Welcome to The Carpentries Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

SCROLL DOWN!

Workshop page: https://ua-carpentries-workshops.github.io/2019-10-26-Tucson/

Pre-worshop survey: https://www.surveymonkey.com/r/swc_pre_workshop_v1?workshop_id=2019-10-26-Tucson

- **Instructors Day 1**
  - Sateesh Peri
  - Noah Nelson
  - Marnee Dearman
- **Instructors Day 2**
  - Xue Pan
  - Travis Struck
  - Kelsey Gonzalez
- **Assistants**
  - Drake Asberry
  - Brian Bell
  - Upendra Devisetty
  - Jennifer Kadowaki
  - David LeBauer
  - Torbet McNeil
  - Cesar Medina
  - Adriana Picoral
  - Fernando Rios
  - Austin Rutherford
  - Saren Seeley

Welcome to the UA Software Carpentry Workshop on October 26 and 27, 2019

Please enter your name into the box on the upper right hand corner of this pad, then continue the list below, typing your name again and briefly introducing yourself.

Then, complete the pre-workshop survey if you haven't done so, yet: https://www.surveymonkey.com/r/swc_pre_workshop_v1?workshop_id=2019-10-26-Tucson

---

Reminder: paste a red sticky note on the frame of your computer if you need help!

Introductions:

1. Hi, I'm Uwe Hilgert. I work at the BIO5 Institute and hail originally from Germany.
2. Hi, I'm Kelsey Gonzalez and I'm a PhD Student in Sociology. I'll be teaching part of the R workshop tomorrow. (She/her)
3. Hi, I'm Drake Asberry and I'm a PhD student in Second Language Acquisition and Teaching. I'll be helping out with this workshop.
4. Hello, my name is Yulia, a PhD student from the Arid Lands Resources Sciences department.
5. Hi, everyone! Elliot here. They/them pronouns, Sociology PhD student right here at U of A.
6. Hi, I'm Natalie. I am a Genetics PhD student at U of A.
7. Hi, I'm Stacey (she/her). I'm an Assoc. Professor of Anthropology at UA
8. Hi all, I'm Heather a adjunct Biology faculty at CNM, Albuquerque NM.
9. Hi everyone. I'm David. I'm a visiting scholar at SLAT researching corpus linguistics.
10. Hi, my name is Jazmyn and I am a undergrad biology major here at the U of A (she, her, hers)
11. HI Everyone! I'm Brian Bell from the Applied Math PhD Program
12. Hi everyone, I'm Dennis, an undergrad in mathematics at UA
13. Goodmorning everyone. Jen here from Albuquerque. Biology faculty at CNM or Central New Mexico Community College
14. Blaine here - Psychology PhD student. She/her and they/them pronouns. Tweets @LifeWithoutAwe
15. Hi. My name is Jen and a first year Master's Student in the Hydrology department at UA
16. Good morning everyone. Research specialist in ACBS of UA.
17. Hello, My name is Katherin I am a first year ABBS student.
18. Hi, Molly Douglas here, acute care surgeon at BUMCT.
19. Hi, Samer Asmar here, Research fellow in trauma surgery at BUMCT ***#TRAUMAREADY***
20. Hi I'm David Williams I am a grad student in the cancer bio program.
21. Hi I',m Xinru Wang. I am a posdoc from the chemistry and biochemistry department.
22. hiHi This is Senthil from CBC
23. Hi, this is Juliane Weber, I am a staff scientist at lunar and planetary science at UoA. I am also from Germany originally.
24. Hi, I'm Megan and I am a post-doc in Psychology here at UA.
25. Morning everyone, I'm Mohamad, I'm a research fellow at the department of surgery at BUMCT
26. Hi, I'm Kamil Hanna,trauma surgery research. #AZTRAUMACATS
27. Hi everyone! My name is Lia and I'm a Research Specialist in the Department of Environmental Science. She/her pronouns.
28. Hi this is Khurrum, Research Fellow at the department of Acute care Surgery at Banner Heaalth.

29. Hi, I'm Aoxue, PhD student in Optical Science.
30. Hi, my name is Nicole. I am a PhD student at the School of Plant Sciences, former ABBS student.
31. Hi, I'm Isamu, a professor at the Lunar and Planetary Laboratory
32. Hi I am Alma, a MS student in Environmental Science (she/her)
33. Hello I am Chet from the staff of the OSIRIS-REx mission and planetary sciences department (he/his)
34. Hi, I'm Ernesto, a postdoc at the BIO5.
35. Good morning! I'm Ran Wei, a research technician in Immunobiology

UNIX Helpful reminders:

- q : quit/escape
- man: access manual page
    - e.g. man cd
    - e.g. man date
- --help: see quick help (list of arguments for command)
    - e.g. cd --help
    - e.g. date --help
- pwd: print working directory
    - This will show you your current location
    - e.g. pwd
        - /c/Users/[Username]
- ls: list
    - what files do I have in this folder?
- Tab complete: start typing the name of the file or folder, then hit tab to automatically complete the rest of the name
    - This will help you avoid typos and save you time
- cd: change directory
    - cd ../ moves up one folder
    - don't have to use absolute path in cd, relative path is okay
- less: allows you quickly scroll through and view contents of file
    - e.g. less example.txt
- **If a Command hangs, you can type "Ctrl + C" to cancel the command**
- wc: word count
    - wc -l : line counts
    - e.g. wc -l example.txt
- cp: copies files usage: cp [file] [copy of file]
    - cp -i [file] [copy of file] will ask permission before overwriting files
- mv: moves file. mv [file] [new path]
    - e.g. mv example_copy.txt experiment/
- rm: delete
    - e.g. rm example_copy.txt
    - there's not trash bin! Be careful
- nano: create or edit a plain text document
    - to save, ctrl + O
    - to exit, ctrl + X
- mkdir: make directory : makes a new folder
    - e.g. mkdir subset
- rmdir: removedirectory : removes a folder
    - e.g. rmdir subset
    - will not remove a non-empty directory unless using rm -r which recursively goes in and delete files and folder

- | : Pipe : moves output of one command into the next command
    - e.g. ls | wc -l
    - e.g. ls data/all_samples/ | wc -l
- > : "send to file" , overides target file
    - represents send the output of one command to this file/folder location
    - overwrites previous output
    - >> append instead of overwrites
- wildcards
    - * : list out any file name that appears at least 0 times
        - e.g. ls *.fq | wc -l
        - e.g. mv *.fq fastq_files/
    - ? : list out any file name that appears at least 1 time
        - e.g. ls fastq_files/sample_1?.fq
- history : lists log of all commands in this session
    - e.g. history >> command -log.txt
- cut: cuts columns from tables (cut [option] [column number] [file])
    - e.g. cut -f 1,3 gene_annotations.tsv | head
    - 1,3 : colums 1 & 3
    - 1-3 : columns 1 through 3
- grep: finds matching text pattern and returns lines
    - e.g. grep blue colors.txt
    - e.g. grep re colors.txt
    - -c : counts how many times pattern matched
        - grep -c K18979 gene_annotations.tsv
- paste: sticks columns or files together horizontally
    - e.g. paste colors.txt colores.txt
    - caution number of rows may not match so it may not make sense
- variables
    - name=value e.g. my_var=TUCSON
    - $ call value / evaluates e.g. $my_var
- echo repeats what you type
    - echo $my_var
- loops
    - for: set the loop variable
    - in: specifgy whatever it is we are looping over
    - do: specify what we want to do with each item
    - done: tell the computer we are done telling it what to do with each item
    - e.g. for item in earth mars venus; do echo $item; echo $item >> planets.txt; done
- cat : concatenate
    - prints out all the contents of the file
    - e.g. for item in $(cat planets.txt); do echo $item; done
- bash : run bash .sh script

UNIX Questions:

What is the difference between the "ls" command and the "dir" command

- one is from windows, one is from linux -- both should (might) be accessible through the git bash script
- dir also produces output in a slightly different format
- Got It. Thanks

How can I go one folder backwards?

- You can use cd ..
- Awesome :)
- I think we'll actually go over this in a minute as well
- cd ../
- Is cd ../ going back to the root directory?
- no, just up one folder "backwards"
- Hmm cd.. and cd../ do the same thing
- Yes, that because it can be built onto
    - eg. cd ../.. moves up two directories
- / represents a directory, but its not completely necessary for the command but its good practice
- (Y)

is there a way to have what you open using the less command to open on a separate window?
try
start example.txt
does that give you what you're asking?

How to change the ending of a file? By accident I made a .tst file instead a .txt file and I want to change it back
we're about to show how to move and rename, and that will remedy the .tst Thanks!

Is there a way to escape a command?
q if you're stuck in a printout or ctrl + c if its on the line itself

What exactly do you use Unix for?
Anything you use a computer for! Most scientific computing is done in linux because it's (MUCH) cheaper and more configurable than windows.
Thank you
in addition, manging files, rewriting file structures, for example changing a csv to a tsv etc.

how is > different than cp or mv
cp and mv work on stuff that already exists as a file/directory > is taking the output from a command and making a file for the first time
ok great thank you

can you use ls to list files that were modified on a certain date or in a certain date range?
yes -- this is a good exercise =D (Hint, check out another command "find" by looking at "find --help")
Thanks! Oh maybe I need to look online. I didn't understand the output
You can also use the "man find" to get a description of the arguments listed by the --help
Thanks!

Git Helpful reminders:

- git config --global user.name "YOUR NAME HERE"
- git config --global user.email "YOUR GITHUB EMAIL HERE"
- git init : initialize git repository
- git add: add files to staging area for committing
- git commit: creates a checkpoint in time which can be reverted to later
    - git add practice.txt
    - git commit --message "practice git commits"

- rm -r remove git repository
- touch : create a file without going into nano
- git status: use this to see the changes that have been made to your git repository
- git diff : see differences
- git log: see the last 10 commits and their associated messages, who changed, etc.
- git push: send to github
- git pull : "download" from github.. kind of "sync"

Git-based Work Cycle

Now that we have set up a remote repository, let's revisit our Git-based work cycle.
Here's what I do every time I sit down to work on a project that I'm tracking with Git:

-
    - git pull -- Pull changes from the remote repository. I do this even if I haven't changed anything just to be safe and make it a habit.
    - git log -- Remind myself of my recent activity in the repository. What are my recent commits?
    - Do some work.
    - git status -- Remind myself of the work I just did, in broad strokes (files added, deleted, or modified).
    - git diff -- Remind myself of the work I just did, in terms of the specific changes made.
    - git add <file> -- Once I'm sure I want to commit changes to a given file, I add the file to the staging area.
    - git commit --message "..." -- Commit all the changes that I have added to the staging area with a clear message.
    - Repeat steps 3-8 as necessary
    - git push -- When I'm done, I push the changes to my remote repository.

Git Questions:

What is the benefit of using nano over adding text with commands (within Git)?
There's no specific benefit, just two ways of doing the same action. Some would say adding the text with --message is just more straightforward.

How do I get rid of a line that says "untracked files"
Untracked files are files that are not currently being tracked by git. If you want to track them, you can use the git add. If you do not ever want to track this file we would add it to the .gitignore file (we may talk about this later in the lesson)

If you are using GIT for a project with many collaborators, do they have to also use Git to track changes?
yeah, tools like github are great ways to work with collaborators so everyone is tracking and uploading from their personal local and shared remote repositories. There are other tools like git, but everyone needs to be using the same tool.

How do we see the message we made for the commit stage?
you can use "**git log**"
I see. Thanks!

So if you dont stage the changes before committing, they will not be commited?
that is correct.
Is there any control to know if you staged changes before committing?
"git status"
will show everything currently staged for commit

thanks!

How are conflicts resolved when you're pushing and pulling and two people have edited the same file?
we won't cover this today, but this lesson https://swcarpentry.github.io/git-novice/09-conflict/index.html will go over this. in short, you will get a merge conflict and have to decide what to keep and get rid of normally using nano.
Thanks!

Git Challenge

1. Change the line you wrote to practice.txt to be something else
2. Have Git tell you what you changed in practice.txt. How is a change to a single line represented?
3. Add a new file to the git_lesson directory. Call it whatever you want.
4. Make a *single commit* with *both* of these two changes included in the commit
5. Review the commit history to confirm your last commit

Day 1 Challenge https://github.com/UA-Carpentries-Workshops/2019-October-Resources/blob/master/Day1_Challenge.md
In the workshop directory, you should have another directory called unix_challenge. Inside the directory are a number of files.
One file is called country.txt, the rest are files corresponding to different countries.Take a look at country.txt and a couple of the other files to get a feel for what they contain.
Here's what we want you to do:

1. Append all the content of the various country files into country.txt. This way, the contents of country.txt will appear like a "header" for the data in the other files. Verify that you did this successfully.
2. Using grep, extract all the rows of data corresponding to the year 1997 and put them into a separate file.
3. Commit these changes to your workshop repository and push your changes up to GitHub!

for item in *.cc.txt; do cat $item >> countries.txt
cat *.cc.txt >> countries.txt
grep 1997 countries.txt > 1997.txt <- but watch out, there's errors!

Where is the challenge link
^^ right above ty

Can we review the shell lesson tomorrow? I just missed all of it and can't catch up 'cause I don't see a page for the lessons :pray

https://ua-carpentries-workshops.github.io/2019-10-26-Tucson/
To review before tomorrow, see
https://github.com/UA-Carpentries-Workshops/2019-October-Resources/blob/master/unix/index.md
https://github.com/UA-Carpentries-Workshops/2019-October-Resources/blob/master/

git_lesson/master_lesson.md

DAY 2

Intro exercise:

1. Open terminal
2. Change directories (using an absolute path) to your workshop directory using `cd`
3. Check which directory you're in using `pwd`
4. List files in directory, including hidden files using `ls -a`
5. Check that our local repo is synced with remote repo using `git pull` before we start changing stuff
6. Create a new directory using `mkdir` called **r_lesson**
7. Copy the file **.gapminder_clean.csv** to **r_lesson** using `cp`, making the file *visible (i.e., remove the dot at the start of the filename so the file is named **gapminder_clean.csv**)*
8. Check that the new file went where you wanted it to go using `ls r_lesson`
9. Check status of local repo (workshop folder) using `git status`
10. Tell git to track new directory we made using `git add`
11. Re-check git status to confirm that we added the new directory and files inside it
12. Commit change using `git commit` and give your change an explanatory message
13. Push changes to remote repo using `git push`

IF YOU GET STUCK IN VIM:

- Press Esc.
- Type **:q!**
- Press Enter.

R Intro Notes:

- If you get stuck in an incomplete command (you get a plus sign in the console instead of an answer or the > prompt):
    - (this means that R is waiting for more information from you)
        - Press Esc then Enter to get back to the ">" prompt
            - You can also hit the "stop" symbol or use CTRL+C
- Comment your code using # at the start of the comment
    - For example, *# This line creates a composite variable from the item-level variables prefixed "pss"*
    - Commenting your code is VERY important and a good habit to get into
        - Other people can understand what your script does
        - ...and YOU can understand what the heck your script does when you come back to it after not looking at it for a few months :)
- What is a variable in R?
    - Variables in R are objects in the global environment that store particular types of data (or *vectors*)
    - For example, if you have run the line "x <- 3", now every time you call x, R will know that x = 3
        - We'll get into this later but we will use variab
        - les again shortly when we start loading data, to store our dataset (for example, making a variable called "my_data" that holds the contents of "my_data.csv"
    - Variable name must start with letter, not a number

Q1. Which of the following will NOT return TRUE?

A. FALSE == FALSE

B. 10-5 == sqrt(25)
C. TRUE > FALSE
D. 'a' > 'b' XxxxXxXxxxxx
5 == 6 XxX

Q2. What is the output when we execute the following code? x <- 3 y <- 2 y <- 17.4 x+y
x <- 3
y <- 2
y <- 17.4
x+y
A. [1] 3 2 17.4
B. [1] 22.4
C. [1] 20.4xXXxxxxXXxxxxx
D. [1] 5

Q3. Use str() to look at the structure of the dataframe and summary() to get information about the variables.
What are its columns?
How many rows and columns are there?1704 , 6
What is the earliest year in the year column?1952
What is the average life expectancy?59.474
What is the largest population?1318683096

R Intro Questions:
I assume that Class() will never come up as string then?
No because string is not the name of a data type in R; equivalent is 'character' https://swcarpentry.github.io/r-novice-inflammation/13-supp-data-structures/
Thank you

When to use == instead of =?

- = is an assignment (i.e., a = 3 means "make a equal to 3") similar to <- that we will learn soon
- == is an evaluation (i.e., a == 3 means "TRUE or FALSE: is a equal to 3?")

Not sure if we are covering this soon; but is there a way to bracket or semi-colon to run multiple commands in a row in a single input? Thank you.

Yes, we will get to this soon :) There are a few different ways to run multiple commands in a single line in R. More on this later, but in brief...

-
    - In base R, we use 'nesting' with ()
    - In dplyr and other packages that use a slightly different syntax where code is read from left to right (instead of innermost parantheses to outermost parentheses), we use something called a "pipe" represented by %>% to chain multiple commands and/or their outputs together
        - We'll go through this in the dplyr lesson this afternoon

- Oh, I may have misunderstood what you were asking - were you asking if you can run multiple lines at once?
- The answer to that is also yes :)
- I meant closer to this, was thinking something along the lines of x<-3; y<-2; x+y or something along there, will get to it later, thanks for help
-
    - What happens when you try it out in your RStudio? :)
        - (Yes, you can use semicolons but for readability/to decrease likelihood of errors and typos, line breaks are preferred)
        - Ok, thanks.
- To run multiple lines of a script at once:
-
    - You can highlight all the lines of your script that you want to run, and hit the Run button (or Command + Enter on Mac)
    - R runs the lines in sequential order, so you can create something in line 1, then reference that thing on line 2
        - For example, if you have the lines
-
    -
        - my_var <- 42
        - my_var / 2
        - By the time R gets to the second line, it knows that when we say "my_var" we mean 42, so it gives us the answer 21 in the console

can you add a column before adding data to the column?
Moving your q down to the Dataframe Q's section so it won't get overlooked...

R Dataframes Notes:

- Dataframes in R hold multiple types of information and multiple columns
    - You can picture this as what you see when you open up an Excel or SPSS dataset, where we have multiple rows and columns, and different columns can hold different types of information (strings would be a "character" column, numeric values would be a "num" column, and so on)
    - To look at your dataframe called "df", you can run:
-
    -
        - **View(df)**

<u>Useful functions for getting info about dataframes:</u>

- **class()**
    - tells you the type of column
- **ncol()**
    - number of columns
- **nrow()**
    - number of rows
- **dim()**
    - tells you the dimension (number of rows and number of columns) of the dataframe
- **str()**

    - provides info about the structure of the dataframe: what's inside the dataframe?
- **summary()**
    - basic descriptive stats
- **methods (class=class(df))**
    - full list of methods that can be used with df (where df is the name of a dataframe)
- **head** or **tail()**
    - shows you first or last several rows of your dataframe (quick way to preview and make sure things look like you are expecting)
- **colnames()**
    - prints the names of all the columns in the dataframe

A few functions for working with dataframes:

- **cbind()**
    - for adding new columns
- **rbind()**
    - for adding new rows
-
    -

Spaces are important! They make code more readable and therefore less prone to error - see Wilson et al 2014 best practices in Scientific Computing for explanation of why readable and consistent style is important https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745. Also for R style guidelines http://adv-r.had.co.nz/Style.html. Rstudio will reformat your code according to these guidelines using Code >> Reformat code or the keyboard shortcut ctrl + shift + A

To reference a particular column within a dataframe, use the **$** operator

- For example, `gapminder$year` references the column "year" within the gapminder dataframe
    - `mean(gapminder$year)` gives us the mean of all the values in the year column

R Dataframes Questions:

can you add a column before adding data to the column?
Yes, you can add an empty column with cbind(New_Column=NA) if New_Column doesn't exist, this new column will be filled with NAs

Could you subset data that are not in consecutive rows/columns?
Good question! Yes, you can. CTRL+F for "non-sequential" in this lesson: https://swcarpentry.github.io/r-novice-inflammation/10-supp-addressing-data/

- The **c()** function in R is used to combine different things together, including non-consecutive rows and columns.
    - **new_df <- df[c(1, 5, 7, 9), c(2, 4, 6, 8)]** *# makes a new dataframe called new_df containing ONLY rows 1, 5, 7, & 9 from ONLY columns 2, 4, 6, & 8*
    - You can reference columns either by their name OR by their location (first, second, third.....fiftieth column in the dataframe)
        - CAUTION!!! It is often preferable to reference column by name rather than location, because it's possible that column order may change as you work with the data. For example, something changes and now

what you think is in the first column is actually now in the second column...

What is indexing?

Indexing means referencing a particular row and / or column to return one or more elements of a dataframe. A row (or column) number is its index

what's difference between "["and "(" when to use which?

- Parentheses () are for passing arguments to functions, and you'll generally see them used like this: **do_this_function(to_this_data)**
- Brackets [] are used for indexing and subsetting. There are two arguments that go within brackets, separated by a comma. The first argument is for row or rows, the second is for column or columns, like so: [rows, columns]

R Plotting Notes:

To install the package ggplot2:
install.packages("ggplot2")

- then you will need to load the package once it is installed, using **library(ggplot2)**
    - If it asks if you want to install from source, answer 'no' (type n and hit Enter)

The basic concept for ggplot is that you have a plot base, then specify the plot type and modify the elements of the plot by adding "layers":
ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp)) *# this tells ggplot to get the data from the gapminder dataframe, and tells it that we want "year" on the x axis and "lifeExp" on the y axis)*

So far, this is just our base plot, so next we have to tell it what kind of plot we want. We want a scatterplot so we add a layer using the "+". This tells ggplot that we want it to take the information from the previous layer (our base plot) and pass it to ggplot's scatterplot function **geom_point()**:

ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp)) + geom_point()

From there, we can keep adding layers to modify different elements of the plot, like the axis labels, color and size of the datapoints, etc. using a "+" each time.

- *NOTE:* The + needs to be on the preceding line (or the same line), not on the new line.
- This will *not* work because the + is on the next line:
- **ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp))**
- **+ geom_point()**
- ...But this does work:
- **ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp)) +**
- **geom_point()**
- The "+" tells ggplot, "wait, I'm not done yet! Stay tuned for more stuff coming on the next line"
- You can keep adding new layers pretty much indefinitely.

It is often useful to store your plot as an object, so that we can do things like save it, display it again in the future without having to run the ggplot code again, etc.
my_plot <- ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp)) + geom_point()

Now if we type my_plot, highlight, and run it, it will display the plot again!
We can also save it as an image file:
ggsave(filename = "my_plot.png", my_plot) *# saves the plot that we assigned to the object "my_plot" to a file called "my_plot.png"*

What if I want to save my plot image in a specific location?

We can specify where we want R to save the image by specifying the path in the first argument:
ggsave(filename = "/Users/sarenseeley/Desktop/my_plot.png", my_plot) *# saves my_plot as a .png file to my Desktop*

EXERCISE:
3e. Now change the shape of the points and color the points by the continent
Note: You'll get multiple linear model fits
Note: ?geom_point

Answer:
```
ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp,
color=continent)) +
geom_point(shape=17, size=3) + scale_x_log10() +
geom_smooth(method = 'lm')
```

Unpacking the syntax of the exercise answer:
 # this first part of the syntax says "make a plot object using data from gapminder, put gdpPercap and lifeExp on the x and y axis, and make data points from each continent a different color"

- *# the + says "we're going to add a new layer after this":*

ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp,
color=continent)) +

 # the second line tells ggplot a few different things:
 # first, that we want this to be a scatterplot, and that we want each datapoint to be a particular size and shape

- *# then we add a new layer*
- *# then we we scale the x axis using a log10 transformation (to scale the x axis, we would use **scale_y_log10()**)*
- *# then we get ready to add another layer:*

geom_point(shape=17, size=3) +

- **scale_x_log10() +**

 # finally, we add a regression line

- *# (method = "lm") means "fit a linear model":*
- *# (use **?geom_smooth** to see other methods you can use)*

geom_smooth(method = 'lm') *# no + at the end of the line here because we're done*

EXERCISE
Create a density plot of GDP per capita
Tips:
We only need one variable
"geom_" is the start of fuctions for displaying data

-
    - TIP: hit Tab after typing "geom_" to see all the different geoms you can use)

ggplot2 REFERENCES AND CHEAT SHEETS:

- https://ggplot2.tidyverse.org/reference/
- https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf (the most up-to-date one)
-
    - The RStudio website has a TON of helpful cheat sheets: https://rstudio.com/resources/cheatsheets/
        - Particularly useful ones for us today are the *RStudio IDE Cheat Sheet* and *Base R Cheat Sheet*

R Plotting Questions:

Copy of this lesson that's on the GitHub for day 2: https://github.com/UA-Carpentries-Workshops/2019-October-Resources/blob/master/r_tutorials/02b_plotting.R

- **What if we want to display a separate plot for each country (or continent, or other category...)?**
    - We can do this a few different ways.
        - The first option is to make ggplot give us a set of SUB-plots instead of one big plot:
            - We add the function **facet_wrap()** (*remember how to add a new layer?*)
                - **facet_wrap(~ continent)** means "create subplot plot by continent"
                    - the tilde (~) means "by"
                    - there is nothing in front of the tilde, because ggplot2 knows to apply facet_wrap to the plot you created in the preceding lines.
                        - When we run the whole thing, we get one big plot containing 5 separate subplots.

For-Loops and Conditionals:

What's the purpose of a For Loop? What's an example of an application?

- A for-loop is useful when you have some task you want to do over and over again, and you want to automate that process so you don't have to repeat it by hand.
    - Benefits: Less tedious for you, less prone to human error, more efficient.
        - For example:

        - I have 500 files in my directory, and I want to apply the same process to each file. But I really don't want to do it by hand 500 times...
            - I can write a for-loop where I tell the computer the file names to look for, and to apply that process to each file in turn.
- A for-loop can also be used to go through each unit of a dataset stepwise and apply a function to each one.
    - So we could create a for-loop that makes a separate plot of life expectancy (lifeExp) by GDP per capita (gdpPercap) for each continent in the dataset.
        - This would give us 5 separate, individual plots (one per continent) instead of a big plot with 5 subplots, as in the example above where we used **facet_wrap()**

Exercise:
make a loop to print the mean life expectancy by continent

Answers:

```
for(cont in unique(gapminder$continent)){
mean_life <- mean(gapminder[gapminder$continent == cont, "lifeExp"])
print(paste(cont, "has a mean life expectancy of", mean_life))
}
```

```
[1] "Asia has a average life expectancy of  60.0649032323232"
[1] "Europe has a average life expectancy of  71.9036861111111"
[1] "Africa has a average life expectancy of  48.8653301282051"
[1] "Americas has a average life expectancy of  64.6587366666667"
[1] "Oceania has a average life expectancy of  74.3262083333333"
for (cont in unique(gapminder$continent)){
mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
print( paste(cont, "has a mean life expectancy of", mean_life))
}
```

```
for (cont in unique(gapminder$continent)){
mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
print(paste(cont, "has a mean life expectancy of", mean_life))
}
```

```
for (cont in unique(gapminder$continent)){
mean_life <-  mean(gapminder[gapminder$continent == cont, 'lifeExp'])
print(paste(cont, 'has a mean life expectancy of', mean_life))
}
```

```
for(cont in unique(gapminder$continent)) {
mean_life <- mean(gapminder[gapminder$continent == cont, "lifeExp"])
print(paste(cont, "has a mean life expectancy of", mean_life))
}
```

```
for(cont in unique(gapminder$continent)){
mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
print(paste(cont, " has a mean life expectancy of ", mean_life))
}
```

```
for (cont in unique(gapminder$continent)){
 mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
 print(paste(cont,
 "has a mean life expectancy of",
 mean_life))
 }
```

```
for (cont in unique(gapminder$continent)){
 mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
 print(paste(cont, "has a mean life expectance of", format(round(mean_life, 1), nsmall=1),
'years.'))
 }
```

```
for (cont in unique(gapminder$continent)){
 mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
 if(mean_life > 59.47)
 print(paste(cont, "has a mean life expectancy higher than the global mean, and its mean is",
mean_life))
 }
```

```
unique(gapminder$continent) for (cont in unique(gapminder$continent)) {  mean_life <-
mean(gapminder[gapminder$continent == cont, 'lifeExp'])  print(paste(cont,
mean_life,              "has mean life expectancy"))}
```

What is a conditional?

- A conditional is an if or if-else statement that tells the computer to do something ONLY if something (for example, a value in a particular column) meets a particular condition.
    - If we're working with numerical values, we can use equal to, >, <, >= , or <=.
        - For example, we might want to create a new column in our dataframe indicating whether a participant's measurement is an outlier, based on whether or not their value on that measurement is greater than 50.
    - If we're working with strings, we can use evaluation:
        - is TRUE: ==
        - is FALSE: !=
            - For example, we might want to do something ONLY to rows of the dataframe where the value in the column `continent` is "Asia"

If/else conditionals follow the general format:
 For x in source_of_x, do SOMETHING if x meets a given condition;
 if x is anything (or something) else, do SOMETHINGELSE.

You can nest multiple if/else statements together and combine them for more complex conditionals.

What's the difference between "positive number" and 'zero' <---- "" versus '?
 Just preference (as far as I know!)

R doesn't distinguish between single and double quotation marks. I like to use double because they're a bit easier to see which makes your code more readable.
ty! :)

Let's find which continents have a mean lifeExp above 59.47!
Answers

```
for (cont in unique(gapminder$continent)){
 if (mean(gapminder[gapminder$continent == cont, 'lifeExp']) > mean_life_gapminder){
 print(paste(cont, 'has a mean life expectancy greater than the overall mean life expectancy.'))
 }
 }
```

alt:

```
 for (cont in unique(gapminder$continent)){
 mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
 if (mean_life > mean_life_gapminder){
 print(paste0(cont, ' has a mean life expectancy (', mean_life, ') greater than the overall mean life expectancy (', mean_life_gapminder, ').'))
 }
 }
```

```
for(cont in unique(gapminder$continent)){
 mean_life <- mean(gapminder[gapminder$continent == cont, 'lifeExp'])
 if (mean_life>mean_lifeExp_gapminder){
 print(paste(cont, 'higher mean life exp'))
 }
 }
```

Exercise
For 2007
Find which countries have a gdp at or below the 1st quartile
and which countries have a gdp at or above the 3rd quartile
and which countries have a median gdp (there may not be any)
Tip:

```
gapminder[gapminder$country == country & gapminder$year == 2007, 'gdpPercap']
```

By median here you mean between the quartiles??
just the raw median (== 6124.4 in this case)
thank you

I think third quartile is [5], not [4] when assinging values from summary
you are correct!

two_countries example: I tried == before %in% introducted and got the following, any reason why half the lines were cut?

```
> filter(gapminder,country ==two_countries)
country year     pop continent lifeExp gdpPercap
1           Albania 1952 1282697    Europe  55.230  1601.056
2           Albania 1962 1728137    Europe  64.820  2312.889
3           Albania 1972 2263554    Europe  67.690  3313.422
4           Albania 1982 2780097    Europe  70.420  3630.881
5           Albania 1992 3326498    Europe  71.581  2497.438
6           Albania 2002 3508512    Europe  75.651  4604.212
7  Dominican Republic 1957 2923186  Americas  49.828  1544.403
8  Dominican Republic 1967 4049146  Americas  56.751  1653.723
9  Dominican Republic 1977 5302800  Americas  61.788  2681.989
10 Dominican Republic 1987 6655297  Americas  66.046  2899.842
11 Dominican Republic 1997 7992357  Americas  69.957  3614.101
12 Dominican Republic 2007 9319622  Americas  72.235  6025.375
```

Adriana's answer: That's a weird output. This shouldn't have worked at all.

The year variable (second column) does not overlap between the two countries. So I'm guessing it finds the first instance of Albania, and then the first Instance of Dominican Republic in the next year (for some reason) and so on. It's an impredictable behaviour. If you switch the order in the vector, two_countries<- c("Dominican Republic", "Albania"), you will get the oppositve pattern (12 data rows totals still) of years across both countries.

Exercise

Write a single command (which can span multiple lines and includes pipes) that will produce a dataframe that has the African values for lifeExp, country and year, but not for other Continents. How many rows does your dataframe have and why?

Africa

lifeExp, country, year

624

624

624 --> data for year from 1952-2007

624

333 :(

gapminder %>% select(country, year, continent, lifeExp) %>% filter(continent == 'Africa') %>% select(-continent) %>% glimpse()

624 -> data for each country from 1952-2007

gapminder %>% filter(continent == "Africa") %>% select(lifeExp, country, year) %>% glimpse()

624

also 624

624

df <- data.frame(gapminder %>% select(-pop, -gdpPercap) %>% filter(continent == 'Africa'))

624 gapminder_just_Africa <- gapminder %>% filter(continent == 'Africa') %>% select (lifeExp, country, year)

gapminder %>%

```
 filter(continent == "Africa") %>%
 select(lifeExp, country, year) %>%
 nrow()
```

Challenge 2
 Create a new dataframe that contains the minimum (MinExp) and maximum (MaxExp) life expectancies for each country.

```
gapminder %>%
 select(country, lifeExp) %>%
 group_by(country) %>%
 summarize(min_life = min(lifeExp),
 max_life = max(lifeExp))

gapminder_minmax_life <- gapminder %>% group_by(country) %>% summarise(MinExp = min(lifeExp), MaxExp = max(lifeExp))

df2 <- data.frame(gapminder %>%
 group_by(country) %>% g
 summarise(MinLife = min(lifeExp),
 MaxLife = max(lifeExp)))
```

Challenge 3
 Calculate the average life expectancy per country. Which has the longest average life expectancy and which has the shortest average life expectancy?

shortest Sierra Leone, longest Iceland

```
gapminder %>% group_by(country) %>% summarise(MeanLife = mean(lifeExp)) %>% filter(MeanLife == min(MeanLife) | MeanLife == max(MeanLife))
 shortest and longest as above
```

Make a ggplot scatterplot of the lifeExp vs GDPperCap for the countries whose average life expectancy is lower than average for all years combined (mean_lifeExp_gapminder, or 59.47444). Color the graph by continent

steps:
 calculate mean_lifeExp_gapminder
 group_by
 summarize
 filter
 ggplot (?) + geom_point()

Not sure? tbh but here goes ??

- gapminder %>%
- group_by(country) %>%

```
    summarize(MeanLife = mean(lifeExp)) %>%
    filter(MeanLife > 59.4744) %>%
    ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp, color =
  continent)) +
    geom_point() + scale_x_log10()
  ^^ jk lol
  gapminder %>%
    filter(lifeExp < mean(lifeExp)) %>%
    group_by(country) %>%
    ggplot(mapping = aes(x = lifeExp, y = gdpPercap, color = continent)) +
      geom_point()
summarise(m
 meanGDPperCap = mean(g
 filter(mean_life_exp < all_mean) %>%
 ggplot(aes(x = mean_life_exp, y = meanGDPperCa
 color = continent)) +
 geom_point()
```