MIT 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 https://etherpad.wikimedia.org).

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

---

Links & Resources

Welcome to the Summer 2022 Carpentries Workshop at MIT. We will use this Etherpad for discussion, shared note-taking, quick practice questions, and anything else that comes up.

General Setup & Tasks

- Workshop website: https://carpentries-mit.github.io/2022-08-02-mit-online/
- Pre-Workshop Survey: https://carpentries.typeform.com/to/wi32rS?slug=2022-08-02-mit-online (Please take the time and fill this out if you haven't already done so)
- Welcome/Setup slides: http://bit.ly/SCW_MIT_setup
- Please fill in your name in the upper right corner of the Etherpad (next to the color square) and then **sign in under the Attendee section and tell us a bit about yourself.**
- **No question is stupid**. Use the Chat window to the right to ask questions anytime.
- Zoom Connection details:
    - Join through the previously-installed Zoom client application on your computer instead of using the link to 'join from your browser'
    - Find the Zoom connection details in your registration confirmation email
    - type your chosen name, pronouns, and/or pronunciation guide into the participant window
    - keep your video on if bandwidth & situation allows
    - wear headphones (if you have them) to hear better
    - please mute yourself to start

Lesson materials

- Unix Shell Lesson: https://swcarpentry.github.io/shell-novice/
- Unix Shell Slides: http://bit.ly/SCW_MIT_unix
- Unix Shell Data: http://swcarpentry.github.io/shell-novice/data/shell-lesson-data.zip
- Python Lesson: http://swcarpentry.github.io/python-novice-gapminder/
- Python Slides: Slide 33 - 36 of http://bit.ly/SCW_MIT_setup

- Git/GitHub Lesson: https://carpentries-mit.github.io/githublesson/notebooks.html
- Open Science Slides: https://bit.ly/SWC-OpenSci

Feedback links

- **Daily feedback**. At the end of each day, use this Jamboard to provide feedback on the day's session: https://bit.ly/dayfeedback-202208
- **Post-workshop Survey.** Please take the time to fill out the post-workshop survey at the end of all classes (after Aug 4): https://carpentries.typeform.com/to/UgVdRQ?slug=2022-08-02-mit-online

Attendees

Add your Name, Department/Affiliation, email, twitter, github username or anything you'd like to share about yourself with the group.

- Ye Li, MIT Libraries, yel@mit.edu, @yelibrarian, GitHub: YeLibrarian
- Christine Malinowski, cmalin@mit.edu, @ChrisMalinow, GitHub: cmalinowski3
- Yonghong Chen, the Children Hospital of Philadelphia, GitHub: Yonghong2014
- Ece Turnator, turnator@mit.edu, GitHub: eturnator
- Phoebe Ayers, MIT Libraries, psayers@mit.edu, GitHub: phoebeayers
- Catherine Clark (she/her), MIT History, clarkce@mit.edu, GitHug: clarkce
- Hannah Toru (She/her), Department of Chemistry/MIT, hannahts@mit.edu, Twitter @hshaytoru, Github: astrochemhannah
- Kimaya Suryarao, kimayasuryarao3@gmail.com, GitHub: kimaya73
- Alex Byrne (He/Him), Department of Chemistry at MIT, lxbyrne@mit.edu, git: lxbyrne
- Alejandro Paz (he/him), MIT Libraries, apaz@mit.edu, Github: alevan2p
- Kristin Bergmann (she/her), EAPS, MIT, kdberg@mit.edu, Github: kbergmann
- Ayse Eren (she/her), Chemical Engineering, MIT, ayseeren@mit.edu, Git: erenayse
- Shan Zhou(She/her/hers), Department of Social Sciences, Michigan Tech, shanzhou@mtu.edu, github username: szhou2022.
- Smith/Wun-Long Jheng (He/his/him), Medical 3D Printing Center, National Defense Medical Center, wuj102@g.harvard.edu/ school111360@gmail.com, GitHub: MDoid111
- Guangqi Wu, Chemical Engineering, guangqiw@mit.edu, Github: wuRoy
- Ceci Xue, Department of Chemistry, cixue@mit.edu, Github: cixue
- Tingting Sui (She/Her), Department of Mechanical Engineering, tsui@mun.ca, Github: SY1515305
- Seung Kyun "SK" Ha (he/him), Dept. of Chem. Eng, MIT, skha89@mit.edu, Github: skha89
- Andrew Harris (he/him), School of Engineering, University of Guelph, harrisa@uoguelph.ca, Github: aharris953
- Ding Jiahe,ding_jiahe@icloud.com,Github:Ding-Jiahe
- Yifan Wang, Architecture and Interdisciplinary, email: zcftaaf@ucl.ac.uk, Github: Eavan-Wang
- Daniel Sheehan, MIT Libraries, dsheehan@mit.edu
- Yuren Jin (he/him), Creative Tech. | Blockchain, EM Lyon Business School, email: yuren.jin@edu.em-lyon.com, Github: YurenJeremyJin
- Song Xue, email: xuesongdsg@outlook.com, Github: AlvusXS
- Yan Chen, ychen70@gmu.edu, Github: ychenfolio

- Barbara Williams, MIT Libraries, barbaraw@mit.edu, Github: BarbaraBW
- Misong Ju, DMSE, MIT, misongju@mit.edu, Github: misongju
- Cristian Cavedon, Department of Chemistry MIT, cavedon@mit.edu, Github: cavedonc
- Jessica Sashihara, J-PAL North America, MIT, jsashihara@povertyactionlab.org, github: Jessica7auren
- Yi Chien Lee, yician.lee@gmail.com, github: amandayclee
- Yin Pan, email: panyin99@gmail.com, github: panyin99
- Yuxuan LIU, email: liuyuxuan0611@gmail.com, github: LIU1999
- Mike B., MITx MicroMasters - Statistics and Data Science, GitHub: xbonafi7x
- Wei Yun, email: 15666134216@163.com, github: soniakim333
- Ana Mendes, artist, ana@anamendes.com, Goldsmiths College, GitHub: anamendesana

Please share your GitHub.com user name if you haven't.

Yun Wei

Xingni Shi

Day 1: The Unix Shell (2022-08-02)

Setup

- Install bash shell: instructions found on course website, https://carpentries-mit.github.io/2022-08-02-mit-online/
- Download data (and move it to your desktop): http://swcarpentry.github.io/shell-novice/data/shell-lesson-data.zip
- Open your bash shell
    - Mac: open the built-in terminal via Applications/Utilities/Terminal
    - PC: open terminal by running Git Bash program from Windows start menu
    - Linux: open terminal
- Open these browser tabs:
    - Etherpad: this window
    - (optional) Unix lesson: http://swcarpentrgithub.io/shell-novice

Nano text editor troubleshooting (info included here in case issues arise during lesson - some WINDOWS users have an issue getting nano to work):

- The issue and workaround are explained here: *https://www.scivision.dev/git-for-windows-nano-error-fix/*
- Short version of solution to try: Open Git Bash using “Run as Administrator” via the Windows Start menu and type into this Terminal: *dos2unix /usr/share/nano/git.nanorc

Group notes

Link to the presentation slide: http://bit.ly/SCW_MIT_unix
Unix Shell is a command line interface and a scripting language.
Why use Shell?
- enables repetitive tasks
- Combine tools
- interact with remot machines
- enhance reproducibility

echo $SHELL --> show what types of Shell you are in.

for zsh people chsh -s /bin/bash which sets up your default as bash (you can undo this afterwords. You might have to close and reopen the terminal.)
If the commandline change doesn't work for you, you may try to change it under System Preference as mentioned on the bottom of this page https://www.howtogeek.com/444596/how-to-change-the-default-shell-to-bash-in-macos-catalina/ .
chsh -s/bin/zsh switch to zsh from bash

- Bash vs Z shell: A Tale of Two Command Line Shells
https://medium.com/@harrison.miller13_28580/bash-vs-z-shell-a-tale-of-two-command-line-shells-c65bb66e4658
- What are the practical differences between Bash and Zsh?
https://apple.stackexchange.com/questions/361870/what-are-the-practical-differences-between-bash-and-zsh

changing the prompt on terminal (temporary, per session): export PS1="desired name$"
\w = working directory, \u = username

Info on all the many bash prompt options:
https://phoenixnap.com/kb/change-bash-prompt-linux
https://medium.com/macoclock/how-to-change-the-colour-of-your-bash-prompt-on-mac-b06032543353p

changing color: export PS1="\[color???]sessionname$ \[\color]"

directory is same as folder

pwd = print working directory
cd= change dir
ls= list contents of your current dir

ls -F : flags your listing; foldername/ indicates folder
ls -F NAME (Use tab to autocomplete)

ctrl+A to get to beginning of command prompt; ctrl+E to get back to end

Cheatsheet for commands https://swcarpentry.github.io/shell-novice/reference

Get help:
Mac - man ls --> get help ('manual') for the command ls . Type q to close
Windows - ls --help

clear --> clean up the screen

cd NAME -> go down one level
cd .. -> go up one level
cd --> back home
cd - --> prev directory
tab --> auto-completion

Using the filesystem diagram on the slide, if pwd displays /Users/thing, what will ls -F ../backup display?

1. ../backup: No such file or directory
2. 2012-12-01 2013-01-08 2013-01-27
3. 2012-12-01/ 2013-01-08/ 2013-01-27/
4. **original/ pnas_final/ pnas_sub**/ (correct answer)

ls will list files and folders if you want folders flagged, use -F

- can list multiple locations: ls . folder/folder2 (this lists what's in current folder and in folder/folder2)

mkdir make dir will create a folder in your working dir

mkdir -p folder/folder/folder/folder : -p lets you make all at once

ls -R folder: lets you see everything in folders

Resource: https://libraries.mit.edu/data-management/store/organize/ pwd

nano draft.txt --> create a txt file named draft and open it in the text editor nano
In nano, ctrlO to save and ctrlX to exit

touch my_file.txt: creating an empty txt file (but not open it)

ls -l : file info including length(size)

ls . thesis: list what is in current folder and in thesis folder

mv filepath(or filename) filepath(filename): move the file (e.g. mv thesis/draft.txt thesis/quotes.txt) (mv thesis/draft.txt .) '.' means current directory
mv filename1 filename2: rename the file.

copy files:
cp dir/file1 dir2/file2
cp -r dir1 dir2 ('-r' means grab everything in dir1)/????'

EXERCISE: Suppose that you created a plain-text file in your current direcdtory to contain a list of the statistical tests you will need to do to analyze your data, and named it: statstics.txt
After creating and saving this file you realize you misspelled the filename! You want to correct the mistake, which of the following commands could you use to do so?
1 cp statstics.txt statistics.txt
2 mv statstics.txt statistics.txt (correct answer)
3 mv statstics.txt .
4 cp statstics.txt .

rm filename: remove the file
rm -r -i dir: remove the dir (-r: recursive, -i: interactive)

-i: "are you sure"

echo remove -r thesis --> tells you what files and folders will be removed

* matches zero or more characters. ? matches only one character
When run in the proteins directory, which ls command(s) will produce this output?
ethane.pdb methane.pdb
1 ls *t*ane.pdb
2 ls *t?ne.*
3 ls *t??ne.pdb
4 ls ethane.*

$wc cubane.pdb
    20   156   1158 cubane.pdb --> lines words   charac count
    wc -l = justr lines
    wc - w= just words
    wc -m = #chrc

ctrl+c = bailout when stuck

wc -l *.pdb > lengths.txt   ('>': save the content as a txt file) ('>>' append instead of overwriting)
cat lengths.txt: view contents of the file
less lengths.txt: show a file's content one screen at a time
q   --> get out of a dead loop

sort -n lengths.txt --> sort the content in the file numerically ('-n' means numerical)
head -n 1 sorted_lengths.txt --> display the first line ('-n' means # lines)

Sorting CSV - make sure your data is tidy (https://towardsdatascience.com/what-is-tidy-data-d58bb9ad2458?gi=cc5a47e68d21)

command 1 | command 2 | command 3: command 2 uses the output of command 1 as input, then output to command 3 as command 3's input (e.g. wc -l *.txt | sort -n | head -n 5, wc -l *.txt | sort -n | tail -n 5)

ls *[AB].txt shows txt files with their names ending with 'A or B'

$for filename in *.dat
> do head -n 2 $filename | tail -n 1
> done
CLASSIFICATION: basiliscus vulgaris
CLASSIFICATION: bos hominus
CLASSIFICATION: equus monoceros
for every file that is in *.dat within the directory, take first two line of file and take the last line of that specified two lines, for loop:
To do multiple tasks in the loop:
> do
> task1 (e.g. echo $filename)
> task2s
> task3
...
>done

less all.pdb --> less can preview a file in bash

Overwrite vs append in a loop:
for alkanes in *.dat
> do
> echo $filename
> cat $filename >> alkanes.pdb ('>' overwrites alkanes.pdb and '>>' appends entries to a single alkanes.pdb file)
> done

Nested loop:
for date in 20220802 20220803 20220804
do
for temp in 25 30 37 40
do mkdir $date-$temp
done
done

Save previous commands into a shell script file:
history | tail -n 5 > cleanData.sh (save previous 5 commands in history into cleanData.sh file)
cleanData.sh file can be edited by using nano

To run a shell script:
bash FILE.sh

Day 2: Python (2022-08-02)

Setup

- Install Anaconda with Python 3 : https://carpentriesmit.github.io/2022-08-02-mit/
- Download and unzip the linked data file http://swcarpentry.github.io/python-novice-gapminder/files/python-novice-gapminder-data.zip
- Start Anaconda Navigator
    - Mac: Launchpad > Anaconda-Navigator
    - Windows: Start > Anaconda Navigator
    - Linux: open terminal and type anaconda-navigator
- Start Jupyter Lab on the Anaconda Navigator window in your browser
- Open these browser tabs:
    - Etherpad: https://pad.carpentries.org/2022-08-02-mit Python lesson: http://swcarpentry.github.io/python-novice-gapminder/

Group Notes:

Python Part 1 ipnb shared:
https://drive.google.com/file/d/1XA9EwBzFNG9bOZO0EhcT696ksGSIW_oa/view?usp=sharing

shift + enter to run code block
Switch to markdown from code by 'esc' and then 'm' in a code cell
Markdown headings: start with # and/or ##
Markdown bullets: start with *
Markdown numbered list: start with 1., 2., etc.

Summary of shortcut with cells
Change cell mode
Esc + m --> switch to markdown mode
Esc + y --> switch to code mode

Esc + b --> add a cell below the current cell
Esc + a --> add a cell above the current cell
Esc + x --> delete the current cell
Esc + z --> undo the last operation.

Cheat sheet for markdown:
https://www.markdownguide.org/cheat-sheet/

Code for attaching an image in markdown:
![][giraffe]

[giraffe]:http://upload.wikimedia.org/wikipedia/commons/thumb/b/b7/

South_African_Giraffe,_head.jpg/877px-South_African_Giraffe,_head.jpg "Picture of a Giraffe"

Python rules:

A variable name must start with a letter or the underscore character
A variable name cannot start with a number
A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
Variable names are case-sensitive (age, Age and AGE are three different variables)

index starts from 0;
len(name): the lenghth of the name
name[index]:the element on the index position;
name[0:4]:the element from 0-3(0,1,2,3)
space between two words also worth one index

EXAMPLE: Fill the table showing the values of the variables in this program after each statement is executed.
element
# Command # Value of x # Value of y # Value of swap #
x = 1.0 # # # #
y = 3.0 # # # #
swap = x # # # #
x = y # # # #
y = swap # # # #
Answer key: x = 3.0, y = 1.0, swap = 1.0

EXAMPLE: Given the following string:

!! Do any 3 of the below:

species_name = "Acacia buxifolia"
What would these expressions return?

species_name[2:8]
species_name[11:] (without a value after the colon)
species_name[:4] (without a value before the colon)
species_name[:] (just a colon)
species_name[11:-3]
species_name[-5:-3]
What happens when you choose a stop value which is out of range? (i.e., try species_name[0:20] or species_name[:103])

Answer key:

1. [2:8] returns the substring 'acia b'
2. species_name[11:] returns the substring 'folia', from position 11 until the end
3. species_name[:4] returns the substring 'Acac', from the start up to but not including position 4
4. species_name[:] returns the entire string 'Acacia buxifolia'

5. species_name[11:-3] returns the substring 'fo', from the 11th position to the third last position
6. species_name[-5:-3] also returns the substring 'fo', from the fifth last position to the third last
7. If a part of the slice is out of range, the operation does not fail. species_name[0:20] gives the same result as species_name[0:], and species_name[:103] gives the same result as species_name[:]

type() --> show the data type of a variable

addition "+" cannot be applied to different types of data. But can apply to numeric data as addition and to strings as concatenate.

What type of value (integer, floating point number, or character string) would you use to represent each of the following? Try to come up with more than one good answer for each problem. For example, in # 1, when would counting days with a floating point variable make more sense than using an integer?

Number of days since the start of the year.
Time elapsed from the start of the year until now in days.
Serial number of a piece of lab equipment.
A lab specimen's age
Current population of a city.
Average population of a city over time.

Arithmetic with Different Types
Which of the following will return the floating point number 2.0? Note: there may be more than one right answer.

first = 1.0
second = "1"
third = "1.1"

first + float(second)
float(second) + float(third)
first + int(third)
first + int(float(third))
int(first) + int(float(third))
2.0 * second

shift + tab: read docstring

Difference between functions and method - a couple of useful posts
https://www.geeksforgeeks.org/difference-method-function-python/
https://pythongeeks.org/python-methods-vs-functions/
Function -

- can be called by name.
- Can be EXPLICITLY passed data to operate on and
- can optionally return data

Method -
- Called by a name that is associated with an object
- method is implicitly passed the object on which it was called
- a method is able to operate on data that is contained within the class. (an object is an instance of a class)

Predict what each of the print statements in the program below will print.

Does max(len(rich), poor) run or produce an error message? If it runs, does its result make any sense?

```
easy_string = "abc"
print(max(easy_string))
rich = "gold"
poor = "tin"
print(max(rich, poor))
print(max(len(rich), len(poor)))
```

ASCII number of the chaacter is returned when the iterable is a string

Jigsaw Puzzle (Parson's Problem) Programming Example
Rearrange the following statements so that a random DNA base is printed and its index in the string. Not all statements may be needed. Feel free to use/add intermediate variables.

```
bases="ACTTGCTTGAC"
import math
import random
___ = random.randrange(n_bases)
___ = len(bases)
print("random base ", bases[___], "base index", ___)
```

Link to download data if you haven't http://swcarpentry.github.io/python-novice-gapminder/files/python-novice-gapminder-data.zip Please unzip them under the folder you created for practice.

import pandas as pd --> import pandas library

https://www.py4e.com/lessons (for a whole lesson on dictionaries).
More about Data Structures in general, including Dictionary as a data type.
https://docs.python.org/3/tutorial/datastructures.html

pd.read_csv ('data/XXX.csv') --> read the CSV file into the workspace.
If you cannot import the data file, please check if your present working directory (using "pwd") is the directory one level up your data folder.

data.info() --> show basic info of the data frame

data.describe() --> show statistical summary of the data frame

data.to_csv --> save the data frame into a csv file.

Aggregate function explained:

aggregate and axes explained: https://www.dataday.life/notes/data-engineering-python/difference-between-axis-0-axis-1-python-pandas/

Fill in the blanks below to plot the minimum GDP per capita over time for all the countries in Europe. Modify it again to plot the maximum GDP per capita over time for Europe.

europe = pd.read_csv('data/gapminder_gdp_europe.csv', index_col='country')

plt.plot(europe.___, label=__)
plt.legend('upper left')
plt.xticks(rotation= __)

Plt.Scatter example:
Documentation: https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html

data_all = pd.read_csv('data/gapminder_all.csv', index_col='country')
plt.scatter(x=data_all['gdpPercap_2007'], y=data_all['lifeExp_2007'], s=data_all['pop_2007']/1e6)

10 to the power of 6, which is 1 million

s= size in points**2 meaning the area of the size of the marker is times 2 larger. Countries with higher pop in 2007 have larger bubble

LISTS:

del is not a function (but functions like one) but a statement in the language.

Exercise:

Fill in the blanks so that the program produces the output shown. Remember the tools that we have looked at that manipulate lists.

values = ____
values.____(1)

Output:
first time: [1, 3, 5]
second time: [3, 5]

Show variables in Jupyter Notebooks.

- dir() will give you the list of built-in scope variables for an object
- globals() will give you a dictionary of global variables
- locals() will give you a dictionary of local variables

More about global, local, and built-in scope.
https://www.geeksforgeeks.org/global-local-variables-python/
https://realpython.com/python-scope-legb-rule/

Accumulators:
First we need to set a place for it to begin
If it is a number start with a number of preference
Accumulators can also be empty "" strings
Accumulators can be empty lists [] , etc.

At the end of the loop accumulates the data in the type and order needed.

Fill in the blanks:

```
# Total length of the strings in the list: ["red", "green", "blue"] => 12
total = 0
for word in ["red", "green", "blue"]:
    ____ = ____ + len(word)
print(total)
```

Identifying Variable Name Errors
Read the code below and try to identify what the errors are without running it.
Run the code and read the error message. What type of NameError do you think this is? Is it a string with no quotes, a misspelled variable, or a variable that should have been defined but was not?
Fix the error.
Repeat steps 2 and 3, until you have fixed all the errors.
Looking at the code, it seems like it should identify multiples of 3, 3, 6, and 9.

```
for number in range(10):
    # use a if the number is a multiple of 3, otherwise use b
    if (Number % 3) == 0:
        message = message + a
    else:
        message = message + "b"
print(message)
```

If statements will always run, execute when True,
else if (elif) statements will only run if the above statement is False and the else if stattement is True
else runs and executes when all of the above are False

What does this program print?

```
pressure = 71.9
if pressure > 50.0:
    pressure = 25.0
elif pressure <= 50.0:
    pressure = 0.0
```

```
print(pressure)
```

Fill in the blanks so that this program creates a new list containing zeroes where the original list’s values were negative and ones where the original list’s values were positive.

```
original = [-1.5, 0.2, 0.4, 0.0, -1.3, 0.4]
result = ____
for value in original:
    if ____:
        result.append(0)
    else:
        ____
print(result)
```

```
original = [-1.5, 0.2, 0.4, 0.0, -1.3, 0.4]
result = []
for value in original:
    if value < 0:
        result.append(0)
    else:
        result.append(1)
print(result)
```

```
original = [-1.5, 0.2, 0.4, 0.0, -1.3, 0.4]
result = []
for value in original:
    if value <0:
        result.append(0)
    elif value==0:
        result.append("zero")
    else:
        result.append(1)
print(result)
```

Modify this program so that it only processes files with fewer than 50 records.

```
import glob
import pandas as pd
for filename in glob.glob('data/*.csv'):
    contents = pd.read_csv(filename)
    ____:
        print(filename, len(contents))
```

```
import glob
import pandas as pd
for filename in glob.glob('data/*.csv'):
    contents = pd.read_csv(filename)
    if len(contents)<50:
        print(filename, len(contents))
```

Modify this program so that it finds the largest and smallest values in the list no matter what the range of values originally is.

```
values = [...some test data...]
smallest, largest = None, None
for v in values:
    if ____:
        smallest, largest = v, v
    ____:
        smallest = min(____, v)
        largest = max(____, v)
print(smallest, largest)
```

Multiple ways for representing infinity in Python
https://www.geeksforgeeks.org/python-infinity/

fewest = min(fewest, dataframe.shape[0]) df.shape[0] gives us number of rows, df.shape[1] columns. we are looking at rows in this particular case.

Which of these files is not matched by the expression glob.glob('data/*as*.csv')?

1. data/gapminder_gdp_africa.csv
2. data/gapminder_gdp_americas.csv
3. data/gapminder_gdp_asia.csv

Modify this program so that it prints the number of records in the file that has the fewest records.

```
import glob
import pandas as pd
fewest = ____
for filename in glob.glob('data/*.csv'):
    dataframe = pd.____(filename)
    fewest = min(____, dataframe.shape[0])
print('smallest file has', fewest, 'records')
```

Exercise:

```
def another_function():
    print("Syntax errors are annoying.")
    print("But at least python tells us about them!")
    print("So they are usually not too hard to fix.")
```

What does the following program print?

```
def report(pressure):
    print('pressure is', pressure)
```

print('calling', report, 22.5)

Fill in the blanks to create a function that takes a single filename as an argument, loads the data in the file named by the argument, and returns the minimum value in that data.

```
import pandas as pd
def min_in_data(filename):
    data = pd.read_csv(filename)
    return data.min()
```

REMEMBER that varname.min() returns a minimum value in pandas, assming you read a CSV file into a dataframe.

Variables defined inside functions will be only visible inside that function.

Coding Style
http://swcarpentry.github.io/python-novice-gapminder/18-style/index.html
More on coding style
https://peps.python.org/pep-0008/

More resources for learning Python
Coursera course https://www.coursera.org/specializations/data-science-python
Microsoft Python learning : https://youtube.com/playlist?list=PLlrxD0HtieHhS8VzuMCfQD4uJ9yne1mE6
Google Machine Learning Crash Course https://developers.google.com/machine-learning/crash-course
Data Quest https://www.dataquest.io/
Havard Data Science courses https://online-learning.harvard.edu/subject/data-science

Day 3 Git, GitHub and Open Science

Setup

- Install Git and GitHub Desktop if you haven't - instruction on workshop website https://carpentries-mit.github.io/2022-08-02-mit-online/
- Open your bash shell
    - Mac: open the built-in terminal via Applications/Utilities/Terminal
    - PC: open terminal by running Git Bash program from Windows start menu
    - Linux: open terminal
- Open your GitHub Desktop
- Open these browser tabs

    - and log into your account
    - Etherpad: this window
    - (optional): Git/GitHub lesson https://carpentries-mit.github.io/githublesson/notebooks.html

- (optional) slides for open science: https://bit.ly/SWC-OpenSci

Group Notes

Git/GitHub lesson https://carpentries-mit.github.io/githublesson/notebooks.html

GitHub configuration

$ git config --global user.name "Happy Hiker"
$ git config --global user.email "happy.hiker@moun.tai.ns"
$ git config --global core.editor "nano -w"

Check your settings and get help
$ git config --list
Type Q to get out of long config list

Creating a new repository in GitHub Desktop:
- Check to initialize thie repository with a README - good practice to always do so
- Git Ignore = Python
- License = None

To toggle display hidden files,

- On Mac, Command + Shift + Dot
- On Windows 10 / 8, from the File Explorer window (shortcut: windows key + E) choose –> View –> Show/hide –> check Hidden Items.

Tip: You can always drag a directory from the Finder/Explorer to the terminal to get its full file path

For macs ATOM might be a good text editor option (but there are others), for PCs Sublime Text might be a good one (again, there are many other options).

https://stackoverflow.com/questions/12265754/are-git-emails-of-authors-in-commits-publicly-readable

Merging pull requests: both owner and collaborators can merge pull requests. Owners can also apply rules about merging. If you are the owner, you can check or change settings for collaborators as needed.

Example repos (that Ye showed):

Examples of different projects on GitHub,
https://github.com/dhmit/lang_learn
https://openclimatedata.net/
https://docs.open-reaction-database.org/en/latest/ (https://doi.org/10.5281/zenodo.6028405)
https://github.com/connorcoley/rexgen_direct
https://ualibweb.github.io/UALIB_ScholarlyAPI_Cookbook/content/about/introduction.html