
Please sign in below with name, affiliation, something interesting you like to share and your github profile link

1. Annajiat Alim Rasel, Brac University, https://github.com/annajiat
2. Laurie Heller, CMU Psychology Dept, https://github.com/hellerl
3. Ben Ostrowski, CMU Organizational Behavior, https://github.com/benostrowski
4. Weibin Zeng, CMU Heinz College, https://github.com/WeibinZ
5. Mohamed Alimamy Fofanah, CMU Africa, https://github.com/mohamedfofanah
6. Ivy Liang, CMU Heinz College, https://github.com/ivyliang5
7. Hedy Zhang, CMU Heinz College, https://github.com/Hedyxddd
8. Ewuzie Chimauchem Paul, CMU Africa, https://github.com/PaulChimzy
9. Adamo Abass Tope, CMU Africa, https://github.com/iamtope
10. Eugene Mukeshimana, CMU Africa, https://github.com/emukesha
11. Michael Trujillo, CMU Psychology Dept., https://github.com/michael-trujillo
12. Poli Esther ,CMU Africa ,https://github.com/estherpolie
13. Sumeet Shinde, CMU Heinz college, https://github.com/sumeetshinde7
14. Saurabh Bhargava, CMU Social and Decision Sciences, https://github.com/williwaw44
15. Lauren Churilla, CMU Dietrich College of Humanities and Social Sciences, History https://github.com/Lchurill
16. Emma Slayton, CMU Libraries (Host)
17. Melanie Gainey, CMU Libraries (Host)
18. Hannah Gunderman, CMU Libraries (Helper)

19. Patrick Campbell, CMU Libraries (Helper), https://github.com/jaxgoodlabs
20. Agrippina Mwangi, CMU Africa Research Labs, https://github.com/grypy
21.https://github.com/cgatete
22. Percy Chukwudi, CMU Africa, https://github.com/Pchukwudi
23. https://github.com/seposso
24. Tanner Jones, Heinz College https://github.com/tannerjones2
25.Trina Washington CMU SEI https://github.com/pariscapri
26. Shameika Ansley, CMU Heinz College https://github.com/msannslee
27. Huajin Wang, CMU Libraries (helper)

Day 1: Shell and Git

Important links

- Course website: https://sanjayfuloria.github.io/2021-12-01-cmu-online/
- Today's lesson: https://swcarpentry.github.io/shell-novice/
    - Setup steps: https://swcarpentry.github.io/shell-novice/setup.html
- git for Windows: https://gitforwindows.org/
- Help documentation for Shell: https://explainshell.com/
- Stack Overflow (community supported tech support): https://stackoverflow.com/
- Collaboration with internal teams / private-like project: https://github.com/annajiat/cmu-internal-collaboration
- Collaboration with external teams / public project: https://github.com/annajiat/cmu-external-collaboration

Data visualization with ggplot2 cheatsheet
https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf

For Mac or PC open your terminal or command line and type 'BASH'

Code

Important keystrokes:
| - pipe
$ - dollar sign
^ - carrot
/ - forward slash
? - question mark
* - asterisk
tab - auto complete shortcut (if multiple names match the criteria, a list will appear for you to choose between)

Terminology: folder/directory (interchangeable)

Command Line Interface (CLI)
Graphical User Interface (GUI)

ls - list

https://swcarpentry.github.io/shell-novice/01-intro/index.html

https://www.go-fair.org/fair-principles/

https://swcarpentry.github.io/shell-novice/02-filedir/index.html

Home folder/directory which contains

Desktop
Downloads
Pictures
Video

Going to home dir
cd
cd ~

cd means change directory
pwd - print working directory

clear screen:
clear
CTRL + L

cd shell-lesson-data (change directory to location of lesson data)

classify list of files and folders
ls -F

ls -l

-F and -l are called 'flags' - these modify the behavior of the command
You can combine these

short flags
ls -l -h (human readable format)
ls -h -l
ls -lh
ls -hl

long flags

seeking help
Mac/linux:
man ls
info ls

windows
ls --help

https://explainshell.com/
https://explainshell.com/explain?cmd=+ls+-aF
https://stackoverflow.com/questions/tagged/shell
https://stackoverflow.com/questions/tagged/bash
https://stackoverflow.com/questions/tagged/python
https://stackoverflow.com/questions/tagged/git

https://stackoverflow.com/help/how-to-ask

https://codeblog.jonskeet.uk/2010/08/29/writing-the-perfect-question/

go back to previous/last dir
cd -

Speed check? please use a + sign below:
okay: ++++++++++++++
go-faster: ++
go-slower:

please explain in ..... way

To go three levels up

cd ../../..

Exercise: Absolute vs Relative Paths

https://swcarpentry.github.io/shell-novice/02-filedir/index.html#absolute-vs-relative-paths
Starting from /Users/amanda/data, which of the following commands could Amanda use to navigate to her home directory, which is /Users/amanda?

1. cd .
2. cd / ++
3. cd /home/amanda ++
4. cd ../..
5. cd ~ ++++
6. cd home ++
7. cd ~/data/..
8. cd +++
9. cd .. ++++++++++

ls -F
add "/" to end of dir names
add "*" to end of programs/executable
add "@" at the end of shortcuts/links

https://swcarpentry.github.io/shell-novice/03-create/index.html

mkdir (make a directory)

to check what is in thesis:

cd thesis/

To create directories:

-p project/data project/results

you can then use ls to check project directory

and then to check what is in project directory as ls -FR

To create a text file use:
touch draft.txt

To create a text file and open a text to edit it:
nano draft2.txt

Once GNU nano opens with the blank file named draft.txt
type publish

press CTRL+O to save
press ENTER/RETURN to confirm filename
press CTRL+X to exit GNU Nano

mv
to rename: (if used same location)
or to cut/paste the file   (if used diff location)

To rename our draft to qoutes:

mv draft.txt qoutes.txt

to move a file:
mv qoutes.txt ..

or

mv qoutes.txt ~/Downloads

To bring a file back to where it was:

ls ~/Downloadss/qoutes.txt

mv ~/Downloads/qoutes.txt .

to check if the file has moved use:

ls

To copy a file:

cp qoutes.txt thesis/qoutations.txt

To copy a directory:
```
$ cp -r thesis thesis_backup
```

CRTL+A to move to the beging of the line
CTRL+E to move to the end of the line

* is a **wildcard**, which matches zero or more characters.

ls *.txt can be used to show all .txt files

to find all files with a b in it;

ls *b*.txt

? -> one letter placeholder
* -> zero or more letters placeholder
can be used to find files with prefix/suffixe

To repeat a message or text use echo

echo hello returns hello

Collaboration with internal teams / private-like project:
https://github.com/annajiat/cmu-internal-collaboration

Collaboration with external teams / public project:
https://github.com/annajiat/cmu-external-collaboration

Once in GitHub, you can run a veriety of processes. If you need to follow these steps, please review the recording

https://git-scm.com/doc

Settings
Pages
source
main
save

https://github.com/**annajiat**/cv/settings/pages

https://annajiat.github.io/cv/
https://hellerl.github.io/cv/
https://michael-trujillo.github.io/cv/
https://benostrowski.github.io/cv/
https://ivyliang5.github.io/cv/
https://msannslee.github.io/CV/
https://pariscapri.github.io/cv/

https://pages.github.com/
https://choosealicense.com/
https://www.gitlab.com/

https://en.wikipedia.org/wiki/Newline#Representation

```
  659  git config --global user.name "Annajiat"
  660  git config --global user.email "annajiat@gmail.com"
  661  git config --global core.autocrlf true
  662  git config --global core.editor "nano -w"
  663  git config --global init.defaultBranch main
  664  git config --list
```

Keep notes about your project in a read me (description) file
README
README.TXT
README.MD

EDIT:

To edit a readme file, click the file, then click the blue pencil icon

in line 6, write your name, next line a description of your file

COMMIT CHANGES:

once you are satisfied with changes, scroll down to the bottom of the page and commit changes. when commiting changes, make sure to write a description of what was changed. Dont write a repeat of what you have changed, but explain why you made the change.

you can commit to a main file, or create a new branch of your file for version control. Creating a new branch is a good pratice (especaily when adding to others files). It also allows for individuals to create a file with their individual files. You can then merge branches between team members at a later date.

commit code:
COMMIT_MSG

ADD FILE:

To add a file click on the add file button within your main github project page. Then commit the new file o the main page. It is then important to create a record of your new file in the readme file.

CREATE A FOLDER:

Go to add file. In name file, create a directory name by adding a name to the name a file with a / afterward. Then it will ask you to add a file name. You can also use this process to add new files to an existing folder

INVITE MEMBERS

in setting, go to project setting, repository settings, manage acccess

From here you can search for people to add, or add people. You can add people through email, username, or full name.

Resources for continued learning:
Audit for free at https://www.edx.org/course/introduction-to-linux
nano editor:

- Long video: https://www.youtube.com/watch?v=L9Lc08Zb7cY
- Short video: https://www.youtube.com/watch?v=45KO4KO2DTo
- Tutorial: https://staffwww.fullcoll.edu/sedwards/Nano/IntroToNano.html

Video recording from another workshop:
Playlist: Unix Shell - Software Carpentry
https://www.youtube.com/playlist?list=PLu7d3po48tBobUSZ3dvFScMBD2oDp-CtQ

https://tldp.org/guides.html
http://copeland.ece.gatech.edu/jac/6612/info/learnUNIXin10minutes.html

```
mkdir  ~/.ssh
cd ~/.ssh
ssh-keygen -t ed25519 -C "annajiat@gmail.com"
press ENTER three times
cat  id_ed25519.pub
```

References:

Creating online CV / portfolio/project page / lab page

https://pages.github.com/

Other documentations:
https://git-scm.com/doc
https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_04B-3_Meli_paper.pdf

Recording from other workshops:
Split: https://www.youtube.com/playlist?list=PLn8I4rGvUPf6qxv2KRN_wK7inXHJH6AIJ
Medium: https://www.youtube.com/watch?v=fTRtzsYo7Ho
Long: https://www.youtube.com/watch?v=evaOmk10wrw

Learn by exploring Agnes' repository
https://agnescameron.info/
https://github.com/agnescameron/agnescameron.info

I had trouble joining the tuesday meeting -- the zoom link from email does not seem to be working - but the zoom link from the Evite email worked.

Day 2: Python

Important links

- Zoom link: https://cmu.zoom.us/j/96074015548?pwd=TnpsM1FZK2ZLVzNwUEZ5WHZsZis3QT09
    - ID: 96074015548
    - Passcode: w8j448H8
- Course website: https://sanjayfuloria.github.io/2021-12-01-cmu-online/
- Today's lesson: https://datacarpentry.org/python-ecology-lesson/
- Structuring Your Project (guide): https://docs.python-guide.org/writing/structure/

Start by opening Jupiter Notebook

Code

pwd
import pandas as pd
import os

If you need to change your working directory, use:
os.chdir("insert directory location here, e.g., Users/yourname/Desktop/")

surveys = pd.read_csv("surveys.csv")

When looking for help you can look at the help menu, which you can find under the help tab in the user interface. You can also search for specific topics in help as well.

To open Python from terminal:
Windows: Start Menu > type cmd and open command prompt.
Mac:
Three arrows (>>>) means you're in Python

print("Hello world")
# Data types:
text = "Data Carpentry"
number = 42
pi_value= 3.1415

# Prints values of variables:

```
text
number
pi_value

#Check type (or class) of variable
type(text)
type(number)
type(pi_value)

13 % 5 # division, returns the remainder of the operation

3 > 4 #logical test, returns boolean (true/false)

# Lists
# Python indexes start with 0 instead of 1
numbers = [1,2,3]

numbers[0] # prints the first integer in list
numbers[1] # prints 2nd integer, etc.

#For loop
# this function iterates through the items in the list and prints them one by one
for num in numbers:
    print (num) # need indentation on this line

numbers.append(4) # adds one more integer (4) to the list
numbers

for num in numbers:
    print (num) # need indentation on this line

# lists use square brackets [], tuples use parenthes ()
a_tuple = (1,2,3) #tuples are immutable data types
another_tuple = ('blue','green','red')

type(a_tuple)
type(numbers)

a_list[1] #returns error
a_list[1] = 5 #returns error - 'a_list' not defined

a_tuple[2] = 5 #returns error - cannot change items in tuple

numbers[1] = 4 #changes the value of the integer in the second position
numbers
```

Tuples are generally used for different Python Data Types; whereas, lists are used for similar data types.

```
translation = {'one': 'first', 'two': 'second'} #dictionaries hold pairs of values, keys and variables
```

{key:variable}
translation['one']
returns 'first'
translation['two']
returns 'second'
rev = {'first ': 'one', 'second' : 'two'}
rev['first']
returns 'one'

for key, value in rev.items():

- print(key, '->', value)

returns first -> one
second -> two

for key in rev.keys(): #another way to do it

- print(key, '->', rev[key])

returns first -> one
second -> two

z = add_function(20,22) # to asign values
print(z)
returns 42

z= add_function(100,230) #to change what values are assigned
print(z)
returns 330

#opening text editor, you can also use it as a place to store code as a .py file. This may work best for macs, or not using notepad but using nano instead. You may be able to get around this by saving it as a plane text.

#we can then acccess this file in pyhton terminal code

#to access just type python and then the name of the file:

python example.py

Using Pandas in Jupyter Notebooks

import pandas as pd

#press run or command+enter or control+enter to run code in Jupyter Notebooks

pd.read_csv("surveys.csv")

returns the dataset in a table

#you may also need to important something called os to change directories in order too woork with the files

#unless you chnage a working directory it will remain the same. If you open a file in terminal, it will continue to function in that terminal.

#to save your surveys.csv in a file instead of only printing it

surveys_df = pd.read_csv('surveys.csv")

#to see the first 5 lines

surveys_df.head()

type(surveys_df)

surveys_df.types #tells you information about the variable in the file

surveys_df.columns #tells you infoo about the collumns

pd.unique(surveys_df['species_id']) #will tell you unique names of info within variable types

Exercise:

Create a list of unique site ID’s (“plot_id”) found in the surveys data. Call it site_names. How many unique sites are there in the data? How many unique species are in the data?

site_names = pd.unique(surveys_df['plot_id'])
lens(site_names) #will tell you how many items are in a field

surveys_df['weight'].describe() #will tell you count, median, mean, etc.

surveys_df['weight'].min()
returns 4.0

surveys_df['hindfoot_length'].min()
returns 2.0

grouped_data = surveys_df.groupby('sex')

grouped_data.describe()
group_data.mean()

## When you use ## in jupiter note books it allows for a file to be shown as a markdown, or as text that can be used as a header or context in your file

#when typing with * it allows you to return a list. You must also change in the drop down for where it says code to markdown

- *Data types and formats of data

- *Index, slice, and subset Data frames
- *combine dataFrames and using pandass.

#load the surveys as csv and call the object too vieew it
surveys_df = pd.read_csv("surveys.csv")
surveys_df

#tail () funtion lists the last rows
surveys_df.tail()

#you can add the number in () to specifcy which rows
surveys_df.tail(7)

#can we extract a colloumn that has years
surveys_df['year']

type(surveys_df.year)
returns pandas.core.frame.DataFrame

Indexing and Slicing

a= [2,4,6,8,10,12,14,16,18,20] #List of even numbers between 2 and 20
a
returns [2,4,6,8,10,12,14,16,18,20]

len(a) #checks the length of a list
returns 10

a[0]
returns 2

a[6]
returns 14

a[-1]
returns 20

a[-4]
returns 14

surveys_df.head(10) 3calling the first 10 rows from the survey data set

#to extract data from row 2 to row 6, extracting month day and year data,what would be do?

#row2data, need to request both for rows and columns, so below we specify the row and then indicate a blank sspace for the collumns as we want all of them for that row.

Row2 = surveys_df[1:]

Row8 = surveys_df[8:9] #rows, columns
Row8

# Range, always excludes the last value. Example if you want to capture 0 to 5
Row = surveys_df[0:5] #rows, collumns
# Rows 0 to 4 will show, but not roow 5

Row = surveys_df [:5] #start, end, but havign rows empty will also indicate that you want rows up to a specific value

#Q3 to extract data from row 2 to row 6, extracting month day and year data,what would be do?
surveys_df.iloc[2,7] # returns a specific value, but not the range

syrveys_df.loc[0,['month,'day','year']]

#to pull all records for subsect 'month,'day','year', use the following :
surveys_df.loc[:,['month,'day','year']]

surveys_df.loc[[2,3,4,5,6],['month,'day','year']] #rows (2-6), columns (m,d,y)

surveys_df.loc[2:6,['month,'day','year']] # will get you a range, unlike the specific values above

#Data from row 2,4,6,800,35400
surveys_df.loc[[2,4,6,800,35400],['month,'day','year']]

#excersise : What happens when you execute the following
Type your name and a solution below (if you see one you agree with, use a +):

Surveys_df[0:1]
 first row of data +1+1+++

Surveys_df[:4]
first 4 rows and all columns
first 4 rows of data++++

Surveys_df[:-1]
all but last row of data+++
 and all column data

Results:

Surveys_def[0:1] #will return the first row only

Surveys_df[:4] #will return rows up to 4, so rows 0 to 3

Surveys_df[:-1] returns all but the last row

#Excersise What happens when you execute the following
Type your name and a solution below (if you see one you agree with, use a +):

Surveys_df.ilooc [0:4, 1:4]
gives month, day, year for the first 4 rows++
first four rows, columns one up to four (3)

Surveys_df.loc[0:4, 1:4]
error because there isnt any label named 1,2,3,4+++

#Q4: Can we get row data for the year 2002
surveys_df[surveys_df.year == 2002] # the bit between [ ] is a boolean equation

## Q5: Records for the year between 1992 and 2000
#Hint >=   and <=
surveys_df[(surveys_df.year >=1992) & (surveys_df.year <=2000)]

Day 3: Python (cont.) and Version Control with Git

Important Links

- Zoom link:
- Course website: https://sanjayfuloria.github.io/2021-12-01-cmu-online/
- Today's lesson: https://swcarpentry.github.io/git-novice/
- Resources:
    - Data visualization with ggplot2 cheatsheet https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf

Code

grouped_data.describe()
grouped_data.mean()

surveys_df['weight'].mean()

len(surveys_df[pd.isnull(surveys_df.weight)]) # returns number of rows with null values in the weight column

len(surveys_df[surveys_df.weight > 0])

df1 = surveys_df.copy()

```
df1['weight']= df1['weight'].fillna(0) # replaces null values in weight column with 0

df1['weight'].mean()

df1['weight'] = surveys_df['weight'].fillna(surveys_df['weight'].mean()) # replaces NA values in weight column with the mean

df1['weight'].mean()

surveys_df = pd.read_csv("surveys.csv")
df_na = surveys_df.dropna()
df_na

df_na.to_csv('surveys_complete.csv', index = False) #Writes data to csv file and saves to working directory

surveys_df = pd.read_csv("surveys.csv", keep_default_na = False, na_values =[""])
surveys_df

#If you haven't already, need to download species dataset from lesson's setup page

species_df = pd.read_csv("species.csv", keep_default_na = False, na_values = [""])
species_df

surveys_sub = surveys_df.head(10)
surveys_sub_last10 = surveys_df.tail(10)

surveys_sub_last10 = surveys_sub_last10.reset_index(drop = True)
vertical_stack = pd.concat([surveys_sub, surveys_sub_last10], axis = 0)
vertical_stack

vertical_stack.reset.indexing(drop = True) # resets the index count for the dataset

vertical_stack.to_csv("out.csv", index = False)

horizontal_stack = pd.concat([surveys_sub, surveys_sub_last10], axis = 1)
hoizontal_stack

survey_sub = surveys_df.head(10)
species_sub = pd.read_cs("speciesSubset.csv", keep_default_na = False, na_values = [""])

merge_inner = pd.merge(left = survey_sub, right = species_sub, left_on = 'species_id', right_on = 'species_id')
merge_inner.shape

merge_inner

pd.merge(product, customer, left_on = 'Product_name', right_on = 'Purchased_Product') #inner join
```

merged_left = pd.merge(left = survey_sub, right = species_sub, how = 'left', left_on = 'species_id', right_on = 'species_id') # left join

merged_left

merged_left [pd.isnull(merge_left.genus)] #finds null values in the meerge

animals = ['lion, crocodile', 'vulture', 'hippo']
print (animals)

for creature in animals #this will print the list of animals as long as the for and the print () namee is the same

- print (creature)

import os

os.mkdir('yearly_files') #will create a new directory within our main working directory

os.listdir() #will produce a list of files in youur direcctory

surveys_pdf = pd.reac_csv("surveys.csv")
surveys2002 = surveys_df[surveys_df.year == 2002]
surveys2002.to_csv('yearly_files/surveys2002.csv')

surveys_df['year'].unqiue()

for year in surveys_df['year'].unique():

- filename = 'yearly_files/surveys' + str(year) + '.csv'
- print(filename)

or year in surveys_df['year'].unique()

- surveys_year = surveys_df[surveys_df.year == year]
- filename = 'yearly_files/surveys' + str(year) + '.csv'
- surveys_year.to_csv(filename)

### data visulization

#ilibraries
import pandas as pd
import plotnine as p9

#load the data
surveys_df = pd.read_csv('surveys.csv')
surveys_df.head()

#clean data of not a numer results or nulls
surveys_df = surveys_df.fillna(0)

#plotnine
p9.ggplot(data = surveyss_df, mapping = p9.aes(x = 'weeight', y = 'hindfoot_length'))

#geom option
p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length'))+p9.geom_point()

Data visualization with ggplot2 cheatsheet
https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdfData visualization with ggplot2 cheatsheet https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf

#add color

p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length'))
+p9.geom_point(collor='blue')

#Q1: in hindfoot_length against weight, can you spot different plot ids?

p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length', color = 'plot_id')) +
p9.geom_point(alpha = 0.1)

#Q2: Can we have ore descriptive lables for x and y?

```
(p9.ggplot(data_surveys_df,
   mapping = p9.aes(x = 'weight', y='heindfoot_ength', color = 'plot_id'))
   + p9.geom_point(alpha = 0.1)
   +p9(xlab("Weight(lbs)")
   +p9.ylab("Hindfoot_length(ft)")
   +p9.scale_x_log10()
   +p9.theme_background_bw()
   +p9.theme(text = p9.element_text(size = 16))
)
#Q3: Can we plot the plot id and the year surveyed?
#Let's get boxplots
(p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'year', y = 'plot_id'))
   +p9.geom_boxplot()
)
```

#Exercise
p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length', color = 'plot_id')) +
p9.geom_point(alpha = 0.1)

#Represent the data using geom_violin
p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length', color = 'plot_id')) +
p9.geom_violin(alpha = 0.1)

## Plotting time series data

```
yearly_counts = surveys_df.groupby(['year', 'species_id'])['species_id'].count() #counts
yearly_counts = yearly_counts.reset_index(name = 'counts') #convert back to dataframe
yearly_counts
```

```
p9.ggplot(data = yearly_counts, mapping = p9.aes(x = 'year', y = 'counts'))
   +p9.geom_line()

(p9.ggplot(data = surveys_df, mapping = p9.aes(x = 'weight', y = 'hindfoot_length', color = 'plot_id'))
   +p9.geom_line(alpha = 0.1))

##Pyplot
#Import libraries first
import matplotlib.pyplot as plt

surveys_df.head()
mp_plot = surveys_df.plot("hindfoot_length", "weight"), kind = "scatter") # x, y
plt.show()

#normal distribution
import numpy as np

#mean = 0
#standard deviation = 0.1
#sample = 1000
sample_data = np.random.normal(0, 0.1, 1000) #(mean, std, samples)
sample_data #yields different set of random values every time you run

#We want to create a histogram
plt.hist(sample_data)
plt.show() #prints figure only
```

LINK TO SURVEY:
Please complete this survey when you can: https://carpentries.typeform.com/to/UgVdRQ?slug=2021-12-01-cmu-online