function body

```
function drawCircle (radius, xcenter, ycenter, properties, width)
% drawCircle(radius, xcenter, ycenter, properties, width)
% draws circle with the specified radius, centered on location
% (xcenter, ycenter) with the specified properties and width
```

```
angles = linspace(0, 2*pi, 50);
xcoords = xcenter + radius * cos(angles);
ycoords = ycenter + radius * sin(angles);
plot(xcoords, ycoords, properties, 'LineWidth', width);
```

Functions 8-18

Executing drawCircle function

MATLAB workspace

Execution land

```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

Functions 8-19

Create input parameter variables ...

MATLAB workspace

Execution land

```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

```
function drawCircle (radius, xcenter, ycenter, ...  
                    properties, width)  
    angles = linspace(0, 2*pi, 50);  
    xcoords = xcenter + radius * cos(angles);  
    ycoords = ycenter + radius * sin(angles);  
    plot(xcoords, ycoords, properties, ...  
         'LineWidth', width);
```

drawCircle local
workspace

radius

xcenter

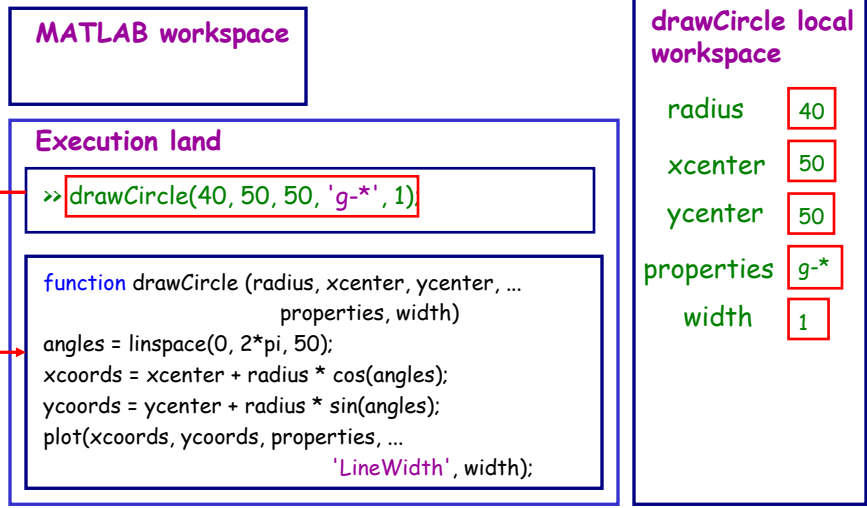
ycenter

properties

width

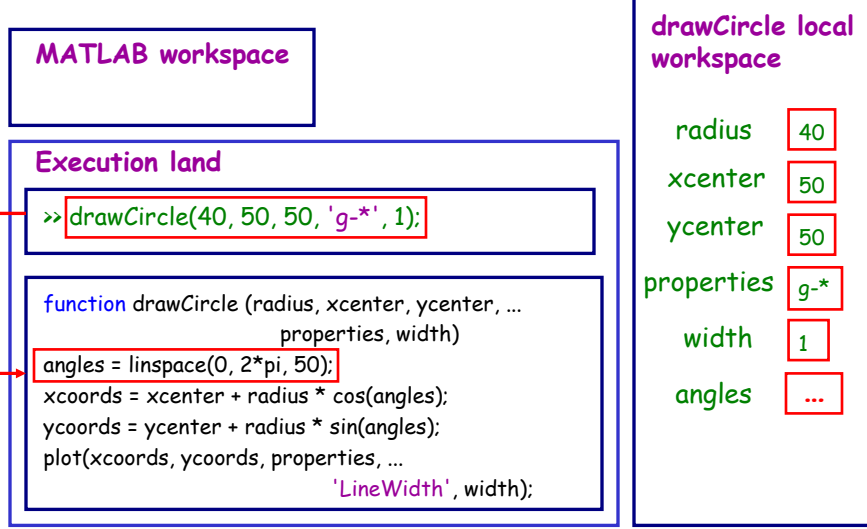
Functions 8-20

... and fill them in from call statement



Functions 8-21

Execute body of function



Functions 8-22

Next statement

MATLAB workspace

Execution land

```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

```
function drawCircle (radius, xcenter, ycenter, ...  
                    properties, width)  
    angles = linspace(0, 2*pi, 50);  
    xcoords = xcenter + radius * cos(angles);  
    ycoords = ycenter + radius * sin(angles);  
    plot(xcoords, ycoords, properties, ...  
         'LineWidth', width);
```

drawCircle local workspace

radius 40

xcenter 50

ycenter 5

properties g-*

width 1

angles ...

xcoords ...

Functions 8-23

Next statement

MATLAB workspace

Execution land

```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

```
function drawCircle (radius, xcenter, ycenter, ...  
                    properties, width)  
    angles = linspace(0, 2*pi, 50);  
    xcoords = xcenter + radius * cos(angles);  
    ycoords = ycenter + radius * sin(angles);  
    plot(xcoords, ycoords, properties, ...  
         'LineWidth', width);
```

drawCircle local workspace

radius 40

xcenter 50

ycenter 5

properties g-*

width 1

angles ...

xcoords ...

ycoords ...

Functions 8-24

And we draw the circle

MATLAB workspace

Execution land

```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

```
function drawCircle (radius, xcenter, ycenter, ...  
                    properties, width)  
    angles = linspace(0, 2*pi, 50);  
    xcoords = xcenter + radius * cos(angles);  
    ycoords = ycenter + radius * sin(angles);  
    plot(xcoords, ycoords, properties, ...  
         'LineWidth', width);
```

drawCircle local workspace

radius 40

xcenter 50

ycenter 5

properties 'g-*'

width 1

angles ...

xcoords ...

ycoords ...

Functions 8-25

Where'd everybody go?

MATLAB workspace

Execution land

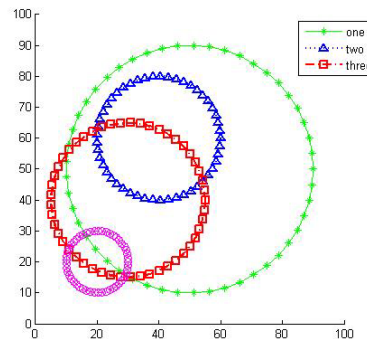
```
>> drawCircle(40, 50, 50, 'g-*', 1);
```

Functions 8-26

A thorough test

```
% testCircle.m  
% tests the drawCircle function
```

```
hold on  
drawCircle(40, 50, 50, 'g-*', 1);  
drawCircle(20, 40, 60, 'b:^', 2);  
drawCircle(25, 30, 40, 'r-s', 2);  
legend('one', 'two', 'three');  
drawCircle(10, 20, 20, 'm--o', 1);  
axis equal  
axis([0 100 0 100])  
hold off
```



Functions 8-27

Functions with multiple outputs

Suppose we'd like to write a variation
of `myMean` that returns both the
arithmetic mean and *geometric mean*

Given n numbers:

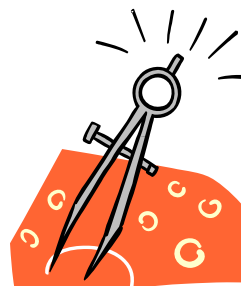
$$a_1, a_2 \dots a_n$$

arithmetic mean:

$$(a_1 + a_2 + \dots + a_n)/n$$

geometric mean:

$$\sqrt[n]{a_1 * a_2 * \dots * a_n}$$



Functions 8-28

Consider first ...

Some built-in functions can return one or more values, depending on how the function is called

For example, consider the `min` function:

```
>> nums = [3 9 6 2 8];
>> minVal = min(nums)
minVal =
     2
>> [minVal minIndex] = min(nums)
minVal =
     2
minIndex =
     4
```

Functions 8-29

New lean mean machine

```
function [arith geom] = myMean2(data)
% [arith geom] = myMean2(data)
% returns both the arithmetic and geometric mean of
% the values in data, which may be a vector or matrix
```

```
dims = size(data);
if (min(dims) == 1)
    arith = sum(data)/length(data);
    geom = nthroot(prod(data), length(data));
else
    arith = sum(sum(data))/prod(dims);
    geom = nthroot(prod(prod(data)), prod(dims));
end
```



Functions 8-30