Penn Libraries
University of Pennsylvania

PYTHON FOR HUMANISTS

A workshop in the "Digital Humanist's Help Desk" series

November 7, 2025

Cynthia Heider, Public Digital Scholarship Librarian

Follow along with links at: **https://bit.ly/43EUIRp**

OVERVIEW

This short workshop will introduce humanists to the programming language Python and the numerous possibilities for its use for research in the humanities. No prior programming experience required!

Learning goals:

- *Set up a Python coding environment suitable for use with your own computer*
- *Recognize potential use cases for Python and its libraries as humanistic research tools*
- *Identify available resources for practicing skills and creating projects with Python*

WHO AM I?

Cynthia Heider

Public Digital Scholarship Librarian

Email: heider@upenn.edu
Book a consultation: www.library.upenn.edu/staff/cynthia-heider

What can I help with?

- Digital projects for public audiences
- Digital projects involving community partners
- Choosing the right tool or platform for your digital project
- Working with data in the humanities
- Connecting with digital project resources at and beyond UPenn

Research Data and Digital Scholarship (RDDS)

- Support across research data & digital scholarship lifecycles
- Encourage thoughtful and informed use of emerging technologies in our fields of expertise
- Aid in the creation & dissemination of traditional + cutting-edge scholarly outputs
- Provide custom workshops, in-class visits, and consultations (one-on-one or team-based).

Website: www.library.upenn.edu/RDDS
Email: LibraryRDDS@pobox.upenn.edu
Workshop calendar: bit.ly/RDDS-events

INTRODUCTION

What is Python? Why use it? What potential does it hold for humanities research?

WHAT IS PYTHON?

Python is a free, popular, open-source programming language that has lots of uses for the humanities.

Why use it?

- Open source
- Has stood the test of time
- Lots of support
- Available for everyone
- "General purpose"
- Human-friendly
- Easy to learn

THINGS HUMANISTS CAN DO WITH PYTHON

Gather information	Parse documents and pulling out relevant information
Exploring patterns	Source analysis
Visualizing data in charts, graphs, infographics, maps, and more	Building interfaces to work with source materials and data
Math - it does any calculations for you :)	Your idea here!

HUMANITIES RESEARCH DATA

What does data look like in the humanities?

correspondence
estate inventory
shipping manifest
diary or journal
financial ledger
photograph
drawing or other work of visual art
burial, birth, or wedding record
newspaper
property record or deed
tax list
notebook
memorandum
telegram
speech
material culture item
genealogy
recipe or menu
film, novel, or other cultural artifact
directory
logbook
blueprints
built structure
price list
advertisement
legislation
government proceedings
transcript
interview
autobiography
schematic or plan
diagram or chart
land survey or map
polling results
census
audio or video recording
publication

… and more!

WORKING WITH HUMANITIES DATA IN PYTHON

Applications we'll be specifically discussing today include:

- Parsing the Chronicling America newspaper database with an API
- Extracting location data from Library of Congress collection items
- Looking for color clusters in LOC collections
- Counting the most frequent words in William Shakespeare's Romeo and Juliet

A NOTE ON AI

Generative AI has introduced new paths to programming and learning how to code. Many programmers use tools like Microsoft Copilot, GitHub Copilot, and Claude Sonnet to write, streamline, and troubleshoot their code. They can be very helpful tools, especially when you are a little more experienced using a programming language.

But be aware:

- GenAI can produce really messy, bloated code, and you may not be able to learn much about **why** it is choosing a particular approach. This isn't very helpful when you're trying to learn.
- Using genAI is highly resource-intensive and its data centers are contributing to climate change. You may want to use it sparingly.
- Guidelines about how genAI gathers and uses the data that is input into it is often not clearly discernible. Do not enter sensitive information, and use it in accordance with Penn's guidance on its use.

THE BASICS

What does a script look like? What tools can you use to work with them? What are some general programming basics, like data types and logic?

A PYTHON SCRIPT

```
tell_time.py
"""
A script to show the user the current time.

Author: Matthew Pitkin
Email: m.pitkin@lancaster.ac.uk
Date: 22/06/2020
"""
```

a comment

```
# import the required modules
import datetime
```

import a module from a library

```
def gettime():
    """
    A function to return the current time.

    Returns
    -------
    tuple:
        A tuple containing the hour, minutes and seconds.
    """

    now = datetime.datetime.now()

    return now.hour, now.minute, now.second + 1e-6 * now.microsecond
```

define a function (mini, reusable piece of code)

```
# get the time
hour, minute, seconds = gettime()

print(f"The current time is {hour}:{minute}:{seconds}")
```

perform a task and format the result as output

GETTING STARTED

Python is an *interpreted* language.

Downloadable interpreters and Integrated Development Environments (IDEs)

Web-based IDEs

Interactive notebooks

LOAD THE INTERACTIVE NOTEBOOK

Navigate to our Google Colab environment to access the Notebooks at:
https://colab.research.google.com/github/upenndigitalscholarship/p4h/blob/main/content/notebooks/p4h.ipynb

Note: You will need to log into a Google account to use this tool

VARIABLES

Use variables to store values.
A variable is a kind of "sticky note."

- Variables are names for values.
- In Python the = symbol assigns the value on the right to the name on the left.
- The variable is created when a value is assigned to it.

VARIABLES

Use variables to store values.
A variable is a kind of "sticky note."

Variable names:

- cannot start with a digit
- cannot contain spaces, quotation marks, or other punctuation
- may contain an underscore (typically used to separate words in long variable names)

VARIABLES

Use variables to store values.
A variable is a kind of “sticky note."

You can also use them for calculations!

Example:
>>> age = age + 3
>>> print('Age in three years:', age)

EXPLORING DATA TYPES

Integers (whole numbers)
Floats (decimals)
Strings (text)
Booleans (true/false)

Practical Implications

- Every value has a type.
- Types control what operations can be done on values.
- Strings can be added and multiplied.
- Strings have a length (but numbers don't).
- Must convert numbers to strings or vice versa when operating on them.
- Can mix integers and floats freely in operations.
- Variables only change value when something is assigned to them.

SHOWING YOUR WORK WITH THE 'PRINT' FUNCTION

Use print to display values

- Python has a built-in function called print that prints things as text.
- Call the function (i.e., tell Python to run it) by using its name.
- Provide values to the function (i.e., the things to print) in parentheses.
- To add a string to the printout, wrap the string in single quotations.
- The values passed to the function are called ‘arguments’

Example

```
>>>print(first_name, ‘is’, age, ‘years old’)
```

MORE CONCEPTS

What does a script look like?

LISTS

Lists are a common data structure to hold an ordered sequence of elements.

Each element can be accessed by an index. Note that Python indexes start with 0 instead of 1:

EXAMPLE:

```
>>> numbers = [1, 2, 3]
>>> numbers[0]
```

LISTS

Lists are a common data structure to hold an ordered sequence of elements.

Other things you can do with lists:

Add items to the end of a list with append:

```
>>> numbers.append(4)
>>> print(numbers)
[1, 2, 3, 4]
```

LISTS

Lists are a common data structure to hold an ordered sequence of elements.

Other things you can do with lists:

Sort items in a list with sort:

```
>>>Ages = [28, 19, 60, 80, 23]
>>>ages.sort()
>>>print(ages)
```

'FOR' LOOPS WITH LISTS

A for loop can be used to access the elements in a list or other Python data structure one at a time:

Example

```
for num in numbers:
...     print(num)
...
1
2
3
```

Indentation

Indentation is very important in Python!
Note that the second line in the example is indented.

Just like three chevrons >>> indicate an interactive prompt in Python, the three dots ... are Python's prompt for multiple lines. This is Python's way of marking a block of code.

WHAT'S NEXT?

What does a script look like?

RESOURCES

Cynthia Heider

Public Digital Scholarship Librarian

Email: heider@upenn.edu
Book a consultation: www.library.upenn.edu/staff/cynthia-heider

At Penn:

- Materials from this workshop
- Python research guide
- “ENGL 1670: Data Science for the Humanities” offered in the spring semester
- “COMM 3180: Stories From Data: Programming for Data Journalism” offered in the spring semester
- Carpentries instruction for graduate students the week of Spring Break (last year’s event)
- Statistical software consultants

Elsewhere:

- PyCon
- PyLadies and Philly Python Users Group (Philly PUG)
- Self-directed Dataquest modules (sign up here)
- Guide to Programming for the Digital Humanities: Lessons for Introductory Python by Brian Kokensparger
- Exploratory Programming for the Arts and Humanities by (Penn alum) Nick Montfort
- Introduction to Cultural Analytics & Python by Melanie Walsh
- Humanities Programming syllabus from UVA

TAKE THE WORKSHOP SURVEY

Your feedback is important to us!
Please take our quick survey to
help us better serve your needs.

https://upenn.libwizard.com/f/rdds_survey_fa2025