L03: Scripts and Functions

Building blocks of programming

Lucas A. J. Bastien

E7 Spring 2017, University of California at Berkeley

January 23, 2017

Version: release

Announcements

Do the textbook reading!

- In class, we introduce, explain, and discuss the fundamental concepts. The book contains additional details
- Do the reading before lecture! Come prepared to discuss and practice these fundamental concepts

Correction to the syllabus (already updated on bCourses):

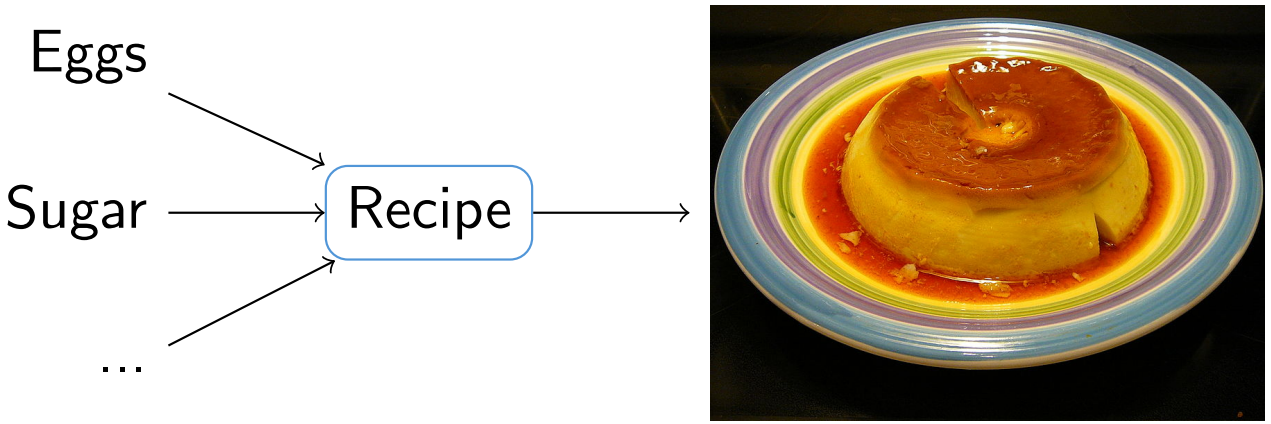
- Drop-in hours in 1109 Etcheverry Hall:
  - Open hours on **Fridays 8 am – 5 pm** (9 am – 12 pm with GSIs)
  - Also open **Monday/Wednesday 3 pm – 4 pm**

Today:

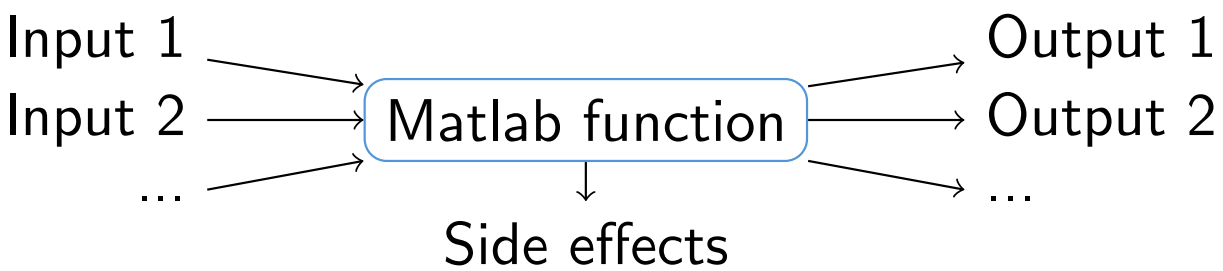
- Scripts and functions: the fundamentals

Introduction to functions

One can use ingredients (**inputs**) and a recipe (*i.e.* a set of **instructions**), to create a flan (**output**).



A Matlab function is a set of **instructions** that can operate on a set of **input** data and that can **output** a set of different data (or the same data). A function can have desired side effects (*e.g.*, write data to disk)



Flan image by Cary Bass, retrieved from https://en.wikipedia.org/wiki/File:Homemade_Flan.jpg on January 23rd 2017

Scripts versus functions

Both scripts and functions are text files that contain sets of Matlab commands that, when executed, perform specific tasks

- **A script can access and modify its caller's Workspace**
- **A function uses a pre-defined interface to communicate data between its own Workspace and the caller's Workspace, via input and output arguments**

What do these (loose) definitions mean? It is what we will be learning and practicing today

Functions will be our fundamental building blocks for programming this semester

Note: today, the caller is always the Command Prompt, but it could be something else, for example another function

Example of script

Assuming we start with an empty Workspace, we type the following commands at the command prompt:

```
>> P = 101325;
>> V = 1;
>> T = 293;
>> my_ideal_gas_law
```

my_ideal_gas_law.m

```
1       % Ideal gas constant (J mol-1 K-1)
2 -     R = 8.314;
3       % Calculate the number of moles
4 -     n = P * V / R / T;
```

Which of the following statements are true?

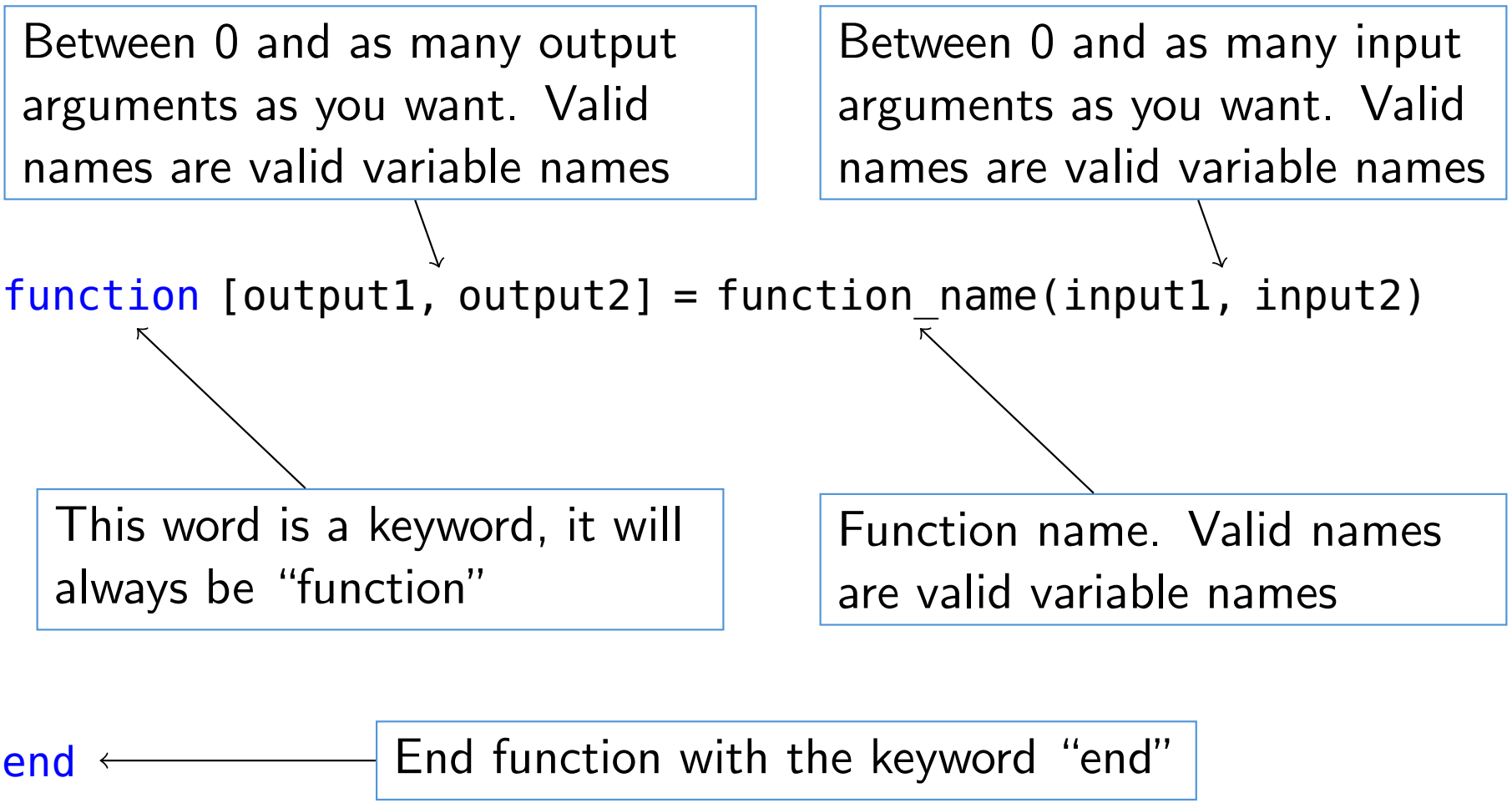
(A) Matlab throws an error because P, V, and T are not defined in the script

(B) There are now 5 variables defined in the Workspace

(C) P, V, and T are now undefined in the Workspace

Function declaration syntax

The first non-blank non-comment line of the function file must follow the syntax:



Examples of functions

my_sphere_1.m

```
function [area] = my_sphere_1(radius)
% Calculates the area of a sphere
area = 4 * pi * radius^2;
end
```

my_sphere_2.m

```
function [area, volume] = my_sphere_2(radius)
% Calculates the area and volume of a sphere

% One can use intermediate variables
r_square = radius^2;
% We must define the output arguments
area = 4 * pi * r_square;
volume = 4/3 * pi * r_square*radius;
end
```

Pass specific data to functions when calling them:

```
>> area = my_sphere_1(2)
area =
   50.2655

>> radius = 5;
>> area = my_sphere_1(radius)
area =
  314.1593

>> r = 10;
>> area = my_sphere_1(r)
area =
   1.2566e+03
```

```
>> [a, v] = my_sphere_2(2)
a =
   50.2655
v =
   33.5103

>> r = 5;
>> [a, v] = my_sphere_2(r)
a =
  314.1593
v =
  523.5988
```

Functions and Workspace: important remarks

You must know and understand these important remarks:

In most cases, functions do not have access to the variables defined in their caller's Workspace. The caller is the place where the function was called from (here: the command prompt)

When called, a function creates and uses its own separate Workspace. This Workspace lives for the duration of the function call

Use input arguments to provide data to the function, these input arguments will be available in the function's Workspace

Use output arguments to retrieve data from the function's Workspace as the function "returns" (*i.e.* finishes its execution)

Functions (Important examples)

Assuming we start with an empty Workspace, we type the following commands at the command prompt:

```
>> a = my_disk_area_1(10);
```

my_disk_area_1.m

```
1   function [area] = my_disk_area_1(radius)
2   % Calculates the surface area of a disk
3   % given its radius
4 -   area = pi * radius^2;
5 -   end
```

Which of the following statements are true?

(A) Matlab throws the error Undefined function or variable 'pi'

(B) Matlab throws the error Undefined function or variable 'radius'

(C) Matlab throws the error Undefined function or variable 'a'

(D) Matlab throws the error Undefined function or variable 'area'

(E) There is now a variable named a in the workspace, with value 314.1593

Functions (Important examples)

Assuming we start with an empty Workspace, we type the following commands at the command prompt:

```
>> radius = 10;
>> a = my_disk_area_2();
```

my_disk_area_2.m

```
1   function [area] = my_disk_area_2()
2   % Calculates the surface area of a disk
3   % given its radius
4 -   area = pi * radius^2;
5 -   end
```

Which of the following statements are true?

(A) Matlab throws the error Undefined function or variable 'radius'

(B) Matlab throws the error Undefined function or variable 'area'

(C) There is now a variable named a in the workspace, with value 314.1593

Note: this function is buggy and does not work: it cannot access the command prompt Workspace. See previous example for correct implementation.

Functions (Important examples)

What is the value of the variable *r* after we execute the following commands at the command prompt:

```
>> r = 10;
>> x = my_weird_function(2);
```

my_weird_function.m

```
1   function [result] = my_weird_function(a)
2   % What am I doing?
3   r  = a;
4   r = r + 10/r;
5   r = 5 - a;
6   result = r;
7   end
```

(A) Matlab throws the error Undefined function or variable 'result'

(B) -2

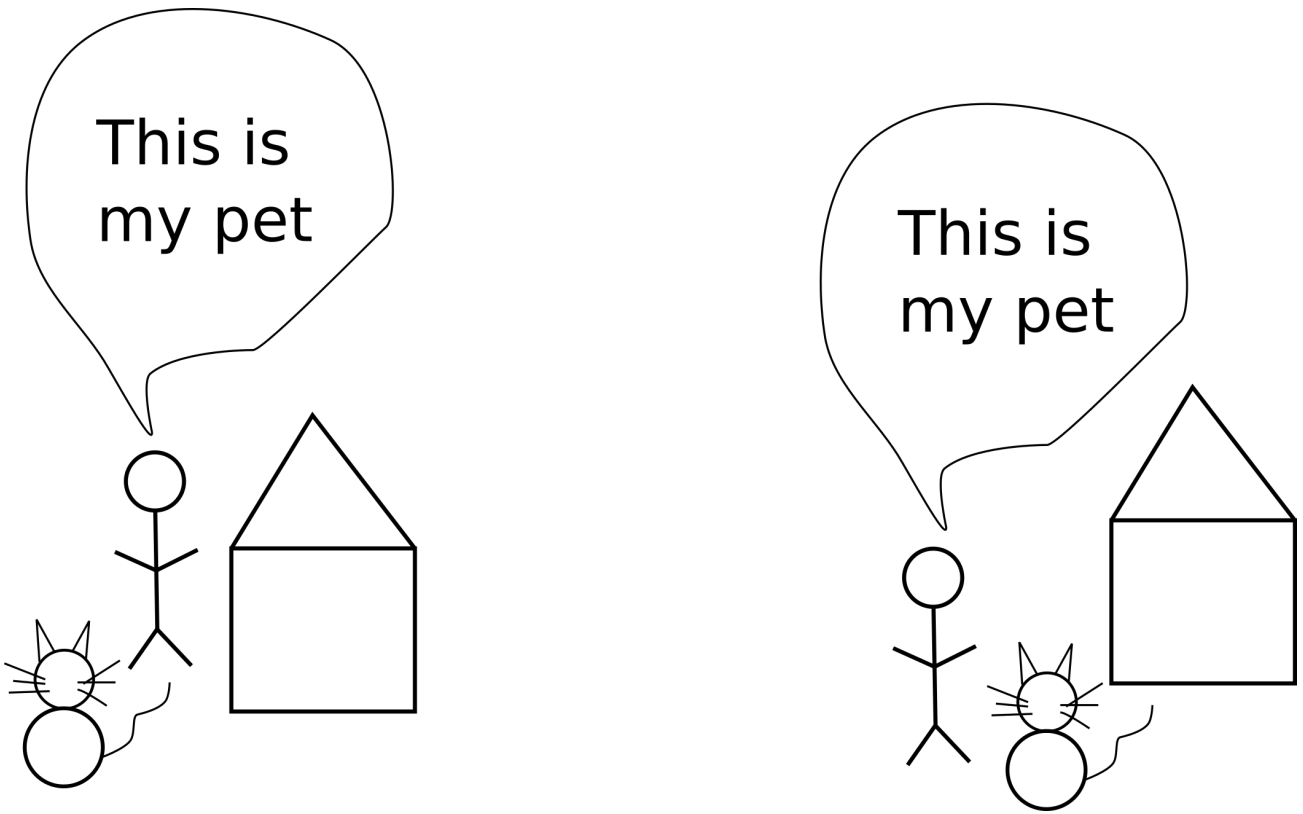
(C) 3

(D) 10

Remember: this function does not have access to the command prompt's Workspace. The variable r it manipulates is different from the variable r we defined before calling the function

A matter of point of view

In the drawing below, both people are referring to their pet as "my pet", yet they are talking about two different animals. In the previous slide, the variable "r" defined in the command prompt's Workspace and the variable "r" used in the function `my_weird_function.m` have the same name, yet they are different objects.



Functions (Important examples)

Assuming that we start with an empty Workspace, what is the value of the variable result after we execute the following commands at the command prompt:

```
>> r = 10;
>> x = my_weird_function(2);
```

my_weird_function.m

```
1   function [result] = my_weird_function(a)
2   % What am I doing?
3   r  = a;
4   r = r + 10/r;
5   r = 5 - a;
6   result = r;
7   end
```

(A) -2

(B) -1

(C) The variable result is not defined

(D) 3

Remember: the function's own Workspace lives only for the duration of the function call, and is separate from the command prompt's Workspace.

Functions (Important examples)

What are the values of the variables x and y after executing the following commands?

```
>> a = 10;
>> b = 20;
>> [x, y] = my_function_ab(b, a)
```

my_function_ab.m

```
1   function [x, y] = my_function_ab(a, b)
2 -   x = a - 2;
3 -   y = b + 2;
4 -   end
```

(A) $x = 18$ and $y = 12$

(B) $x = 8$ and $y = 22$

Again, the variables a and b defined in the command prompt's Workspace are different from the variables a and b used in the function definition. When passing data to a function when calling it, it is the order of the input arguments that matters

Functions (Important examples)

What are the values of the variables x and y after executing the following commands?

```
>> a = 10;
>> b = 20;
>> [y, x] = my_function_ab(a, b)
```

my_function_ab.m

```
1   function [x, y] = my_function_ab(a, b)
2 -   x = a - 2;
3 -   y = b + 2;
4 -   end
```

(A) $x = 8$ and $y = 22$

(B) $x = 22$ and $y = 8$

Again, the variables x and y defined in the command prompt's Workspace are different from the variables x and y used in the function definition. When retrieving a function's output data when calling it, it is the order of the output arguments that matters

Functions (Important examples)

After the following code is executed, which of the following are true?

```
>> x = 2;
>> y = my_square(x+2);
```

my_square.m

```
1    function x_square = my_square(x)
2     % Calculates the square of number x
3 -   x_square = x * x;
4 -   end
```

(A) x == 4

(B) x_square == 16

(C) x == 2

(D) Answers A and B

(E) Answers B and C