

CALLING FUNCTIONS

An Introduction to Computer Science



Let's learn about Calling Functions.

What Are Functions?

Functions: Reusable chunks of code



Functions are reusable chunks of code.

Once code is wrapped in a function, we can use it in other places.

First, we'll learn how to use functions.

Later, we'll learn how to make them for ourselves.

Terminology

"Using"

"Calling"

"Invoking"



When you want to "use" a Function, we say that you "call" it.
Another term is "invoking" a function.

Syntax

```
print()
```

Name

Parentheses



There are two essential parts to calling a function:
The name of the function, and then parentheses following the name.
These parentheses are **CRITICAL** to calling the function.

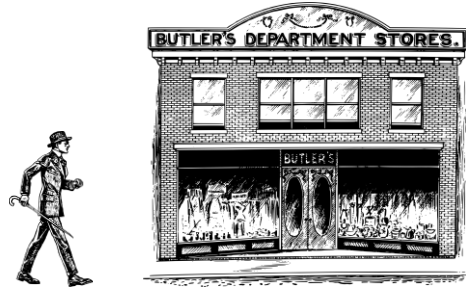
Calling or Not

`go_to_store`



Thinking about going to the store
(without parentheses)

`go_to_store()`

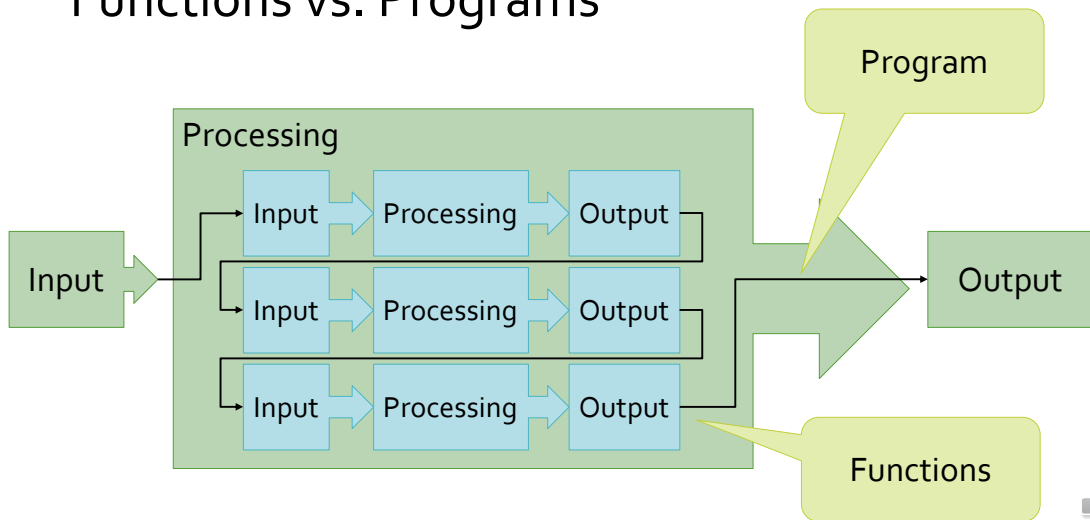


Actually going to the store
(with parentheses)

<https://openclipart.org/user-detail/j4p4n>

To call a function, you **MUST** add parentheses.
Otherwise, you are simply referring to the name of the function rather than executing it.
Think of it as the difference between "thinking about going to the store" as opposed to "actually physically going to the store".

Functions vs. Programs



A program and a function are similar, but not quite the same.

They both have the same "Input-Process-Output" pattern.

However, functions are typically much smaller, and are meant to accomplish a single specific task.

In fact, most industrial programs are actually just a bunch of functions working together!

Functions vs. Methods

```
"MAKE THIS LOWERCASE".lower()
```

lower: A method that makes a string lowercase



Methods are a special kind of function.

They are attached strongly to a value.

The way we write a method call is slightly different too, since it needs to incorporate that value.

Calling Methods

```
"MAKE THIS LOWERCASE".lower()
```

Value or variable

Period

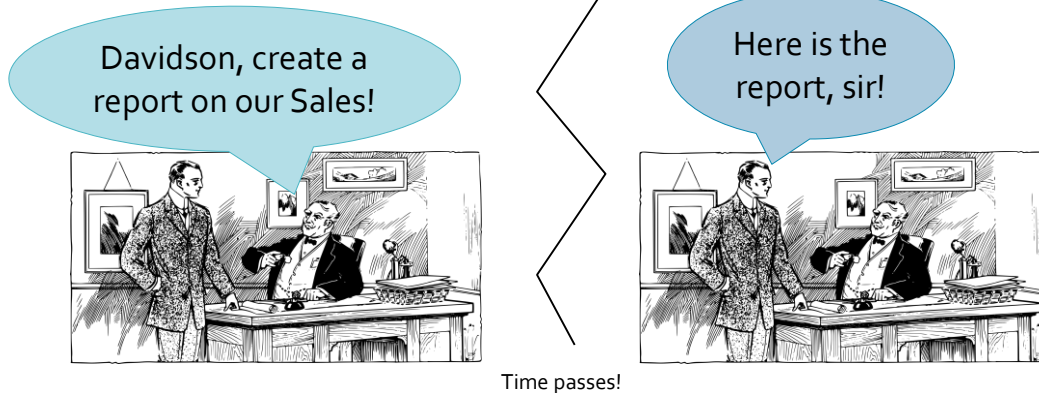


When you want to use a Method, you need the name, parentheses, AND two more pieces: The calling value or variable and a period.

In later lessons, we'll learn more about this strange syntax.

For now, get comfortable with the difference between calling functions and methods.

Metaphor



You don't have to know **how** a function accomplishes its goal, in order to use it. Think of it like a well-run office.

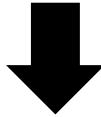
When you ask an employee to complete some work, you don't care how it happens, just the inputs and outputs to their process.

You give them the assignment and tell them what you need, and they make it happen and get you back a report.

This separation of concerns allows you to focus on the big picture of a program without worrying about the minor details.

Returning Values

```
"MAKE THIS LOWERCASE".lower()
```



```
"make this lowercase"
```



Functions are useful because they can return values.

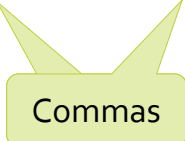
Functions can return any type of value: String, Boolean, Integer, whatever you need.

Mentally, when a function is called, you can imagine the result replacing the entire function call.

In a way, this is similar to what happens when you do addition or subtraction.

Arguments

```
print("First", "Second", "Third")
```



Commas

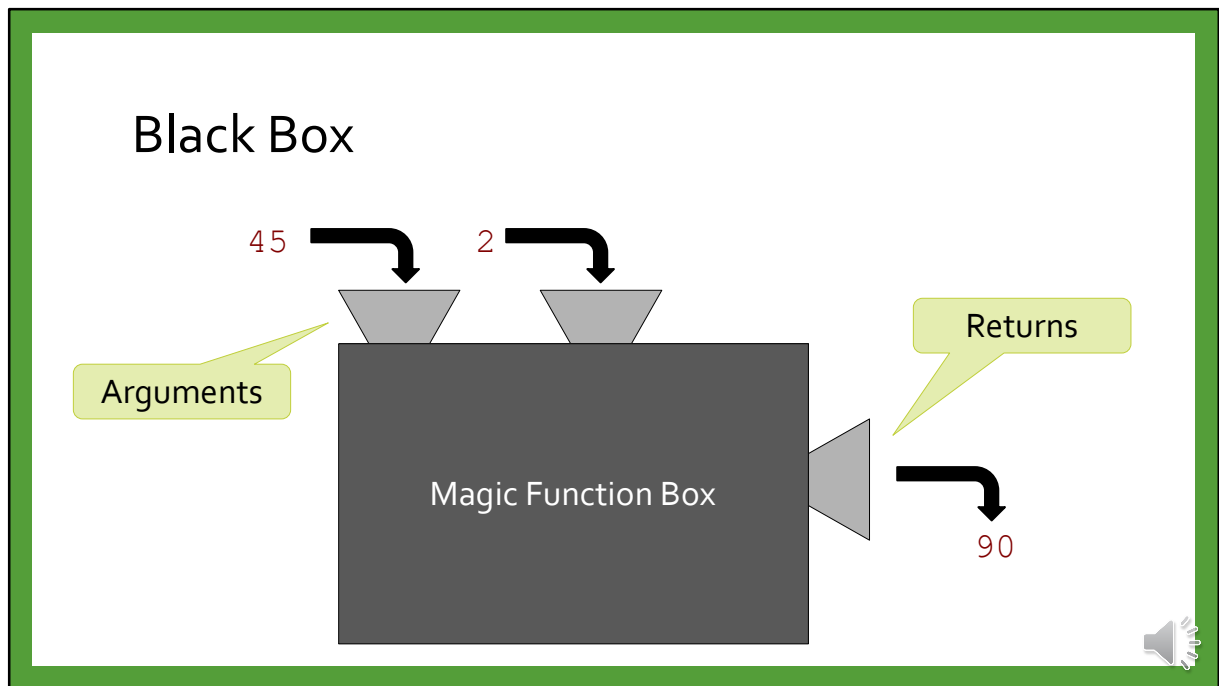


An important part of functions are "arguments".

Arguments are values that are given to functions that affect the behavior of the function.

Arguments for a function are placed inside the parentheses, and each one is separated by a comma.

We say that these arguments are being "passed" to the function.



Think of a function as a magic black box.
You cannot see inside the box to know how it works, but you do not need to.
The arguments come in through a funnel.
Returned values are dispensed through a slot