Software Carpentry 14-17 April, 2020

Day 4

Code of Conduct

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

- Use welcoming and inclusive language
- Be respectful of different viewpoints and experiences
- Gracefully accept constructive criticism
- Focus on what is best for the community
- Show courtesy and respect towards other community members

If you believe someone is violating the Code of Conduct, we ask that you report it to The Carpentries Code of Conduct Committee completing **this form**, who will take the appropriate action to address the situation.

License

This Document:

https://tinyurl.com/2020-04-14-swc-day4

Master Document:

https://edu.nl/tcyt9

Workshop website:

https://escience-academy.github.io/2020-04-14-SWC-online/

Zoom:

https://zoom.us/j/652564572?pwd=dk9hVmVkK25tYUo3dkk4R1c1QVhsdz09

Instructor:

Pablo Rodríguez Sánchez and Johan Hidding

Helpers:

Evert Rol, Sarah Alidoost, Jaro Camphuijsen

Roll Call:

name / affiliation / pronouns (optional) / contact, social media etc. (eg. twitter) / expertise

- Johan Hidding / Netherlands eScience Center / j.hidding@esciencecenter.nl tw:@jhidding gh:jhidding / Astrophysics
- Mateusz Kuzak / the Netherlands eScience Center / he, him / t: @matkuzak
- Pablo Rodríguez-Sánchez / eScienceCenter / t: @DonMostrenco / pabrod.github.io / physicist
- Sarah Alidoost / eScience Center / she, her / f.alidoost@esciencecenter.nl / (geo)statistician
- Victor Bernal/ University of Groningen/ v.a.bernal.arzola@rug.nl / victor.arturo.bernal@gmail.com/ Physics, Math, Genomics
- Elisabeth Heijmans/ Leiden University/ e.a.r.heijmans@hum.leidenuniv.nl / history
- Evert Rol (helper) / Netherlands eScience Center / e.rol@esciencecenter.nl
- Terezinha Souza/Maastricht University/t.souza@maastrichtuniversity.nl/ Life Sciences
- Baharak Hooshiar /Helmholtz Centrum Munich //baharak.hooshiar@helmholtz-muenchen.de / Biologist
- Banafsheh Abdollahi/ TU Delft/ b.abdollahi@tudelft.nl
- Paula Martinez Lavanchy / TU Delft/ p.m.martinezlavanchy@tudelft.nl/ Microbiology-Research Data Officer
- Alessia Vitiello/ Wageningen University & Research, Post-doc at Laboratory of Entomology/ alessia.vitiello@wur.nl / Plant Biotechnology
- Julien Dupeyroux / TU Delft / Postdoc in Aerospace Engineering / j.j.g.dupeyroux@tudelft.nl
- Bruna Vieira / Radboudumc / bruna.dossantosvieira@radboudumc.nl
- Hossein Asadi / ULG / asadi.hosein@gmail.com / PhD student in Mechanical engineering
- Michela Busana / Groningen university / michebusana@gmail.com / Biology, ecology

Day Agenda:

09:00 Welcome, coffee, tea, testing setup

09:30 Python

10:45 Morning break

11:15 Python(continued)

12:30 Wrap-up, feedback, post-workshop survey

Contents:

Download the Data

Download the top-most CSV file on this page, called **time_series_covid19_confirmed_global.csv**:
`https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases`

then unzip the file.
Open `jupyter lab` to have the notebook interface and see the file.
In Terminal, lets go to the directory where you placed the file. Then type:
`jupyter lab`

- to make a list

```
pressure =[0.234, 0.235, 0.276]
```

- to print the list

```
pressure
```

- to know about the length of the list

```
len(pressure)
```

- get the first value in the list

```
pressure[1]
```

- reverse the order in the list

```
pressure[::-1]
```

```
name = ”Pablo”
name[3] = r
**give TypeError**
```

- to change the first item in the list

```
pressure[1] = 0.229
pressure
```

- Appending to a list

```
primes = [2, 3, 5, 7]
primes.append(11)
primes
```

```
primes = [2, 3, 5, 7]
teen_primes = [11, 13, 17, 19]
midle_aged_primes = [37, 41, 43, 47]
print(‘primes =’, primes)
primes.extend(teen_primes)
print(‘primes =’, primes)
```

```
primes.append(midle_aged_primes)
print('primes =', primes)

[2, "johan", 4.1, 5.6+3j]
```

- shrinking lists

```
del primes[-1]
primes
del primes[4]
primes
del primes[:]
primes

primes[3]
**IndexError**
```

- make an empty list

```
[]
```

list exercise：we want to fill in the blanks

```
values = ---
values.---(1)
values.---(3)
values.---(5)
print('first time:', values)
values = values[---]
print('second time:', values)
output should be:
first time: [1, 3, 5]
second time: [3, 5]
```

Answers:

```
values = []
values.append(1)
values.append(3)
values.append(5)
print(values)
values = values[1:]
print(values)

values = []
values.append(1)
values.append(3)
values.append(5)
print('first t

ime:', values)
values = values[1:]
print('second time:', values)
```

```
values = []
values.append(1)
values.append(3)
values.append(5)
print(values)
values = values[1:]
print(values)
```

```
values = []
values.append(1)
values.append(3)
values.append(5)
print('first time:' , values)
values = values[1:]
print('second time:' , values)
```

```
values = []
values.append(1)
values.append(3)
values.append(5)
values = values[1:3]
```

```
values = []
values.append(1)
values.append(3)
values.append(5)
print("first time", values)
values = values [1:]
print("second time", values)
```

Another question:

`value[low:high], assuming len(value) >= high, gives a list of what length?`

Answers:

`value[0, high+1]`

Note:

Rule of indexing: len(value[low:high]) == high-low

```
values = [1, 1, 1, 1, 1, 1, 1, 1, 1]
len(values[4:8])
```

output is 4

- to sort a list

```
letters = list('gold')
print(letters)
print("sorted":, sorted(letters))
print(letters)
letters.sort()
print(letters)
```

- For loops

```
total =0
for word in ['red', 'green', 'blue']:
    total = total +1
total
3

total = 0
for word in ["red", "green", "blue"]:
    total = total + 1
print (total)
```

```
total = 0
for word in ["red", "green", "blue"]:
        total = total + len(word)
print(total)
```

```
total = 0
for word in ["red", "green", "blue"]:
   wordc = len(word)
   total = total + wordc
print(total)
```

```
total = total + len(word)
```

- Functions:

```
number = 34
5 * number
```

```
sorted([3, 5, 2])

def print_greeting():
    print(“Hello”)

print_greeting()

def print_date(year, month, day):
    joined = str(year) + ‘/’ + str(month) + ‘/’ + str(day)
    print(joined)

print_date(1871, 3, 19)
output is 1871/3/19

print_date(day=17, month=4, year=2020)
output is 2020/4/17

def average(values):
    if len(values) ==0:
       return None
    return sumn(values) / len(values)

average(range(1, 11))
output is 5.5

m = avergae(range(4, 37498787))
m
output is 18749395.0

d= print_date(2020, 4, 18)
d
d is None
print(d)
```

(Coffee time)

- Function with an explanatory help (a so-called "doc-string")”

```
def print_date(year, month, day):
    “”” Prints the date in a reader’s friendly way

    Details here. Such as… it takes the year, month and day as input.
    “””
    joined = str(year) + "/" + str(month) + "/" + str(day)
    print(joined)
```

- Data acquisition with Pandas

- Loading external data

Tidy Data https://vita.had.co.nz/papers/tidy-data.pdf

- To import data, we want to load pandas

```
import pandas as pd
```

- lets type `pd.` and then press tab, it shows pandas methods
- we choose read_csv()
- then enter the path where the downloaded data is located

```
data = pd.read_csv("data/time_series_covid19_confirmed_global.csv")
data
```

it shows the data

- to get values of latitudes

```
data[['Country/Region', 'Lat']]

data['Lat'] > 24

non_tropical_northern = data['Lat'] > 24
data[non_tropical_northern]

data[data['Lat'] > 24]

data[data['Country/Region'] == 'Netherlands']
```

- Clean data
  - The latitude and longitude are particularly difficult to summarize. …

```
clean_data = data.drop(['Lat', 'Long'], axis=1)
clean_data

clean_data = clean_data.groubby('Country/Region').sum()
clean_data
clean_data.loc['Netherlands']
```

- to select a part of the data based on index

```
clean_data.iloc[0]
```

- **start a new notebook**
- Plotting
  - some basics of plotting

- lets import matplotlib.pyplot

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 16, 25]

plt.plot(x, y)
```

it shows a plot

- to add labels for axes

```
plt.xlabel(‘x variable”)
plt.ylabel(‘y (dependent) variable”)
```

- Good practices

```
fig, ax= plt.subplots(1, 1)
ax.plot(x, y)
ax.set_xlabel(“Independent variable”)
```

- **let's go back to the previous notebook**
- Plotting from data

```
import matplotlib.pyplot as plt
plt.plot(clean_data.loc[‘Netherlands’])
plt.title(‘Cases of COVID19 in NL’)
plt.xticks([0, 15, 30, 45, 60, 75])

plt.plot(clean_data.loc[‘Netherlands’])
plt.plot(clean_data.loc[‘Spain’])
plt.title(‘Cases of COVID19 in NL and ES’)
plt.xticks([0, 15, 30, 45, 60, 75])

plt.plot(clean_data.loc[‘Netherlands’], label=‘Netherlands’)
plt.plot(clean_data.loc[‘Spain’], label=‘Spain’)
plt.title(‘Cases of COVID19 in NL and ES’)
plt.xticks([0, 15, 30, 45, 60, 75])
plt.legend()
```

- Converting columns to datetime:

```
clean_data.columns = pd.to_datetime(clean_data.columns)
```

- then, we don't need to specify ticks

```
plt.plot(clean_data.loc[‘Netherlands’], label=‘Netherlands’)
plt.plot(clean_data.loc[‘Spain’], label=‘Spain’)
plt.title(‘Cases of COVID19 in NL and ES’)
plt.legend()
```

- Plotting data the easy(er) way

```
plt.plot(clean_data)

plt.plot(clean_data.loc[‘Netherlands’:’Tanzania’].T)

plt.plot(clean_data.loc[‘Spain’:’Uganda’].T);
```

- to do a fast plotting

```
clean_data.loc["Netherlands":"Norway"].T.plot(logy=True)
```

https://matplotlib.org/gallery/index.html

```
Answer for Julien's question:
-------

subset = data[data['Country/Region'] == 'France']
france = subset['Province/State'].isnull()
subset[france]
```

Feedback

what went well

The material is very well design
time management was good
Great variety of subjects covered
All the basics are detailed so that anyone can start projects
Well-suited for beginners or people who never programmed before
I liked the organization scheme

what needs improvement

sometimes the pace is slow
The arguments were not all covered, so for the next workshop I suggest to make it for a full week!
I would recommend organizing groups according to their skills in computer programming. Most of the time the pace was too slow for me and I expected the workshop to go way deeper into details.