Introduction to Scripting for Scientists

A Very Brief Guide for the Intrepid Labratorian

Delta Alexander Sharp

May 29, 2023

Outline

Who am I?

I'm Delta Alexander Sharp (they/them.)
Mad scientist extraordinaire.

- I've been programming since 2014 specializing in:
  - Data Science
  - Embedded Devices
  - Reproducible Research
- I've written multiple pieces of internal business software which have increased productivity.

Discussion Expectations

What we'll be covering

- What is programming / scripting.
- How computers represent things.
- How scripting can make your work more reproducible.
- Why introducing scripts can make your work easier as a scientist.

On Jargon

It can be intimidating learning new skills with lots of jargon.

Computer Science is racked full of jargon

Examples of Jargon

- Strings, floats, dictionaries
- Sequences, maps, iterators
- Functions, class, objects
- URL, API, blobs

Figure: A jargon goblin

What Is Programming?

Definition (Program)

A set of instructions interpreted by a computer.

Definition (Script)

A program that typically:

- Specialized to a task
- Has minimal user interaction
- Can be called from the shell or other programs

How do you talk to computers?

Modern computers run an operating system (OS) used to schedule programs.

Definition (Kernel)

The core of the OS, interacts with the machine.

Definition (Shell)

The part of the OS that exposes the kernel to the user.

Definition (Desktop Environment)

A program that represents the shell graphically.

Computers can handle Massive data sets

The human genome is made up of 3.2 billion base pairs (Toledo and Saltsman 2012).

Exercise: How big is that?

How long would it take to transcribe at:
60 words per minute taking 8 hour days (Toledo and Saltsman 2012)?

- A. 1 year
- B. 100 years
- C. 50 years
- D. 30 years

Rapid Sequencing

Stanford scientists sequenced a human genome in just over 5 hours (Armitage 2012)!

Computers can handle Massive data sets

The human genome is made up of 3.2 billion base pairs (Toledo and Saltsman 2012).

Exercise: How big is that?

How long would it take to transcribe at:
60 words per minute taking 8 hour days (Toledo and Saltsman 2012)?

- A. 1 year
- B. 100 years
- C. 50 years
- D. 30 years

Rapid Sequencing

Stanford scientists sequenced a human genome in just over 5 hours (Armitage 2012)!

Alright, I'll Byte

Definition (Bit)

A bit is an object,that can be in one of two states.

- on or off
- 1 or 0
- True or False

Definition (Byte)

A byte is a sequence of 8 bits.

Definition (Blob / Binary)

An arbitrarily large sequence of bytes.

Computers represent things using bytes

Using a large number of bytes you can represent all sorts of things:

Examples

- Text, numbers, tables
- Images, music, videos
- Programs, databases, hardware

Computers don't understand data!

Computers only see and work on these blobs.

- The meaning comes from people.

Scales of Information

Scale is described using SI notation.

Common prefixes

Prefix	number of bits
	1
Kilo (kb)	1e+3
Mega (mb)	1e+6
Giga (gb)	1e+9
Tera (tb)	1e+12

Examples of files sizes

File type	typical size
Images	~2kb - 1000kb
Documents	~4kb - 5mb
Music	~3mb - 4mb
Video	~4gb - 20gb
Genomic Data	~100mb - 100tb

Data analysis and visualization

People are natural story tellers

Real world data rarely tells neat stories.

Analysis finds the story

Statistical calculations sleuth out the who, what, when, where, why.

Visualization tells it

Tables and charts show the story in a way people can understand.

Example: Sanguine

Blood Transfusion is a Complex Problem

Transfusion is a powerful tool that has saved countless lives.

However as with any clinical procedure comes with risk such as transfusion-associated circulatory overload (TACO), which occurs when blood volumes become too high and can lead to pulmonary edema (Carrigan 2023).

Sanguine (cont.)

Figure: (Carrigan 2023)

Data visualization tips

Strive for accessible visualizations!

Do!

- Use patterns and shapes in addition to color
- Use labels and legends
- Translate the data into clear language
- Provide context; explain the visualization

Don't!

- Rely on color to explain data
- Use very bright or low-contrast colors
- Hide important data behind interactions
- Overwhelm the user with information

Reference

`https://fossheim.io/writing/posts/accessible-dataviz-design/`

Computers and people solve problems differently

People are natural problem solvers, whereas computer need direction.

Example (Image Recognition)

People automatically pick details out of images:

- Color
- Shapes
- Animals
- Text

Figure: Spike Chunsoft (2009)

Computers and people solve problems differently

People are natural problem solvers, whereas computer need direction.

Example (Image Recognition)

People automatically pick details out of images:

- Color
- Shapes
- Animals
- Text

Figure: Spike Chunsoft (2009)

Computers are precise and repeatable

Computers designed to be deterministic

- They do exactly what you tell them
- As many times as you need
- Whenever you want them to

Ideal use cases for computers

- Text manipulation
- Time-stamping
- Continuous monitoring
- Counting and sorting

Computers have Consequences

Computers are complex systems

Computers are run on physical machines, opening them up to intermittent errors.

- Memory leaks
- Buffer overflows
- Hanging or freezing

Resetting the machine generally returns it to a known state.

Computers are fallible!

Malformed and malicious code

- Computers can and will destroy themselves.
- Bad code can and will be exploited.
- Networks scale damage up exponentially!

General advice

That being said, every tool can be misused.

- Don't use admin accounts for daily use.
- Don't use personal computers for work.
- Don't use work computers for personal reasons.
- Don't plug non-secure devices into secure ones.
- If it doesn't need a network connection; don't connect it.

Scripting allows you to redo you work easily

Reusing code can let you standardize and automate your workflow.

Example (Pasteurization Report Generator)

A previous employer pasteurized breast-milk for at-risk newborns.
PRG:

- took daily raw data and separated into batches
- validated that batch met critical control points
- produced reports for each batch for analysis.

Version Control (VC)

Collaborating with colleagues is a non trivial problem.

People Problems

- Overwriting each other's work
- Stale and out-of-date copies
- Breaking changes
- Crediting contributions

VC to the rescue!

- Track changes
- Ensures changes are current
- Revert broken commits
- Access control, and logging

Version control programs

- Git
- Subversion
- Rsync
- Mastercontrol

In short:

Computers can make your work more robust and reproducible.

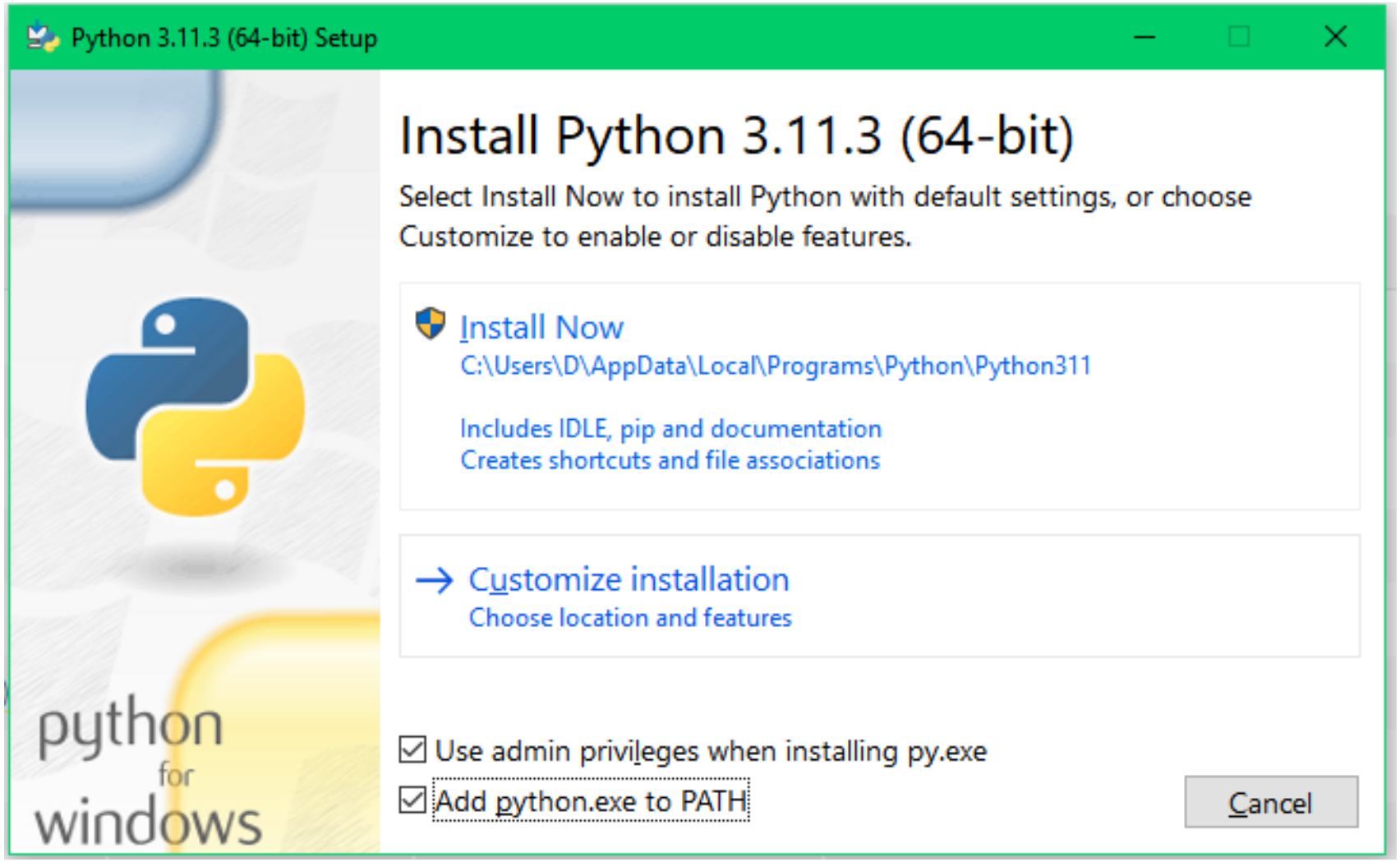
Science is based on reproducibility!

How do you find out how the world works?

- Observe:
  What are you interested in learning?
- Hypothesize:
  What do you think is happening?
- Test:
  Cast your ideas to the crucible!
- Document and Share:
  Show your work!

Installing Python

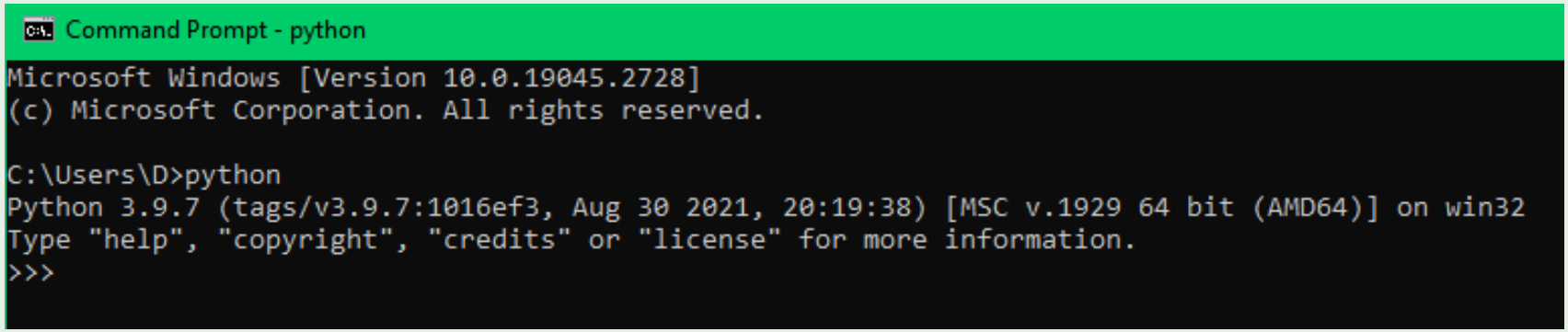
You can download the latest version of python at:
`https://www.python.org/downloads/`



Launching Python

After installing, python can be launched from the command line:

Example (Windows)



Using the REPL (Read, evaluate, print, loop)

1. Enter a valid python command
2. Press enter / return
   - The result is printed to next line of the console
3. To quit type `exit()` + `return` or `CTRL-D`

Integrated Development Environments (IDEs)

While the REPL is useful, it often isn't practical to program excusively using it. Integrated development evironments offer many quality of life improvements.

Pros of IDEs

- Autocomplete
- Integrated shell
- Build tools
- Version control

IDE examples

- Emac
- Pycharm
- Spyder
- VSCode

Types and how things are represented

Built into python are a few representations of things. These things are called objects, which can have different types

Common Types

- Numeric:
  Integers, Floating Points, Complex Numbers
- Sequence and mappings:
  Lists, Tuples, Strings, Dictionaries, Ranges
- Functions:
  Code you want to reuse
- Classes and objects:
  Bundles of code and data

Numeric

Python has three built in ways of representing numbers:

Types of Built in Numbers

1. Integers (int)
   - Whole numbers.
   - Infinite precision.
2. Floating point (float)
   - Decimals (but in base 2)
   - precision limited by hardware.
3. Complex number (complex)
   - exists on the complex plane.
   - has a real and an imaginary part.
   - can have both float and int parts.

Precision, Accuracy, and Uncertainty

It can be tempting to think that computers are better at math than people.

Consider the following:

Computers have a limited memory to represent numbers with.

Code

```
a = 1.2
b = 1.0

print(a - b)
```

What result do you expect?

- A. 0.20000000000000000
- B. 0.19999999999999996
- C. 1.0000000000000001
- D. 0.1999999999999999

Garbage in, garbage out!

Remember to account for the uncertainty or you'll end up with mysterious errors and useless data.

Precision, Accuracy, and Uncertainty

It can be tempting to think that computers are better at math than people.

Consider the following:

Computers have a limited memory to represent numbers with.

Code

```
a = 1.2
b = 1.0

print(a - b)
```

What result do you expect?

- A. 0.20000000000000000
- B. 0.19999999999999996
- C. 1.0000000000000001
- D. 0.1999999999999999

Garbage in, garbage out!

Remember to account for the uncertainty or you'll end up with mysterious errors and useless data.

Sequences and Collections

Sometimes, you need to store a bunch of things together:

Example (List)

A mutable (update-able) collection of objects.

```
example_list = ['a','b','c','d']
```

Example (Tuple)

A frozen (non-update-able) collection of objects.

```
example_tuple = (1,2,3,4)
```

Example (Dictionary)

A collection of key, value pairs (mappings.)

```
example_dict = {"key1": 123, "key2": "abc"}
```

Strings

Computers store textual data as strings

Strings are a collection of symbols including:

- letters
- numbers
- punctuation
- control sequences

Example

H	E	L	L	O		W	O	R	L	D	!
0	1	2	3	4	5	6	7	8	9	10	11

Paths and URLS

How do you represent where a file or resource is?

Uniform Resource Locator (URL)

A string showing:

- **Protocol** What language to use
- **Domain** Which machine to talk to
- **Path** Where a resource is

Definition (Path)

A string telling the machine where a resource is located.

Types of paths

There are two types of paths:

Absolute Path

- Exact
- Inflexible

`/home/delta/Desktop/example.org`

Relative Path

- Relative
- Flexible

(Assuming CWD: `/home/delta/`)

`./Desktop/example.org`

Classes and objects

To define a custom object, you use a class. Classes allow you to bundle code and data as a single object.

Example (Class)

```
class ExampleClass:
    def __init__(self, args):
        self.args = args
        self.bar = "Hello World!!"

    def foo(self):
        print(self.bar)

example = ExampleClass("Hi")
```

You can keep multiple independant copies of an object, allowing for problems to be encapsulated and isolated.

Conditional Logic

The real power of programming comes from the ability to define logical conditions.

Example

```
name = input("What's your name? ")

if len(name) > 10:
    print("Wow that's a long name!")
elif len(name) <= 10 and len(name):
    print(f"Hi {name}!")
else:
    print("Sorry didn't catch that!")
```

Reusing Code: Modules and Libraries

Modules allow you to share python programs and libraries and use them in your projects.

Using modules

```
#import a module
import random
#import a specific submodule
from random import randint
#import all the submodules
from random import *
```

By using modules you don't need to reinvent the wheel every time you want to make something.

Installing new Modules

To install a new module you use the command-line program `pip`:

Example (Using pip)

```
pip install numpy
pip uninstall numpy
```

Common 3rd party libraries for science

- Numpy:
  Handle large arrays efficiently
- Scipy:
  Useful mathematical and scientific functions.
- Sympy :
  Symbolic mathematics.
- Pandas :
  Handle datasets with ease.
- Matplotlib:
  Create graphs, charts, and figures.
- Pint:
  Ensures your math takes units into account.

Why not use office software?

While spreadsheet programs are useful tools, they aren't built for scientific work.

Autocorrect mangles data

30% of published papers contain mangled gene names in supplimentary data (Lewis 2021).

Can't handle large datasets

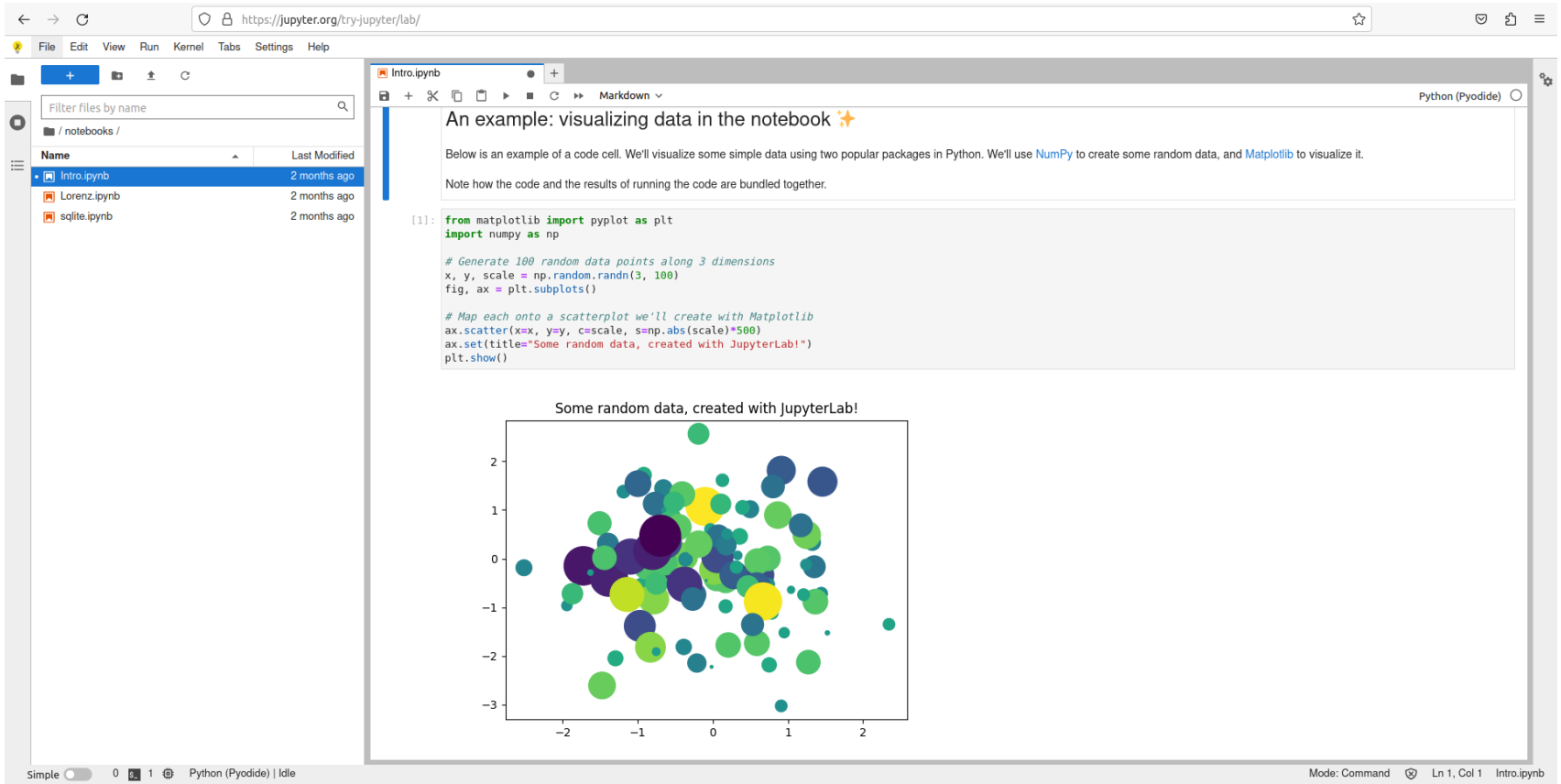
Excel has a limit of 1,048,576 rows and 16,384 columns per sheet (Microsoft 2023).

Bulky

When dealing with larger datasets, typical spreadsheet software can become unresponsive.

Jupyter

Jupyter notebooks allow for you to define create documents that weave code and data together.



Jupyter (cont.)

Jupyter notebooks have many benefits including:

- Live and reactive plots
- Directly interact with APIs
- Sharable format

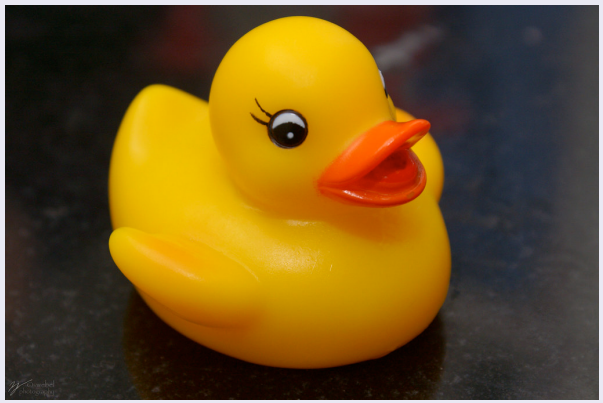
Learning to program is hard, but you can do it.

Learning is a physical process that takes time and energy.

When you encounter trouble

At somepoint, you will hit a metaphorical brick wall.

- Take a break and re-evaluate the problem
  - What are you trying to do?
  - Are you asking the right question?
- Double check your steps
  - Trace-back messages are cryptic at first.
  - Learn to read them for invaluable feedback.
- Try duck debugging!
  - Explain what you're working on to a rubber duck.
  - While silly it helps more often than not.



Ask for help when you need it!

Community Forums

Communities like stackoverflow.com exist for asking and answering questions.

Before asking

Badly formed questions are only going frustrate people.

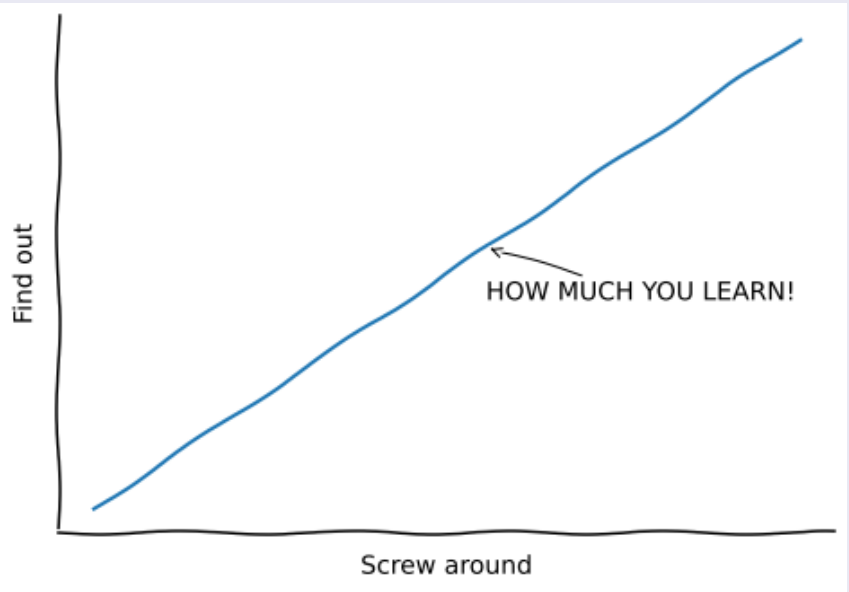
- What are you trying to do?
- What is happening?
- What steps have you tried?

Embrace mistakes when learning

Fail fast; fail often

You will make mistakes, that's part of the learning process.

- Don't be afraid of being wrong!
- Iterate on your previous work!
- Keep it simple!
- Document your progress!
- How do others approach this?
- Build tests to verify your results!



Still not convinced?

This presentation was prepared using the techniques discussed.

Tools used

- emacs
  - orgmode
  - org-ref
- latex
  - bibtex
  - beamer
  - tikz
- python
  - matplotlib

Source

You can find the source document here:

`https://github.com/Nalisarc/intro-to-scripting`

Summary

Computers are powerful tools that can greatly improve your work as a scientist.

Prompt for questions

Thank you for your attention!
Additional questions can be sent to: delta.sharp@astragroup.info

Further Reading

- Al Sweigart: "Automate the Boring Stuff with Python"
  isbn: 978-1-593-27992-9
- Lee Vaughan: "Python Tools for Scientists"
  isbn: 978-1-718-50266-6
- Vince Buffalo: "Bioinformatics Data Skills"
  isbn: 978-1-449-36737-4
- Claus O. Wilke: "Fundamentals of Data Visualization"
  isbn: 978-1-492-03108-6

Bibliography

Armitage, Hanae. 2012. "Fastest DNA Sequencing Technique Helps Undiagnosed Patients Find Answers in Mere Hours." https://med.stanford.edu/news/all-news/2022/01/dna-sequencing-technique.html.

Carrigan, Kellie. 2023. "Sanguine, a New Data Visualization Tool, Predicts Transfusion Needs and Provides Risk-Adjusted Benchmarking to Improve Patient Outcomes." https://www.aruplab.com/magnify23/sanguine-new-data-visualization-tool-predicts-transfusion-needs.

Lewis, Dyani. 2021. "Autocorrect Errors in Excel Still Creating Genomics Headache." https://www.nature.com/articles/d41586-021-02211-4.

Microsoft. 2023. "Excel Specifications and Limits." https://support.microsoft.com/en-au/office/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3.

Toledo, Chelsea, and Kirstie Saltsman. 2012. "Genetics by the Numbers." https://nigms.nih.gov/education/Inside-Life-Science/Pages/Genetics-by-the-Numbers.aspx.