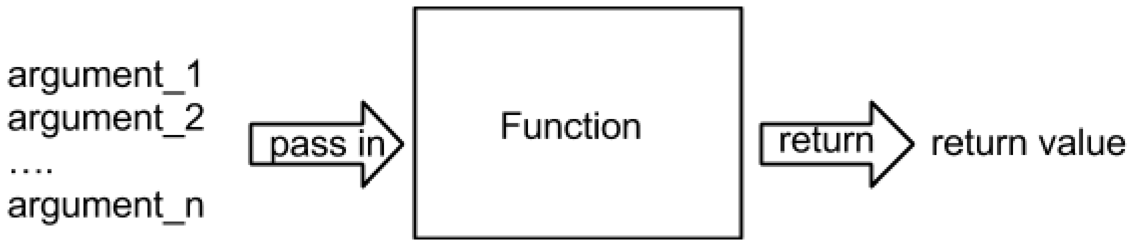
Fantastic Functions and How to Use Them

What's a Function?

A function is a subroutine that can be referenced by name.
A function takes in zero, one, or more arguments, completes some task(s), and returns a single value (or `void` for returning nothing).

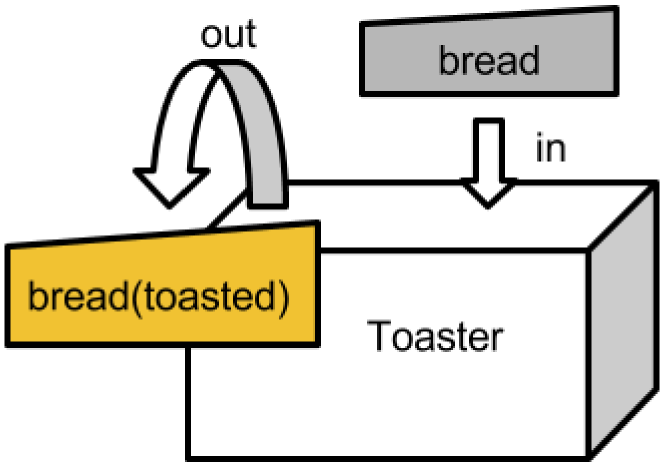


Arguments types: `int, float, String, char, ...`
Return types: `void, int, float, String, char, ...`
Example tasks: compute the sum of 5 numbers, draw a mouse face, print out an sentence

---

Real-world Analogy

Putting a slice of bread into a toaster and it pops out toasted!



If we were to program a makeToast function for our toaster using **Processing**, it should look something like:

output   input

```
toast makeToast(bread imbread) {
   /* some code that toasts
      the bread */
}
```

- Having the return and parameter types as a part of the function header is like setting a contract between the function designer and the user: in this example our **`makeToast`** function only accepts a `bread` type and always returns a `toast` type. Note that we also give names to our parameters (e.g. `bread` **`imbread`**) so that we can refer to them inside the function.

- When we are **writing** a function, there's no actual value associates with either of the arguments and the return value. (you would be very surprised if the toaster you just bought from the store already had bread inside!)

- When we want to use a function, we *call the function by its name and give it a set of argument(s) that match the parameter type(s)*. (Now it's time to add the bread!) In our example, the **`toast`** function needs a `bread` type as input, so we should pass in a `bread` as an argument. Assuming `wholewheat` is a variable of type `bread`, we can call the function by typing `makeToast(wholewheat);`

Section 5 Worksheet – Exercise #3

> Write a Processing function below that computes and returns the average of 3 given numbers.

Step 1: Thinking and Planning
Start by asking yourself some questions when you start designing a function:

1. What do I want the function to *accomplish*?
2. What do I want to *name* the function?
3. What should my function *take in*? (What are the parameter types?)
4. What will my function *return*? (What is the return type?)

Step 2: Write down the function header
Now use your answers to the questions above to fill in this template:

```
[4] return type  [2] function name ([3] parameter(s)) {
    // [1] some code that accomplishes the task(s)
}
```

For our question, we write down something like the following:

```
float average_of_three(float num1, float num2, float num3) {
     // do something
}
```

Step 3: Fill in the function body
To compute the *average* of three numbers, we first need to sum the three numbers and then divide by three. To do this, we can freely create and use a new variable:

```
float average_of_three(float num1, float num2, float num3) {
    float average = (num1 + num2 + num3) / 3;
    // are we done?
}
```

Final Step: Don't forget to return!

```
float average_of_three(float num1, float num2, float num3) {
    float average = (num1 + num2 + num3) / 3;
    return average;
}
```

Side Note: In this case, you can actually skip the variable and directly return the computed average like so:

```
float average_of_three(float num1, float num2, float num3) {
    return (num1 + num2 + num3) / 3;
}
```