Lab 01

Statistical Computing & Programming

Meet your TAs

- Quinn Frank
  - quinn.frank@duke.edu
  - Office hours (Zoom link in Sakai)
    - Wednesday 11:30am - 1:30pm ET
- Pierre Gardan
  - pierre.gardan@duke.edu
  - Office hours (Zoom link in Sakai)
    - Tuesday 10:00am - 11:00am, 6:00pm - 7:00pm ET
- Sarah Mansfield
  - sarah.b.mansfield@duke.edu
  - Office hours (Zoom link in Sakai)
    - Friday 11:30am - 1:30pm ET

We'll also help answer your course-related questions on Slack.

Lab structure

- Work on the assigned lab.
- Work on homework (if outstanding).
- Ask questions about anything.

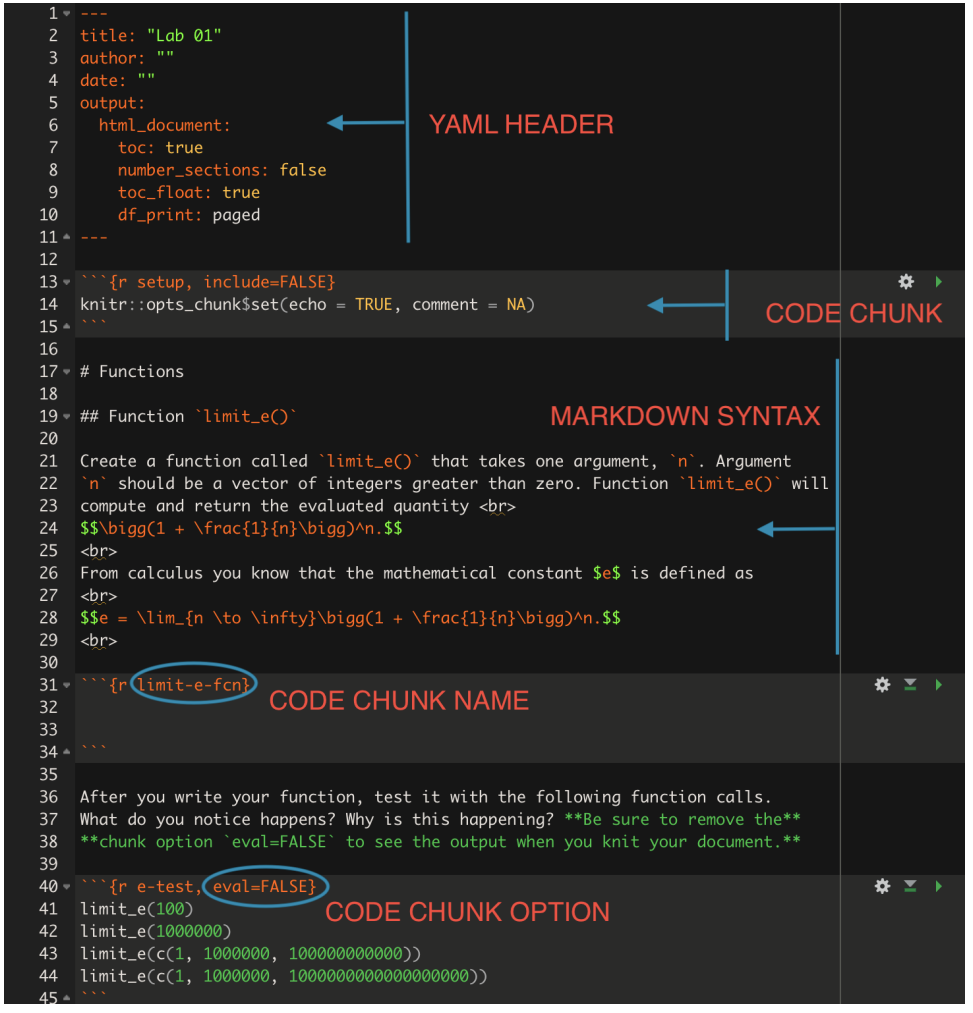
Getting started

- Navigate to https://classroom.github.com/a/CEUBWaok to accept Lab 01 and create your private repository within our GitHub course organization.
- Open an RStudio session (Rook or Knight or locally). If your NetID begins with A - J, use Rook, everyone else use Knight.
- In RStudio go to
  - `File > New Project`,
  - select `Version Control`,
  - select `Git`,
  - navigate to your repo on GitHub: `sta323-523-sp21/lab_01-[github_username]`,
  - click the green `Code` button and copy the git URL,
  - paste the git URL into the dialog box in RStudio,
  - click `Create Project`.

R Markdown overview

- Generate fully reproducible reports - the analysis is run from the beginning each time you knit
- Simple Markdown syntax for text
- Code goes in chunks, defined by three backticks, narrative goes outside of chunks

Navigating R Markdown



Sample R Markdown syntax

Header syntax	Example
# Level one	**Level one**
## Level two	**Level two**
### Level three	**Level three**
#### Level four	**Level four**
##### Level five	**Level five**
###### Level six	**Level six**

Syntax	Example
bold text	**bold text**
italicized text	*italicized text*
- one - two - three	• one • two • three
`` `in-line code` ``	`in-line code`

R Markdown resources

- In RStudio: `Help > Cheatsheets > R Markdown Cheat Sheet`
- In RStudio: `Help > Cheatsheets > R Markdown Reference Guide`
- R Markdown: The Definitive Guide

Getting started with version control

- With your RStudio project open, do a quick git configuration in the console pane with

```
# install.packages("usethis")
library(usethis)
use_git_config(user.name = "Shawn Santo",
               user.email = "shawn.santo@duke.edu")
```

  Replace my info with your name and email address associated with GitHub. On Wednesday we'll cover how to save your credentials.

- Follow the TAs live demo on how to stage, commit, and push. Today you may use the Git GUI in RStudio; on Wednesday you'll learn how to use git from the command line.

- As you work, make commits and push your work after you make significant progress.

Functions overview

Most functions take inputs and end by returning objects. Functions are comprised of arguments, the body, and the environment. For now, our focus will be on functions that take vectors as input and output vectors.

```
count_letters <- function(x, special_characters = FALSE) {
  if (special_characters) {
    return(table(unlist(strsplit(x, split = "")))) # explicit return
  }

  x_split <- unlist(strsplit(x, split = ""))
  x_split_filter <- x_split[x_split %in% c(letters, LETTERS)]
  table(x_split_filter) # implicit return
}
```

- `count_letters` is the function's name
- `x` is a required argument
- `special_characters` is a default argument
- This functions uses an early explicit return and a late implicit return for when `special_characters` is `FALSE`.

Functions overview

```
count_letters("Hello")
```

```
#> x_split_filter
#> e H l o
#> 1 1 2 1
```

```
count_letters("counting letters!!!!", special_characters = TRUE)
```

```
#>
#>   ! c e g i l n o r s t u
#> 1 4 1 2 1 1 1 2 1 1 1 3 1
```

```
count_letters("counting letters!!!!")
```

```
#> x_split_filter
#> c e g i l n o r s t u
#> 1 2 1 1 1 2 1 1 1 3 1
```

Today's objectives

- Get comfortable with the version control cycle and R Markdown.
- Complete Lab 01.
    - Work with those in your breakout room.
    - This is not graded.
- Ask any questions about any recent course materials.