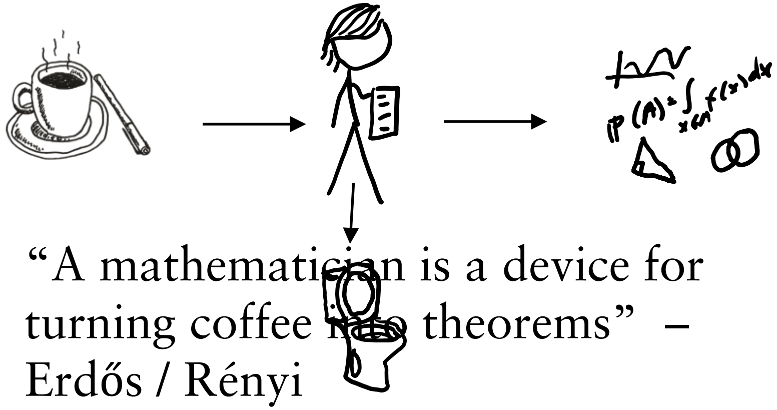
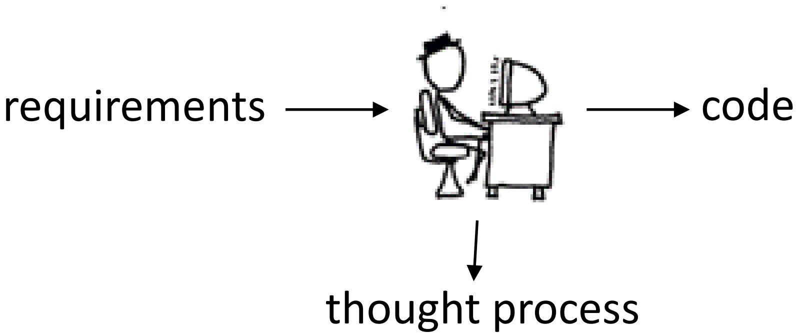
IA Scientific Computing

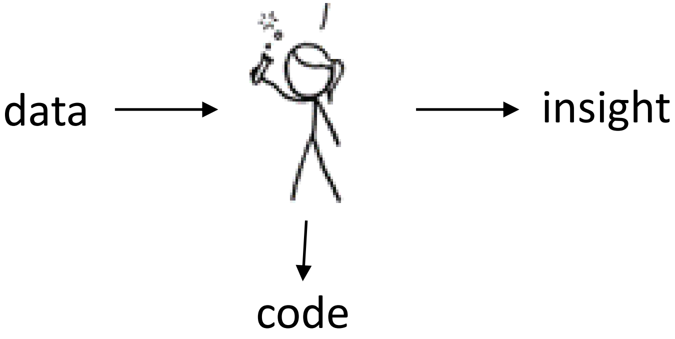
BRIEFING LECTURE



"A mathematician is a device for turning coffee into theorems" – Erdős / Rényi


requirements → code

thought process


data → insight

code

Scientific computing

✦ computing as a tool for doing science

Computer science

✦ the study of computation

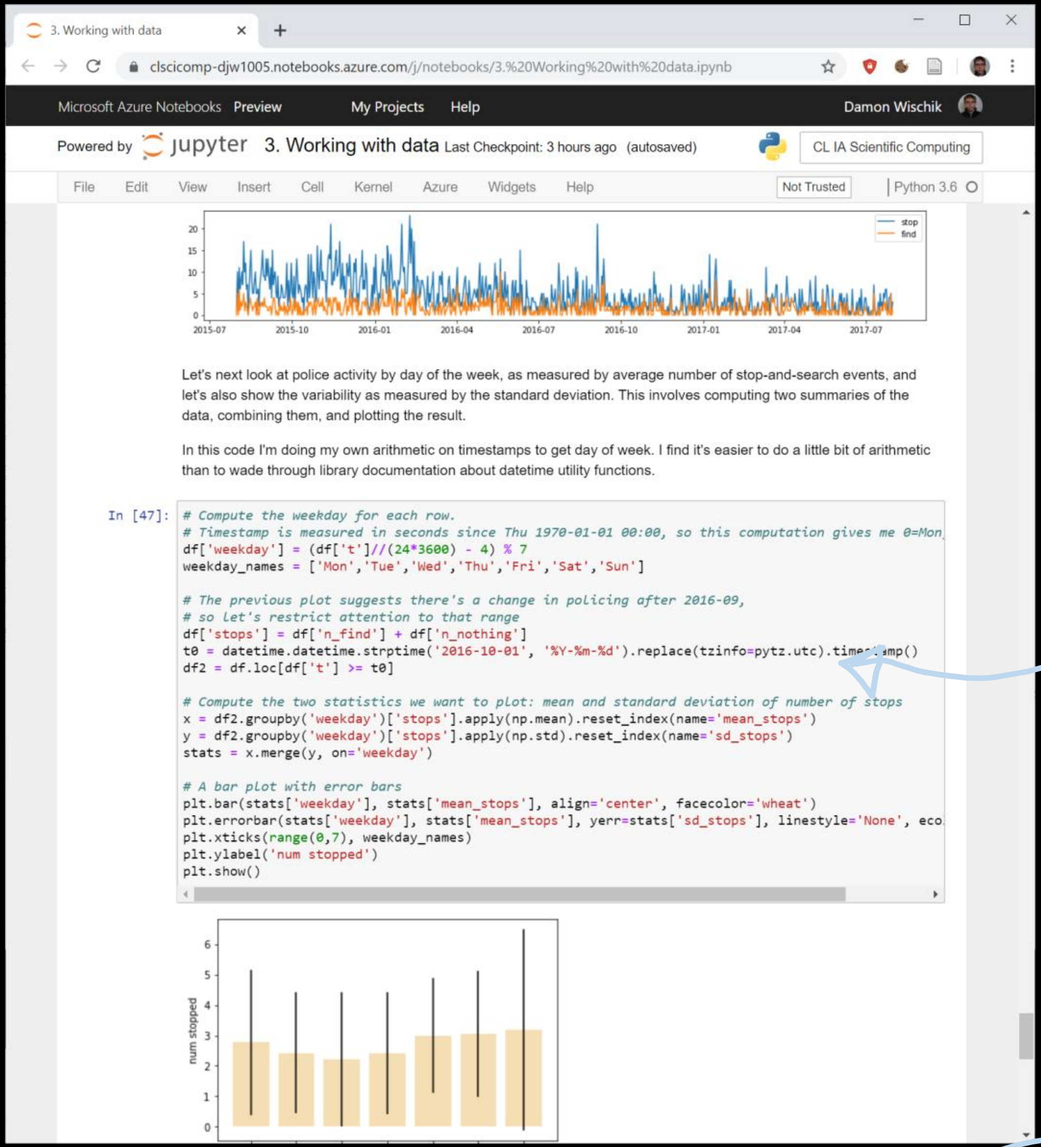
SCIENTIFIC COMPUTING

Try out an idea ✦ see what happens ✦ refine your idea ✦ try something else ✦ iterate … ✦ share what you’ve learnt

CODE AT THE SPEED OF THOUGHT

- Concise one- or two-liners for one-off tasks
- Rich, expressive libraries & glue code

Scientific computing =
Jupyter notebooks + Python + numpy + plotting



Let's next look at police activity by day of the week, as measured by average number of stop-and-search events, and let's also show the variability as measured by the standard deviation. This involves computing two summaries of the data, combining them, and plotting the result.

In this code I'm doing my own arithmetic on timestamps to get day of week. I find it's easier to do a little bit of arithmetic than to wade through library documentation about datetime utility functions.

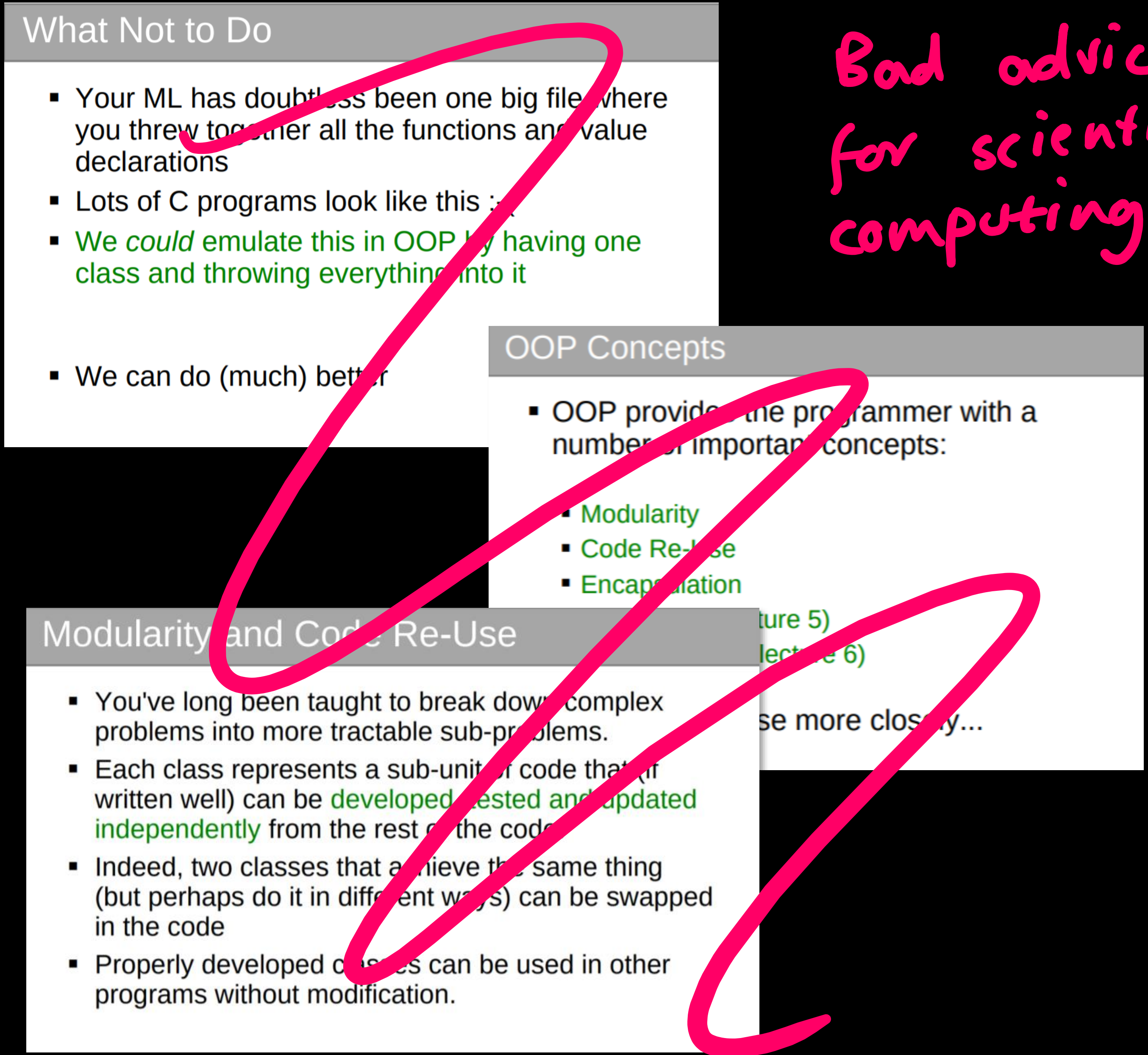
In [47]:

```
# Compute the weekday for each row.
# Timestamp is measured in seconds since Thu 1970-01-01 00:00, so this computation gives me 0=Mon
df['weekday'] = (df['t']//(24*3600) - 4) % 7
weekday_names = ['Mon','Tue','Wed','Thu','Fri','Sat','Sun']

# The previous plot suggests there's a change in policing after 2016-09,
# so let's restrict attention to that range
df['stops'] = df['n_find'] + df['n_nothing']
t0 = datetime.datetime.strptime('2016-10-01', '%Y-%m-%d').replace(tzinfo=pytz.utc).timestamp()
df2 = df.loc[df['t'] >= t0]

# Compute the two statistics we want to plot: mean and standard deviation of number of stops
x = df2.groupby('weekday')['stops'].apply(np.mean).reset_index(name='mean_stops')
y = df2.groupby('weekday')['stops'].apply(np.std).reset_index(name='sd_stops')
stats = x.merge(y, on='weekday')

# A bar plot with error bars
plt.bar(stats['weekday'], stats['mean_stops'], align='center', facecolor='wheat')
plt.errorbar(stats['weekday'], stats['mean_stops'], yerr=stats['sd_stops'], linestyle='None', eco
plt.xticks(range(0,7), weekday_names)
plt.ylabel('num stopped')
plt.show()
```

First I ran this cell up here

And now this cell is producing strange answers

Then this one, I think.



What Not to Do

- Your ML has doubtless been one big file where you threw together all the functions and value declarations
- Lots of C programs look like this :-(
- We *could* emulate this in OOP by having one class and throwing everything into it
- We can do (much) better

OOP Concepts

- OOP provides the programmer with a number of important concepts:
  - Modularity
  - Code Re-Use
  - Encapsulation
  - ture 5)
  - lecture 6)
- se more closely...

Modularity and Code Re-Use

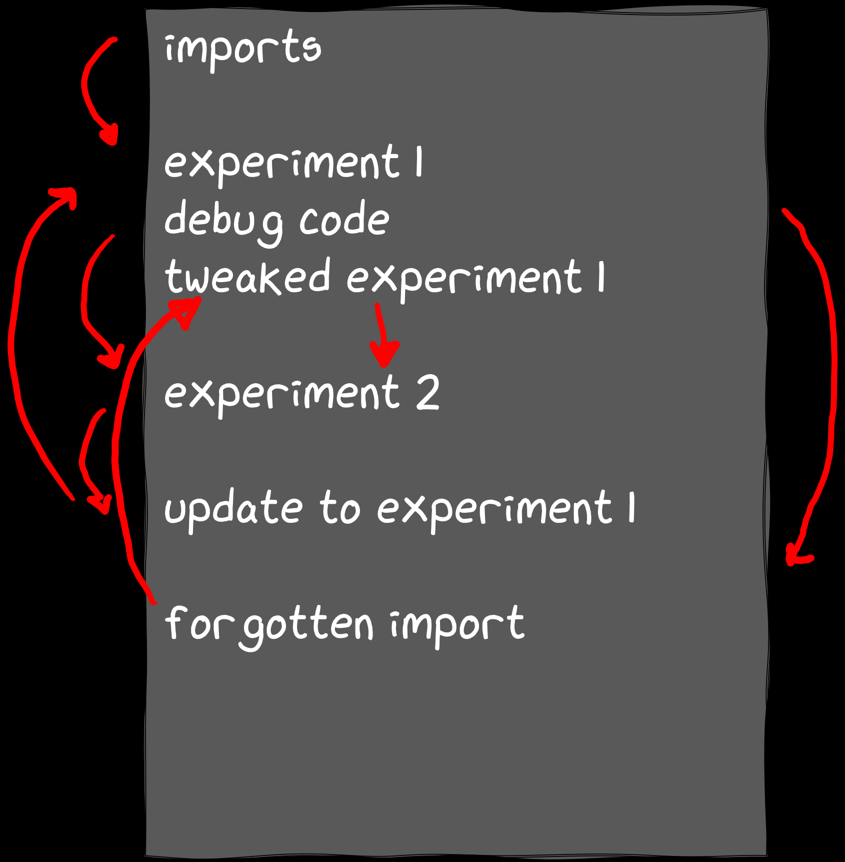
- You've long been taught to break down complex problems into more tractable sub-problems.
- Each class represents a sub-unit of code that (if written well) can be developed, tested and updated independently from the rest of the code.
- Indeed, two classes that achieve the same thing (but perhaps do it in different ways) can be swapped in the code
- Properly developed classes can be used in other programs without modification.

Bad advice
for scientific
computing

Look at each line of your code and ask yourself: ‘does this spark joy?’ If not, delete it.

Marie Kondo

while working



imports

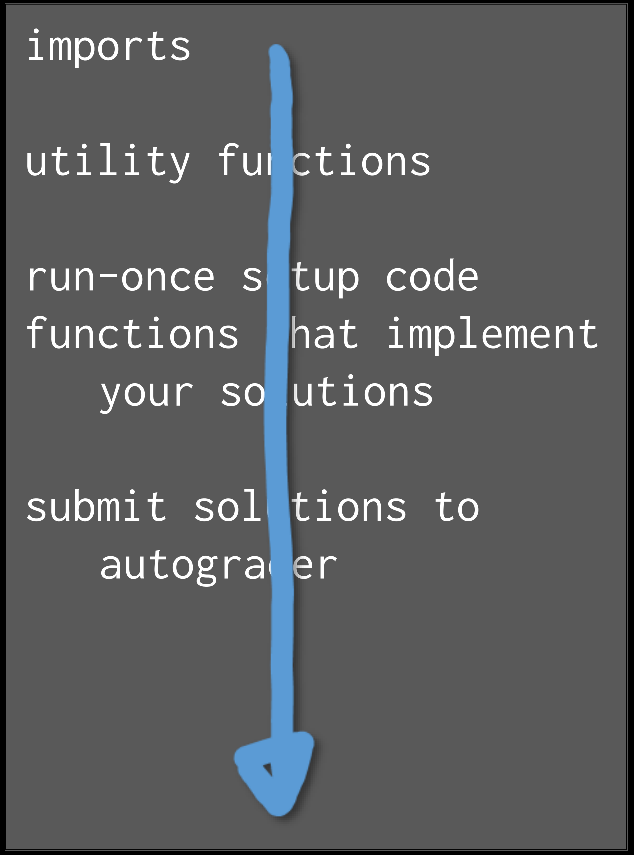
experiment 1
debug code
tweaked experiment 1

experiment 2

update to experiment 1

forgotten import

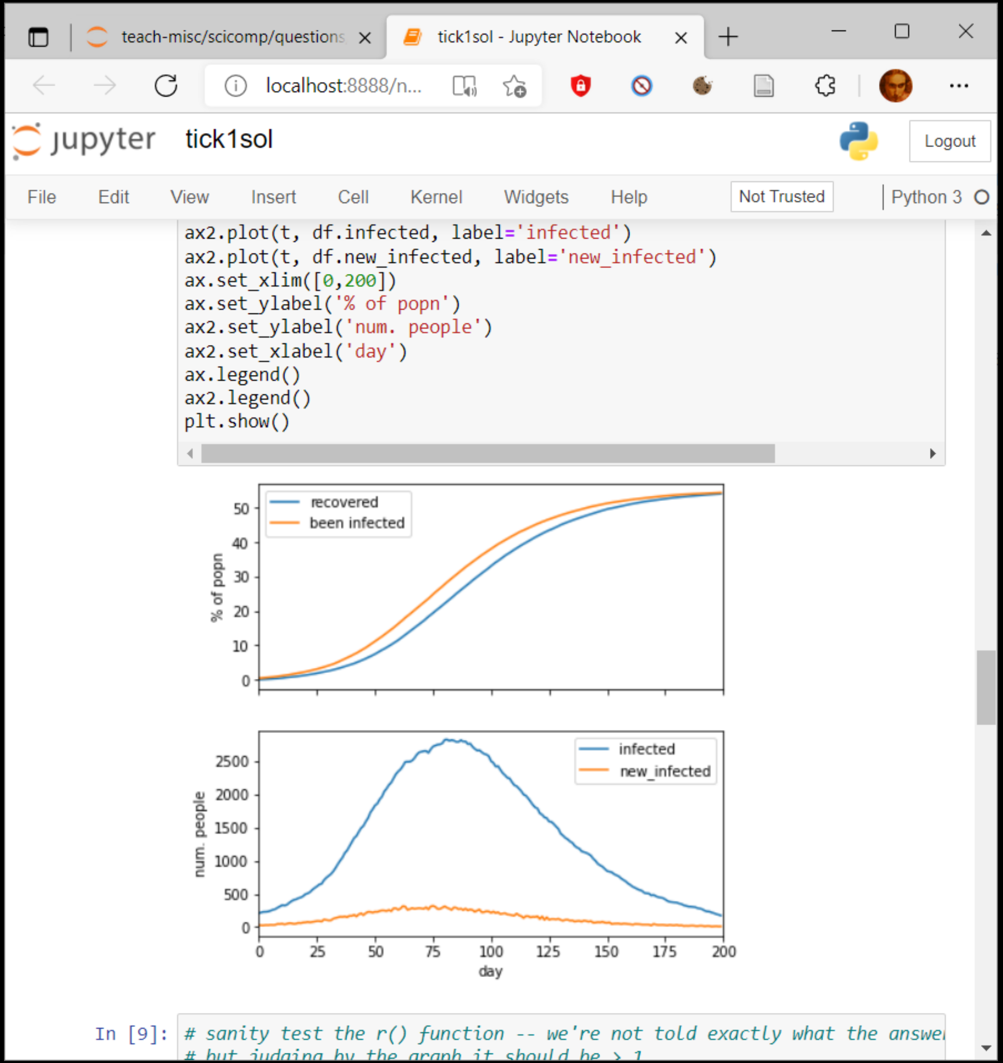
after you've finished



```
imports

utility functions

run-once setup code
functions that implement
   your solutions

submit solutions to
   autograder
```

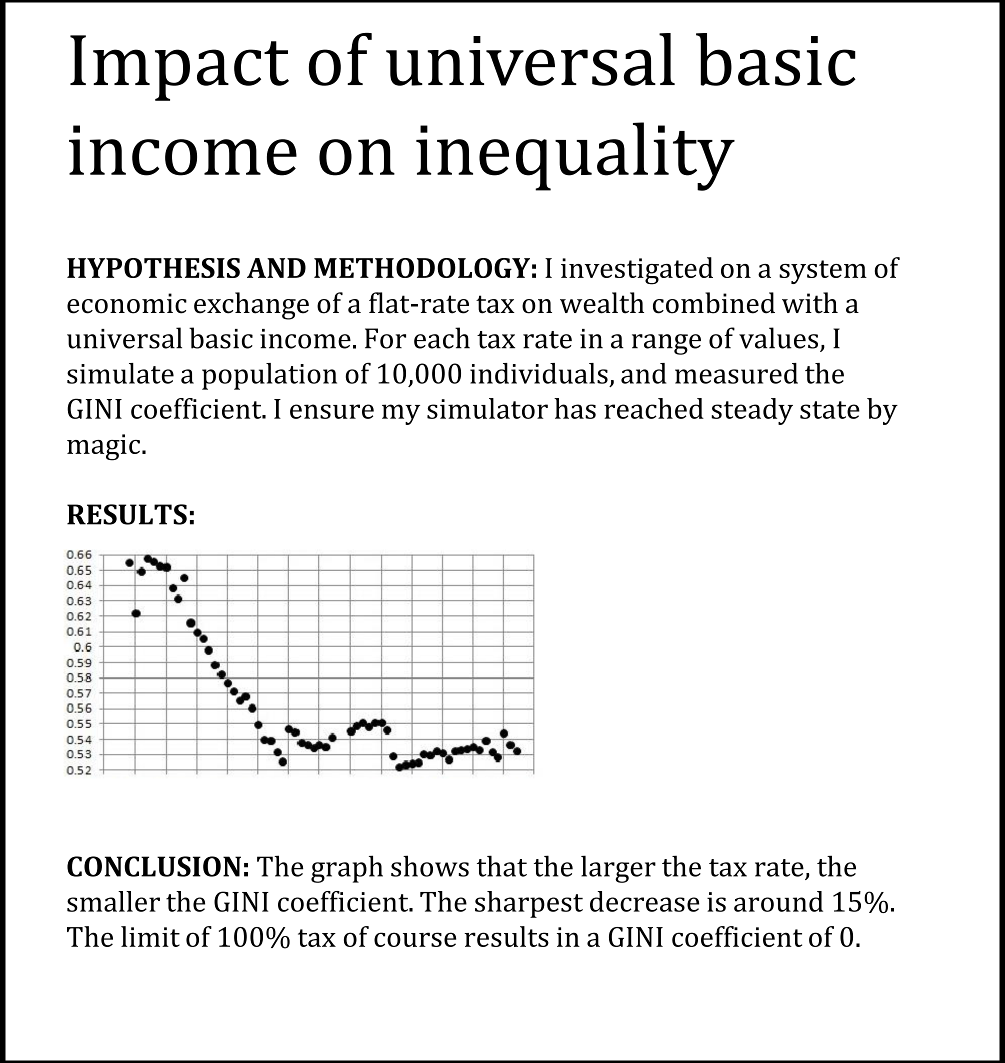
Tick 1,2: Econo-physics simulator (with answers checked by autograder)

Tick 3: plots
Tick 4: One-page scientific report



```
ax2.plot(t, df.infected, label='infected')
ax2.plot(t, df.new_infected, label='new_infected')
ax.set_xlim([0,200])
ax.set_ylabel('% of popn')
ax2.set_ylabel('num. people')
ax2.set_xlabel('day')
ax.legend()
ax2.legend()
plt.show()
```

```
In [9]: # sanity test the r() function -- we're not told exactly what the answe
        # but judging by the graph it should be > 1
```



Impact of universal basic income on inequality

HYPOTHESIS AND METHODOLOGY: I investigated on a system of economic exchange of a flat-rate tax on wealth combined with a universal basic income. For each tax rate in a range of values, I simulate a population of 10,000 individuals, and measured the GINI coefficient. I ensure my simulator has reached steady state by magic.

RESULTS:

CONCLUSION: The graph shows that the larger the tax rate, the smaller the GINI coefficient. The sharpest decrease is around 15%. The limit of 100% tax of course results in a GINI coefficient of 0.

TUTORIALS

0. Programming in Python
   language quirks

1. Numerical computation
   `numpy`
2. Plotting data
   `matplotlib`

3. Working with data
   pandas

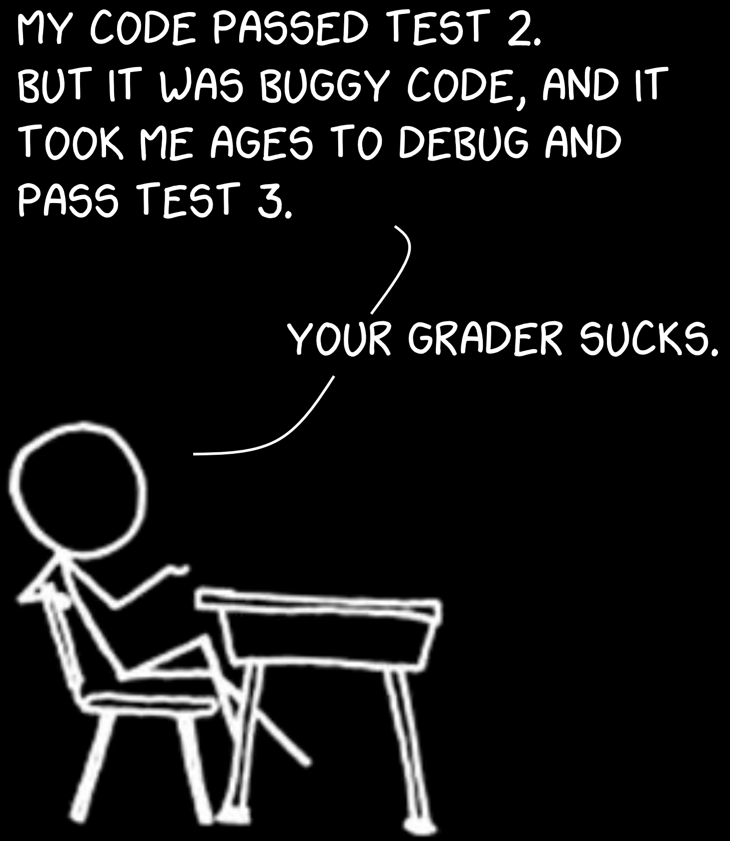
A. Data scraping recipes

ASSESSMENT
(8% of maths paper)

No written exam

Four ticks, each marked pass/fail
Ticks 1 and 2: pass the autograder, submit notebook by 23 Jan
Ticks 3 and 4: submit pdf by 6 Feb

Some of you will have in-person ticking

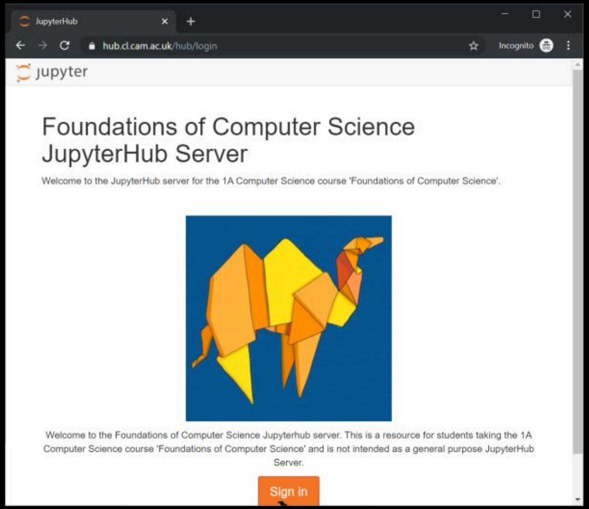

MY CODE PASSED TEST 2.
BUT IT WAS BUGGY CODE, AND IT TOOK ME AGES TO DEBUG AND PASS TEST 3.
YOUR GRADER SUCKS.

Scientific computing isn’t about meeting requirements, it’s about discovery.

✦ chart your own path ✦ write your own tests ✦ invent a few small test cases ✦ work them out with pen and paper ✦ make sure your code agrees

The autograder will run wherever you run Jupyter + Python3

hub.cl.cam.ac.uk

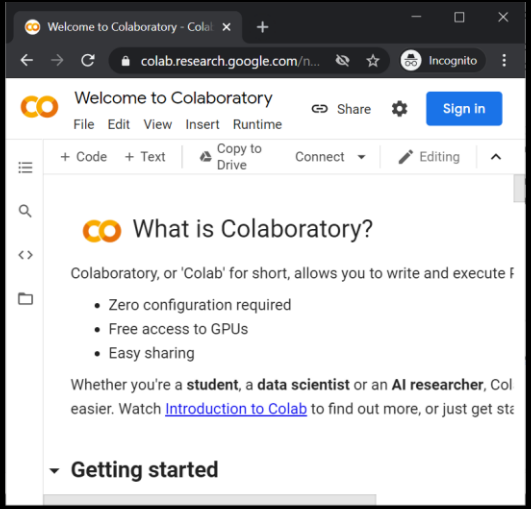


Foundations of Computer Science JupyterHub Server

Welcome to the JupyterHub server for the 1A Computer Science course 'Foundations of Computer Science'.

Welcome to the Foundations of Computer Science Jupyterhub server. This is a resource for students taking the 1A Computer Science course 'Foundations of Computer Science' and is not intended as a general purpose JupyterHub Server.

Google colab



What is Colaboratory?

Colaboratory, or 'Colab' for short, allows you to write and execute F

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Col easier. Watch Introduction to Colab to find out more, or just get sta

Getting started

VSCode

Help and support

- Moodle help forum
- Mini-lectures and help sessions early in Lent term