Week 10: Functions

Elena & Willa

11/2/2020

What is a function?

What is a function?

A reusable chunk of code that lets you do something over and over again.

When you find yourself using copy-paste or repeatedly changing one variable in your code you probably need a function! .

What is a function?

A reusable chunk of code that lets you do something over and over again.

When you find yourself using copy-paste or repeatedly changing one variable in your code you probably need a function!

```
for (i in 1:100){

"do something"

}
```

```
for (i in 1:1000){

"do something"

    }
```

```
for (i in 1:10000){

"do something"

 }
```

What is a function?

A reusable chunk of code that lets you do something over and over again.

When you find yourself using copy-paste or repeatedly changing one variable in your code you probably need a function!

```
for (i in 1:100){

"do something"

}
```

```
for (i in 1:1000){

"do something"

}
```

```
for (i in 1:10000){

"do something"

}
```

Copy-pasting can get messy, increase the chance of errors, and makes your code harder to use and read.

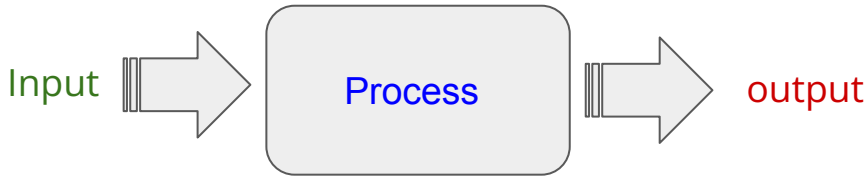
Functions in R

You are already familiar with a lot of built in functions in R and tidyverse functions mean(), print(), case_when() etc.

Functions in R

You are already familiar with a lot of built in functions in R and tidyverse functions mean(), print(), case_when() etc.

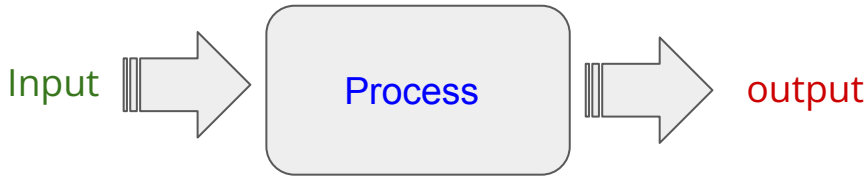
All functions take some kind of input (eg. your data), perform a process (eg. calculate the mean), and then return a value (eg. the mean of your data).



Functions in R

You are already familiar with a lot of built in functions in R and tidyverse functions mean(), print(), case_when() etc.

All functions take some kind of input (eg. your data), perform a process (eg. calculate the mean), and then return a value (eg. the mean of your data).



We can also create our own user defined functions in R.

R syntax for functions

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

R syntax for functions

What will you call your function (Can't use built-in function names)

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

R syntax for functions

What will you call your function (Can't use built-in function names)

*What are the **inputs** to the function?*

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

R syntax for functions

What will you call your function (Can't use built-in function names)

*What are the **inputs** to the function?*

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

What does your function do?

eg. Loop through something, calculate a mean, plot data etc.

R syntax for functions

What will you call your function (Can't use built-in function names)

*What are the **inputs** to the function?*

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

What does your function do?

eg. Loop through something, calculate a mean, plot data etc.

What does your function save out when its finished?

R syntax for functions

What will you call your function (Can't use built-in function names)

What are the *inputs* to the function?

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

What does your function do?

eg. Loop through something, calculate a mean, plot data etc.

What does your function save out when its finished?

Implementing a function

```
var_name <- name(arg1, arg2, etc..)
```

R syntax for functions

What will you call your function (Can't use built-in function names)

*What are the **inputs** to the function?*

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

What does your function do?

eg. Loop through something, calculate a mean, plot data etc.

What does your function save out when its finished?

Implementing a function

```
var_name <- name(arg1, arg2, etc..)

var_name
```

R syntax for functions

What will you call your function (Can't use built-in function names)

*What are the **inputs** to the function?*

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

Things that are specific to the user or change each time.

What does your function do?

eg. Loop through something, calculate a mean, plot data etc.

What does your function save out when its finished?

Implementing a function

```
var_name <- name(arg1, arg2, etc..)

My_mean <- mean(my_data)
```

Let’s consider some built-in functions

Pick a function, and then describe its 4 attributes (in English words).

```
NAME <- function(ARGUMENTS) {

  ACTIONS

  return(OUTPUT)

}
```

sum()	mean()
paste()	print()

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they take, additional parameters etc.)

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they task, additional parameters etc.)

You can choose to have no inputs (this is less common)

```
> f <- function() {
+         cat("Hello, world!\n")
+ }
> f()
Hello, world!
```

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they take, additional parameters etc.)

You can choose to have no inputs (this is less common)

```
> f <- function() {
+         cat("Hello, world!\n")
+ }
> f()
Hello, world!
```

You can set defaults. These become optional arguments.

```
f <- function(data, na.rm = TRUE){

ACTIONS

return (OUTPUT}
}
```

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they take, additional parameters etc.)

You can choose to have no inputs (this is less common)

```
> f <- function() {
+         cat("Hello, world!\n")
+ }
> f()
Hello, world!
```

You can set defaults. These become optional arguments.

```
f <- function( data, na.rm = TRUE){

ACTION

return (OUTPUT}
}
```

User must provide

if user doesn't provide NAs will be removed

```
Var1 <- f(df)
Var2 <- f(df, na.rm = FALSE)
```

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they take, additional parameters etc.)

You can choose to have no inputs (this is less common)

```
> f <- function() {
+         cat("Hello, world!\n")
+ }
> f()
Hello, world!
```

You can set defaults. These become optional arguments.

```
f <- function( data, na.rm = TRUE){

ACTION

return (OUTPUT}
}
```

User must provide

if user doesn't provide NAs will be removed

A little more on function arguments

When you write your own function you have to decide on the arguments.

- What information do you need from the user? (eg. their data)
- Do you want the user to have control over any additional info (eg. number of samples they take, additional parameters etc.)

You can choose to have no inputs (this is less common)

```
> f <- function() {
+         cat("Hello, world!\n")
+ }
> f()
Hello, world!
```

You can set defaults. These become optional arguments.

```
f <- function( data, na.rm = TRUE){

ACTION

return (OUTPUT}
}
```

User must provide

if user doesn't provide NAs will be removed

```
Var1 <- f(df)
Var2 <- f(df, na.rm = FALSE)
```

The anatomy of a function in R

Functions have 4 attributes (adapted from YaRrr!):

1. **Arguments**: What are the ***inputs*** to the function (e.g., a vector of numeric data? Or some text?)? You can specify as many inputs as you want.
2. **Actions**: What do you want the function to do with the inputs? (e.g., Create a plot? Calculate a statistic? Run a regression analysis?) This is where you'll write all the real R code behind the function.
3. **Output**: What do you want the code to ***return*** when it's finished with the actions? (e.g., a scalar statistic? A vector of data? A dataframe?)
4. **Name**: What will you call your function? Should be descriptive. Can be any valid object name, but be careful not to use names of existing functions!

https://bookdown.org/ndphillips/YaRrr/the-structure-of-a-custom-function.html