Acting On Variables

Plan for today

- Mini "Quiz" on **objects**, **classes**, and **indexing**
- Actions we perform on our objects
    - **operators**
    - **functions**

Go take your first mini quiz!

When you're done, come back here

Operators

An **operator** is a simple calculation

+	addition
-	subtraction
*	multiplication
/	division
^	taking powers
%%	modulus

Order of Operations

Important note: Order of operations matters!

```
(8-4)/2
```

```
## [1] 2
```

```
8-(4/2)
```

```
## [1] 6
```

Logical Operators

Test whether a statement is `TRUE` or `FALSE`

`==`	equality
`!=`	inequality
`>`	greater than
`>=`	greater than or equal to
`<`	less than
`<=`	less than or equal to

Logical Operators

- Return a value of `TRUE` or `FALSE`
    - `empire$gender == "female"`
- Which Starwars character is more than 150 cm tall?
    - `empire$height > 150`
- Are any Starwars characters exactly 150cm tall?
    - `empire$height == 150`
    - **WARNING:** `empire$height = 150` will change your data!

Functions

- R is not *just* a calculator. You often want to do something more complex.
- To perform more complicated actions, we use ***functions***
    - **functions** are commands that describe, manipulate, or analyze objects
    - They are the **verbs** of programming languages
    - This is why we use R! No one wants to calculate a regression by hand...

Functions have 3 parts

Function name

- Each function has one and only one name

```
# The function name is `log`
log(10)
```

```
## [1] 2.302585
```

Functions have 3 parts

Arguments

- One argument is always specified -- the input; this is the object that the function acts on.
- Other arguments control *how* the function acts. For example, do you want the natural log? Or log base 10?
- Each function has defaults for it's arguments. You should know where to find these and how to change them (we will talk about this in the next video).

```r
# The argument here is the input, or `10`
log(10)
```

```
## [1] 2.302585
```

Functions have 3 parts

Output

- The output of a function can be *any* of the object types & and of any class or even a combination of these
- Outputs can be a single value, vector, data.frame, matrix, list, or a plot
- *You can store the output by assigning it to another object!*

```r
# The output is `2.302`
log(10)
```

```
## [1] 2.302585
```

```r
# If we want to store `2.302` for later
newObject <- log(10)

# Now print out what is contained in `newObject`
newObject
```

```
## [1] 2.302585
```

Mathematical functions

Some obvious ones:

- `sqrt()` square root
- `round()` rounding a number
- `log()` logarithm
- `exp()` exponentiation
- `abs()` absolute value

Example:

```r
sqrt(85)
```

```
## [1] 9.219544
```

Functions you'll use a lot!

`c()` - combine or concatenate

`length()` - find out how long a vector is (this is the same as getting the last position)

`factor()` - change a character vector into a factor vector (is there meaning? Ex: treatment vs. control, male vs. female, session 1 vs. session 2 etc.)

`table()` - really nice for getting quick counts (ex: how many males and females are there?)

`cbind()` and `rbind()` - add a vector to an existing data.frame. `cbind()` adds a new column. `rbind()` adds a new row

Multiple arguments

Most functions take more than 1 argument (more than just the input object).

Separate these arguments with commas `,`

```r
round(x = 5.86921, digits = 3)
```

```
## [1] 5.869
```

Arguments have names

Use the argument names!

```
# perfect
round(x = 5.86921, digits = 3)
```

```
## [1] 5.869
```

```
# also perfect
round(digits = 3, x = 5.86921)
```

```
## [1] 5.869
```

Arguments have names

Use the argument names!

```r
# right answer bc right order
round(5.86921, 3)
```

```
## [1] 5.869
```

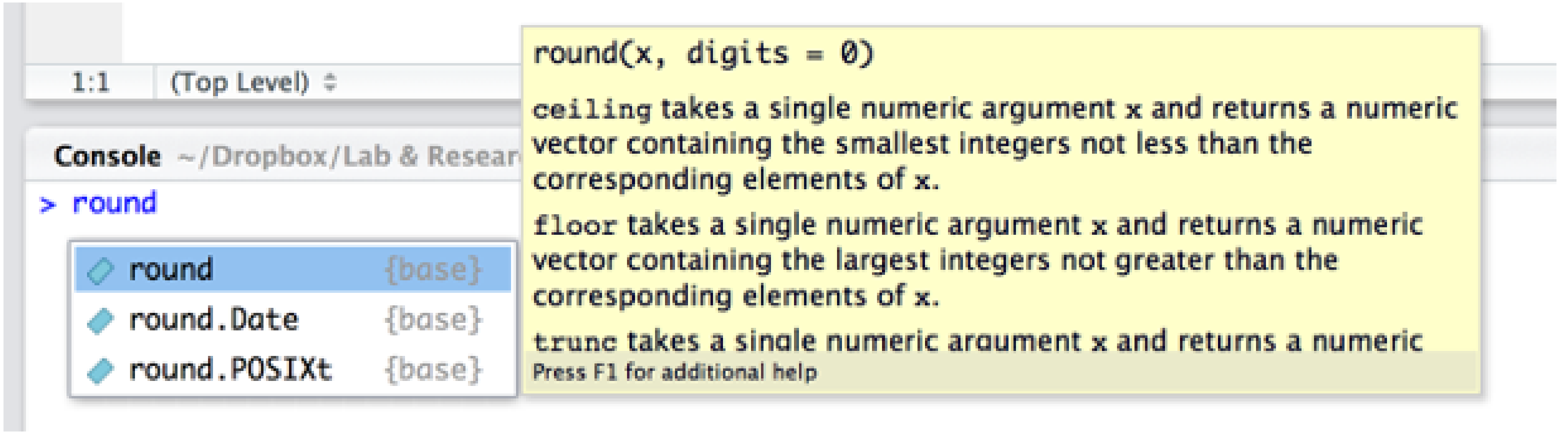
```r
# wrong answer bc wrong order
round(3, 5.86921)
```

```
## [1] 3
```

Great, but how do I know what the arguments are for a function?

Two ways:

1. In RStudio, press the **tab** key to see the names of arguments and descriptions. (*note, this might not work in the online practice assignments, but it should definitely work when running RStudio locally*)



```
1:1   (Top Level) ÷
Console ~/Dropbox/Lab & Resear
> round
  round         {base}
  round.Date    {base}
  round.POSIXt  {base}
```

round(x, digits = 0)

ceiling takes a single numeric argument x and returns a numeric vector containing the smallest integers not less than the corresponding elements of x.

floor takes a single numeric argument x and returns a numeric vector containing the largest integers not greater than the corresponding elements of x.

trunc takes a single numeric argument x and returns a numeric

Press F1 for additional help

Great, but how do I know what the arguments are for a function?

Two ways:

1. Look in the R Documentation
    - This is what we'll talk about in the next video, so stay tuned

Go do Practice #1

When you are finished, go on to the next video