Executing a function

Script vs. Function

- A script is executed line-by-line just as if you are typing it into the Command Window
  - The value of a variable in a script is stored in the Command Window Workspace

- A function has its own private (local) function workspace that does not interact with the workspace of other functions or the Command Window workspace

---

What will be printed?

```
% Script file
p= -3;
q= absolute(p);
disp(p)
```

```
function q = absolute(p)
% q is the absolute value of p
if (p<0)
   p= -p;
end
q= p;
```

What will be printed?

```
% Script file
p= -3;
q= absolute(p);
disp(p)
```

```
function q = absolute(p)
% q is the absolute value of p
if (p<0)
   p= -p;
end
q= p;
```

Command Window Workspace

| p | -3 |

Function absolute's Workspace

| p | -3 |

What will be printed?

```
% Script file
p= -3;
q= absolute(p);
disp(p)
```

```
function q = absolute(p)
% q is the absolute value of p
if (p<0)
   p= -p;
end
q= p;
```

Command Window Workspace

| p | -3 |

Function absolute's Workspace
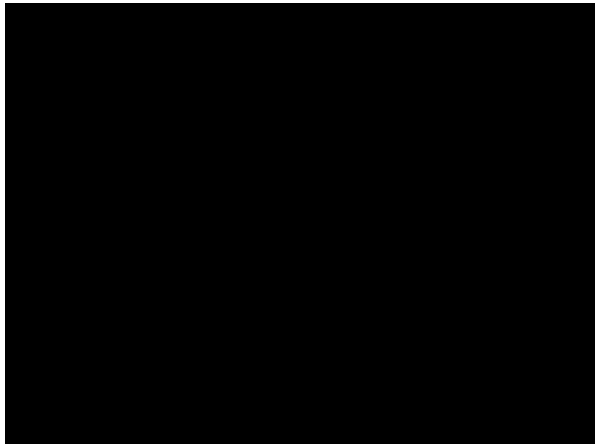
| p | -3 |

What will be printed?

```
% Script file
p= -3;
q= absolute(p);
disp(p)
```

```
function q = absolute(p)
% q is the absolute value of p
if (p<0)
   p= -p;
end
q= p;
```

Command Window Workspace

| p | -3 |

Function absolute's Workspace

| p | 3 |
| q | 3 |

What will be printed?

```
% Script file
p= -3;
q= a  3      ;
disp(p)
```

```
function q = absolute(p)
% q is the absolute value of p
if (p<0)
  p= -p;
end
q= p;
```

Command Window Workspace

p	-3
q	3

Comand window workspace

P [-3]

q [3]

---

function absolute's workspace

P [~~-3~~ 3]

q [3] (crossed out)

* There are 2 p's, one in each work space. Change in p in f space does <u>not</u> affect p in the script's space.

Execute the statement `y= foo(x)`

- Matlab looks for a function called foo (m-file called foo.m)
- Argument (value of x) is copied into function foo's local parameter, in function foo's workspace
  - called "pass-by-value," one of several argument passing schemes used by programming languages
- Function code executes within its own workspace
- At the end, the function's output argument (*value*) is sent from the function to the place that calls the function. E.g., the *value* is assigned to y.
- Function's workspace is deleted
  - If foo is called again, it starts with a new, empty workspace