

Computational Literature Using Python

Zhen Guo & Emily Sullivan

CRWT 1801

Professor Stephanie Young

Fall 2025

<https://bit.ly/fa25-young-crwt1801-pythonlit>

Agenda

- Different Methods for Computational Storytelling
- Introduction to Python and Google Colab
- Hands on: Using Computation in Writing
- Further Resources and Notes on Artificial Intelligence

Teaching Materials

Slides, Python notebooks, and class activities can be found in the shared Google Drive folder titled [FA25-PythonLit-StudentAccess](#) at:

<https://bit.ly/fa25-young-crwt1801-pythonlit>

To run the Python notebooks in Google Colab you will need to sign into Google Drive.

Different Methods for Computational Storytelling

Algorithms

- “[a] set of instructions that is designed to accomplish a task” ([National Library of Medicine](#)).
- A recipe for baking cookies, code to tell whether a picture is of a cat or a dog, and code to write a poem about a cat are examples of algorithms

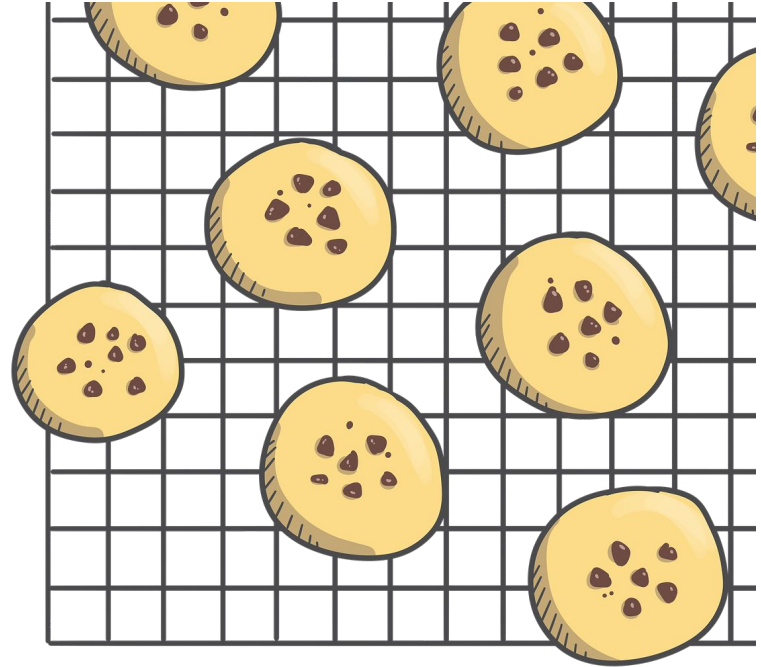


Image by wixin_56k, [Pixabay](#)

Example: Verse by Verse

- Google [Verse by Verse](#)
 - Uses a generative model to create lines of poetry
 - Uses a [semantic model](#) to determine the best next line of poetry
- About Google [Semantic Experiences](#)

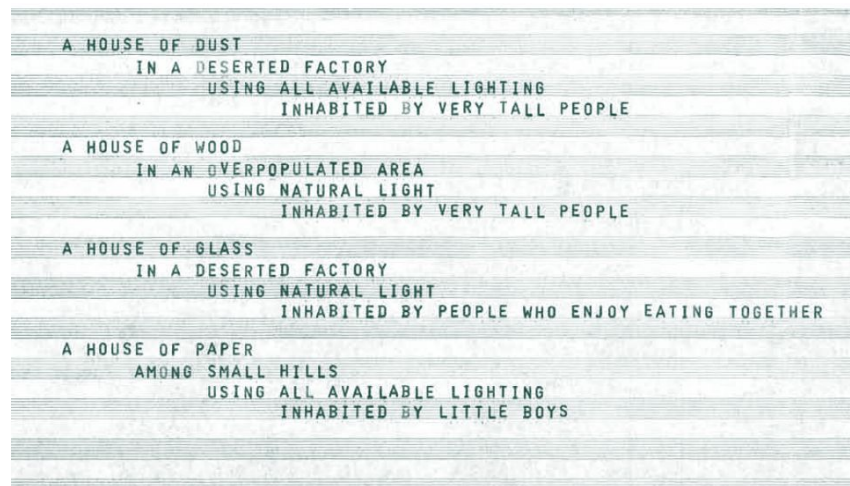


Sample of poets whose works are included in [Verse by Verse](#)

Feel free to ask questions at any point during the presentation!

Example: “The House of Dust”

- [Poem](#) by Alison Knowles and James Tenney (1967)
- Code reimplemented in Python by Nick Montfort and updated as teaching example: [House of Dust Example](#)



A HOUSE OF DUST
IN A DESERTED FACTORY
USING ALL AVAILABLE LIGHTING
INHABITED BY VERY TALL PEOPLE

A HOUSE OF WOOD
IN AN OVERPOPULATED AREA
USING NATURAL LIGHT
INHABITED BY VERY TALL PEOPLE

A HOUSE OF GLASS
IN A DESERTED FACTORY
USING NATURAL LIGHT
INHABITED BY PEOPLE WHO ENJOY EATING TOGETHER

A HOUSE OF PAPER
AMONG SMALL HILLS
USING ALL AVAILABLE LIGHTING
INHABITED BY LITTLE BOYS

Printout of “[The House of Dust](#),”
Gebr König Verlag, Cologne,
1967

Example: “Sea and Spar Between”

Textual remix/deformance

How to Read Sea and Spar Between:

“The words in Sea and Spar Between come from Emily Dickinson’s poems and Herman Melville’s Moby Dick.”

-by Nick Montfort and Stephanie Strickland

cut to fit the bardlurk course
nailed to the rail

loose-fish
cureless sing and dance —

2334620 : 13333306

loose-fish
for enchantless is the sky

one wind one dust one year one rose
paradise! perturb!

cut to fit the bardmilk course
nailed to the room

circle on
cureless sing and go —

Sea and Spar Between

*Feel free to ask questions at any point
during the presentation!*

Generative AI & Computational Writing

Compare the 1) poetry generated using a combination of existing poetry and AI and your own compositional decisions in [Verse by Verse](#) to the 2) computationally determined poems like [“House of Dust”](#) and [“Sea and Spar Between”](#)

- How does computational poetry differ from poetry written using generative AI?

Introduction to Python and Google Colab

Python Summary

The Python code in this workshop covers these topics:

Data

- Variables
- Data types
 - Strings
 - Lists
- Selecting data from lists and dictionaries

Functions

- Built-in functions
- Importing modules
 - Importing functions

Conditionals

- If/else statements

Loops

- While loop

Variables

Variables are created when you assign a value to them. The value can be **numbers** or **strings**.

E.g. `x = 5, y = "Hello, world!"`

Strings

Strings in python are sequence of characters surrounded by either single quotation marks, or double quotation marks.

`'hello'` is the same as `"hello"`.

Lists

Lists are used to store multiple items in a single variable. Items can be duplicated.

E.g. `fruit_list = ["apple", "banana", "cherry", "apple"]`

Selecting Data from Lists

- List items are ordered, changeable, and allow duplicate values.
- List items are indexed, the first item has index `[0]`, the second item has index `[1]` etc.

Code `thislist = ["apple", "ball", "cat"]`
`print(thislist[0])`

Result `apple`

Functions

- A function is a block of code which only runs when it is called.
- You can pass data into a function.
- A function can return data as a result.
- For example, you can display a string with the `print()` function:
- E.g. `print("Hello")`

Python modules

- A module contains functions that someone else has written that you can use.
- When we want to use a module, we import it. For example, we are going to use the **random** module.
 - Code: `import random`
- We want to use a particular function, **choice()**, which gives us a random choice from a list or other data structure. We can import this function from the **random** module.
 - Code: `from random import choice`

Conditionals

- Conditionals can be used to execute code if a condition is true
- To check if a condition is true use `==`
- Use `if`, `elif`, and `else` to designate the code you want to execute

```
Code    color = "pink"
        if color == "pink":
            print("the the color is pink")
        elif color == "blue"
            print("the the color is blue")
        else
            print("the color is not pink or blue")

Result  the color is pink
```

Loops

- Loops execute code repeatedly
- A **while** loop executes code while a condition is true
- A while loop needs to reach a point where the condition is not longer true, or it will continue running

```
Code    while condition == True:
        print("the condition is true")
        condition = False
```

```
Result  the condition is true
```

Important notes about coding

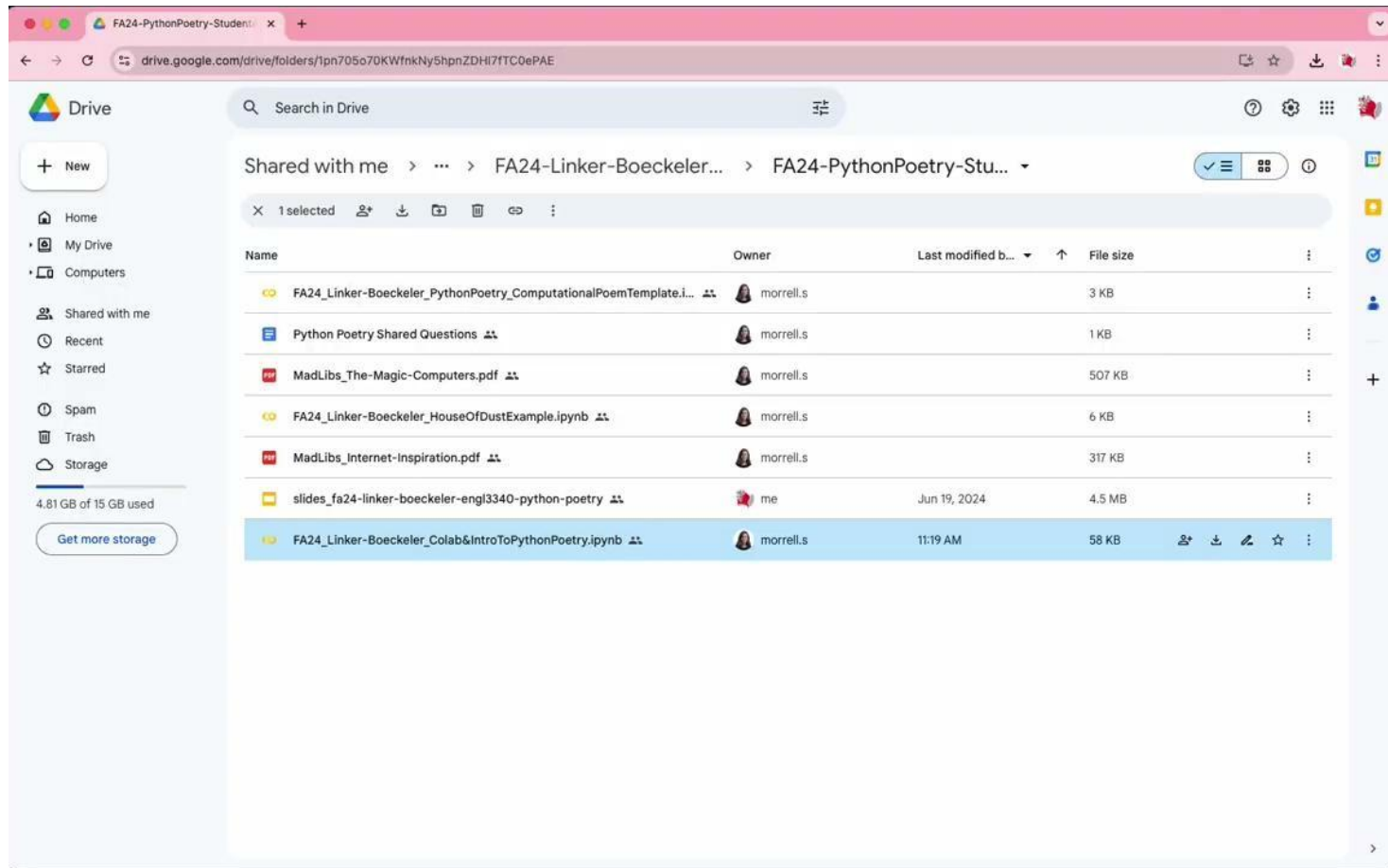
- Capitalization matters
- Spaces and indentations matter
- Spelling matters
- Missing one quote mark/parentheses, mispairing quote marks matter
- If your code doesn't run, check the above first.
- It is okay to make mistakes

Python Google Colab Notebooks

[Intro To Python Poetry](#): This notebook introduces the fundamentals of Python and provides example code for creating computational poetry. The notebook can be accessed by:

- Signing in to your Google account
- Clicking the above link or navigating to the notebook in the [shared Google Drive folder](#) (notebook file name is FA25_Young_Colab&IntroToPythonLit.ipynb)
- Copying the notebook to your Google Drive by selecting 'Save a copy in Drive' under the 'File' menu in the upper left corner of the notebook

Install Google Colab



Hands-On: Using Computation in Prose or Poetry

Student Examples: Computational Prose

- Original writing: [Funny Letter Example](#)
- Remixing literature: [Brontean Letter Example](#)

Reflection Questions: How do you think computational remixing changes across genres? Which of these examples do you think worked better and why?

You Try! (Prose)

1. Make a **copy** of either the [funny letter](#) or [Brontean letter](#) code
2. Make a change to the inputs in each of the text lists
3. Make at least **one** change to the story **structure**, either by adding a new list for an extra line, swapping the order of story elements, etc., and see what comes out!
 - a. You can either use the text provided in the examples throughout this module or input your own generated in class

You Try! (Poetry)

1. Make a copy of either the [House of Dust Introductory](#) notebook or the [House of Dust Advanced](#) notebook.
2. Make a change to the inputs in each of the text lists.
3. Make at least **one** change to the poem **structure**, either by adding a new list for an extra line, swapping the order of poem elements, etc., and see what comes out!
4. Try adding a pause or user input to the poem.

Post-exploration group reflection

- Do you have any reflections on using Python for constructing poetry/prose?
- How does the writing process differ from traditional storytelling?
- How might the code impact the readers' perception of the poem?
- If you could design any creative writing tool using computational methods, what type of tool would want to design?
- How might you use this in the future?

Further Resources and Notes on Artificial Intelligence

Discussion: Generative AI in Computational Writing

- You can prompt GenAI to generate sentences, paragraphs, or complete code for your computational writing
- It is important to verify output and understand generated code
- How might using GenAI shape your computational writing?
 - Is it still **your** writing?
- What are the risks associated with using GenAI?
- If GenAI can write code, why might we still want to learn to code ourselves?

Generative AI Example

- Example prompt:
“Fill in the options with fun poem lines. Here is the code: [your code]”
- Result:

```
poetry_lines = {  
    'space_adventures': {  
        'line1': [  
            "Rocket ships zoom past stars with a grin,",  
            "Through Saturn's rings we bravely spin,"  
        ],  
        'line2': [  
            "Aliens wave as we fly by their moon,",  
            "We dance with comets to a Martian tune."  
        ]  
        # To add more lines, you could add 'line3': ["option1", "option2"]  
    },  
    'lazy_sundays': {  
        'line1': [  
            "Sunlight spills across my cozy bed,",  
            "Pancakes stack higher than my sleepy head,"  
        ],  
        'line2': [  
            "I drift in dreams while the coffee brews,",  
            "Wrapped in a blanket, I snooze and snooze."  
        ]  
        # Add 'line3', 'line4', etc., for a longer poem  
    }  
}  
# To add more topics, just add another key like 'underwater_worlds': {...}
```



AI Ethics Resources

- [The Institute for Experiential AI](#) at Northeastern
- United Nations Educational, Scientific and Cultural Organization (UNESCO) [Global Forum on the Ethics of AI 2024](#)

Image found in [Changing the Landscape of AI Governance](#), UNESCO



Further Exploration:

How might you further randomize stories in Python? Here are some more resources to help you get started:

- [10 Fun Python Scripts That Generate Random Stories](#) by *Medium*
- [Code Club: Story Time](#) by *Raspberry Pi Foundation*
- [How to Build a Random Story Generator Using Python](#) by *Geeks for Geeks*

Handouts

- [Writing Functions in Python](#)
- [Troubleshooting Python](#)

Thank you!

—Developed by Sara Morrell, Dipa Desai, Kasya O'Connor Grant, Avery Blankenship, Claire Lavarreda, Emily Sullivan, and Zhen Guo

- For more information on the DITI, please see: <https://bit.ly/diti-about>
- Schedule an appointment with us! <https://bit.ly/diti-meeting>
- If you have any questions, contact us at: nulab.info@gmail.com
- We'd love your feedback! Please fill out a short survey here: <https://bit.ly/diti-feedback>