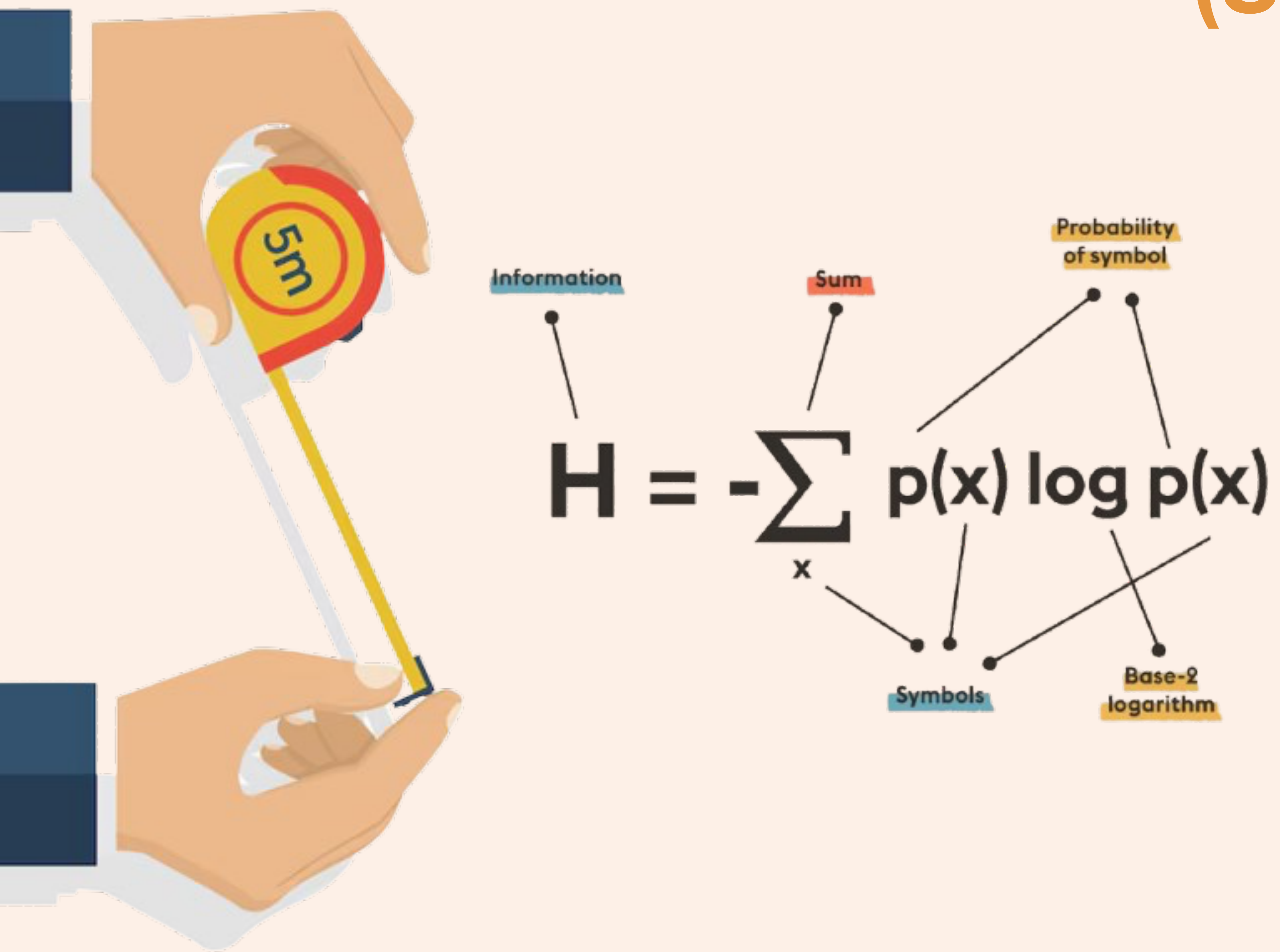
BINGHAMTON UNIVERSITY | THOMAS J. WATSON COLLEGE OF ENGINEERING AND APPLIED SCIENCE

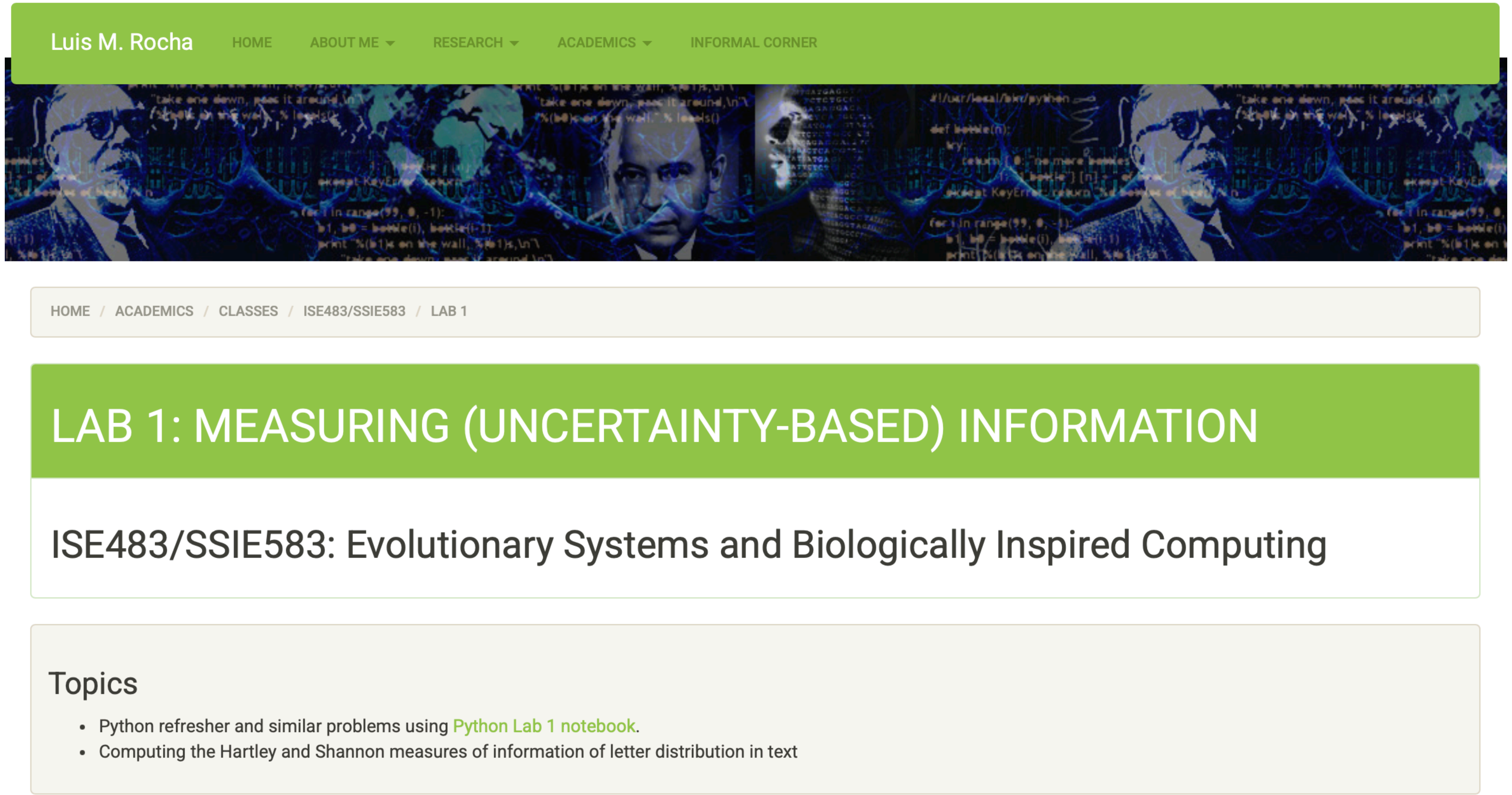
SSIE 483/583 Lab 1

MEASURING INFORMATION (UNCERTAINTY-BASED)

Shayan Esfarayeni



Class Website



LAB 1: MEASURING (UNCERTAINTY-BASED) INFORMATION

ISE483/SSIE583: Evolutionary Systems and Biologically Inspired Computing

Topics

- Python refresher and similar problems using Python Lab 1 notebook.
- Computing the Hartley and Shannon measures of information of letter distribution in text

https://casci.binghamton.edu/academics/i-bic/lab1/

Questions Summary

1. Write functions to calculate Shannon entropy (based on a probability distribution) and Hartley entropy. Use `numpy` to handle probability arrays and perform calculations. Additionally, create small probability distributions, plot them, and analyze how entropy reflects the organization of symbols.

2. Write a program to calculate letter entropy from a text file by reading the file, counting the frequency of each English letter and space, normalizing these counts into a probability distribution using a `numpy` array, and then calculating both Shannon and Hartley entropy. Also, include an analysis of "Gadsby," a lipogram novel that excludes the letter "E," and compare its entropy to a normal text. Additionally, generate a random text of similar length with uniform letter distribution, compute its entropy, and compare it to the original text’s entropy to analyze differences in information measures.

IDE Options

Jupyter Notebook

- Simple, web-based interface ideal for data visualization and exploratory analysis.
- Cell-based execution of code, Markdown, and HTML.
- Limited UI, one notebook per tab.

JupyterLab

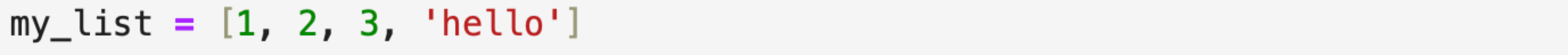
- Advanced version of Jupyter Notebook with a web-based IDE interface.
- Supports multiple files and types simultaneously, including notebooks, text files, and terminals.
- More complex UI customizable for comprehensive projects.

Visual Studio Code (VS Code)

- Extensive code editor with powerful extension ecosystem for various programming needs.
- Integrated support for Git, terminal, debugging, and Jupyter notebooks via extensions.
- Versatile for both software development and data science with additional setup.

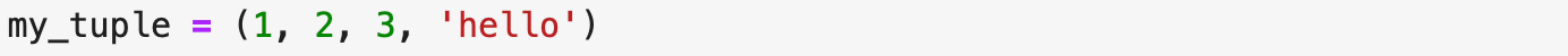
Python Data Structures

List: Mutable and ordered, suitable for dynamic collections of items



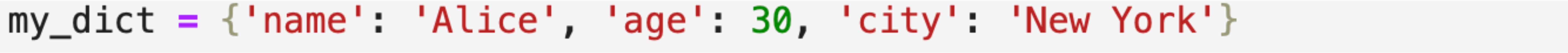
```
my_list = [1, 2, 3, 'hello']
```

Tuple: Immutable and ordered, ideal for fixed data storage.



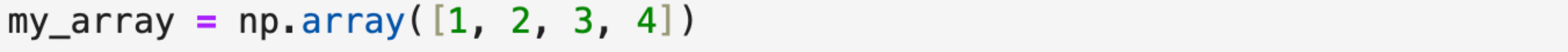
```
my_tuple = (1, 2, 3, 'hello')
```

Dictionary (Dict): Unordered with key-value pairs, best for quick data retrieval.



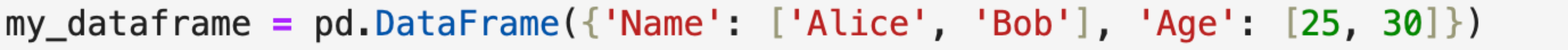
```
my_dict = {'name': 'Alice', 'age': 30, 'city': 'New York'}
```

Array (NumPy): Homogeneous and efficient, optimized for high-speed numeric operations.



```
my_array = np.array([1, 2, 3, 4])
```

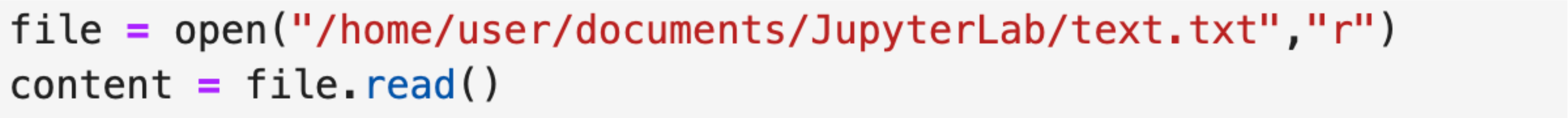
DataFrame (pandas): Two-dimensional and mutable, designed for complex data manipulation and analysis.



```
my_dataframe = pd.DataFrame({'Name': ['Alice', 'Bob'], 'Age': [25, 30]})
```

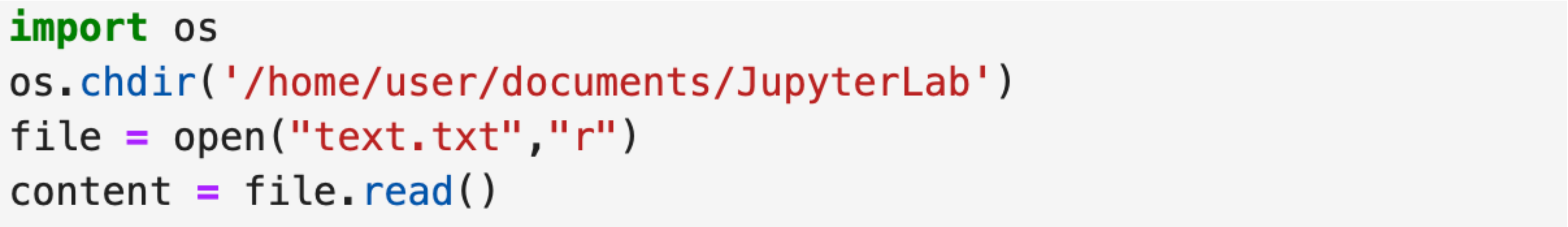
Accessing Files in Python

1. Using the Full Address of the File.



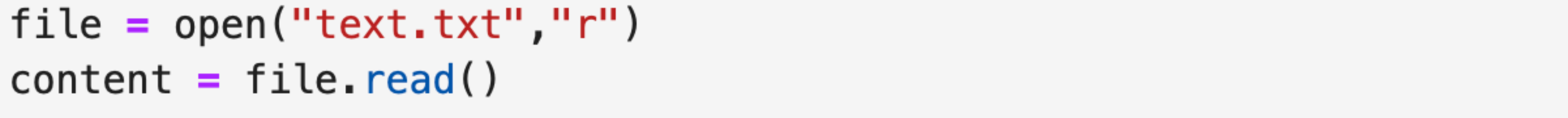
```python
file = open("/home/user/documents/JupyterLab/text.txt","r")
content = file.read()
```

2. Using os.chdir() to Change the Working Directory.



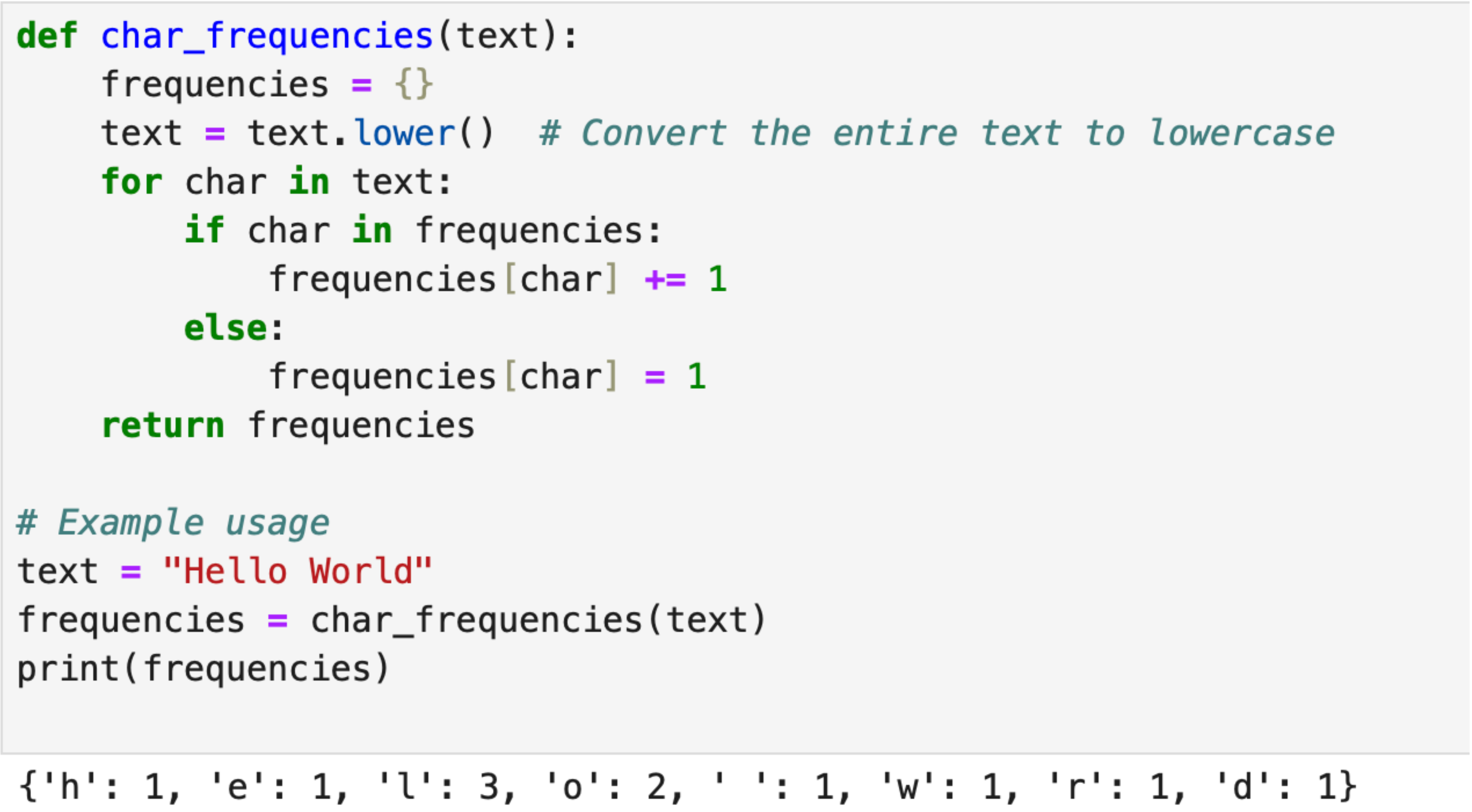
```python
import os
os.chdir('/home/user/documents/JupyterLab')
file = open("text.txt","r")
content = file.read()
```

3. Copying the File to the Current Directory of the Python script.



```python
file = open("text.txt","r")
content = file.read()
```

Character Frequencies

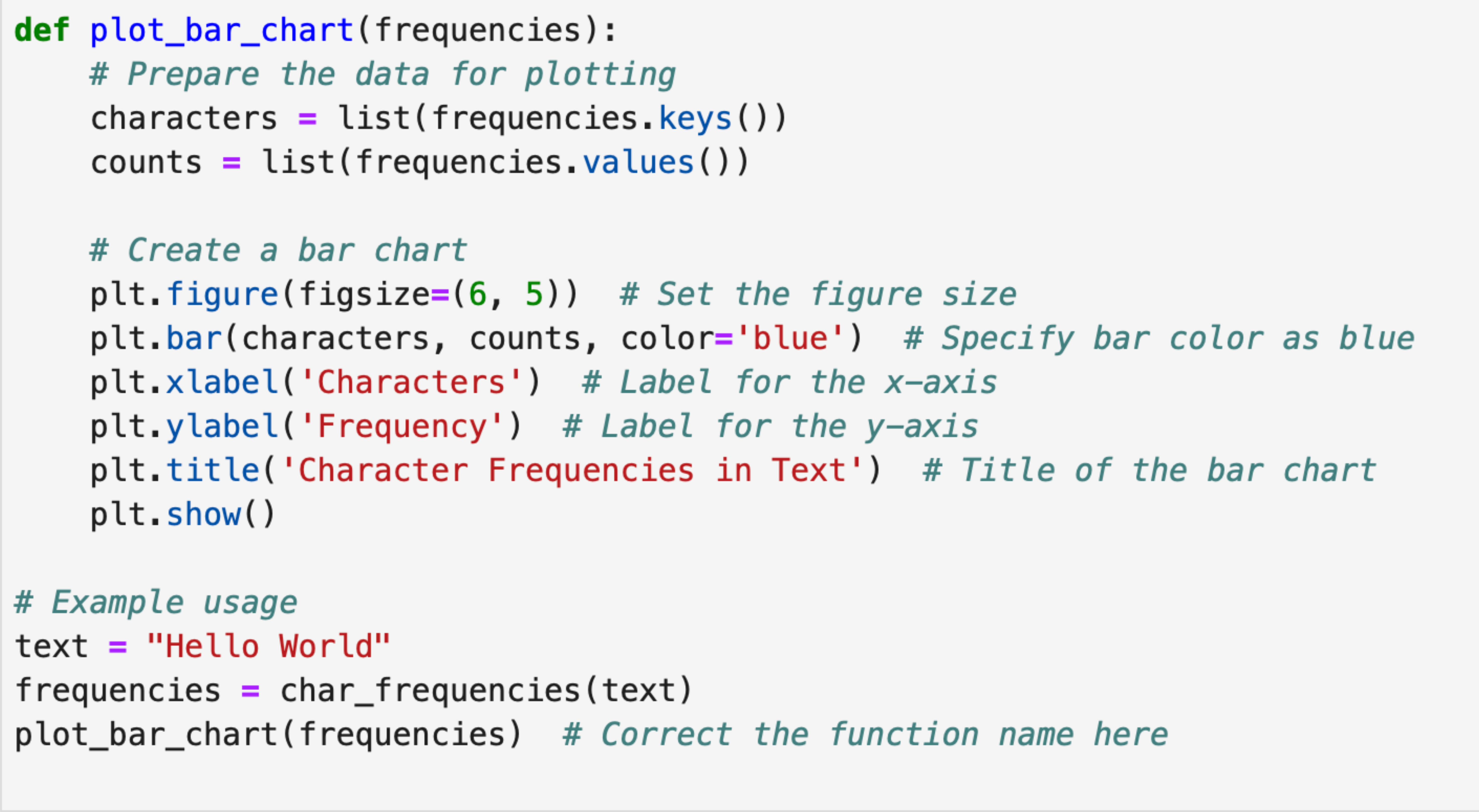


```python
def char_frequencies(text):
    frequencies = {}
    text = text.lower()  # Convert the entire text to lowercase
    for char in text:
        if char in frequencies:
            frequencies[char] += 1
        else:
            frequencies[char] = 1
    return frequencies

# Example usage
text = "Hello World"
frequencies = char_frequencies(text)
print(frequencies)
```

```
{'h': 1, 'e': 1, 'l': 3, 'o': 2, ' ': 1, 'w': 1, 'r': 1, 'd': 1}
```

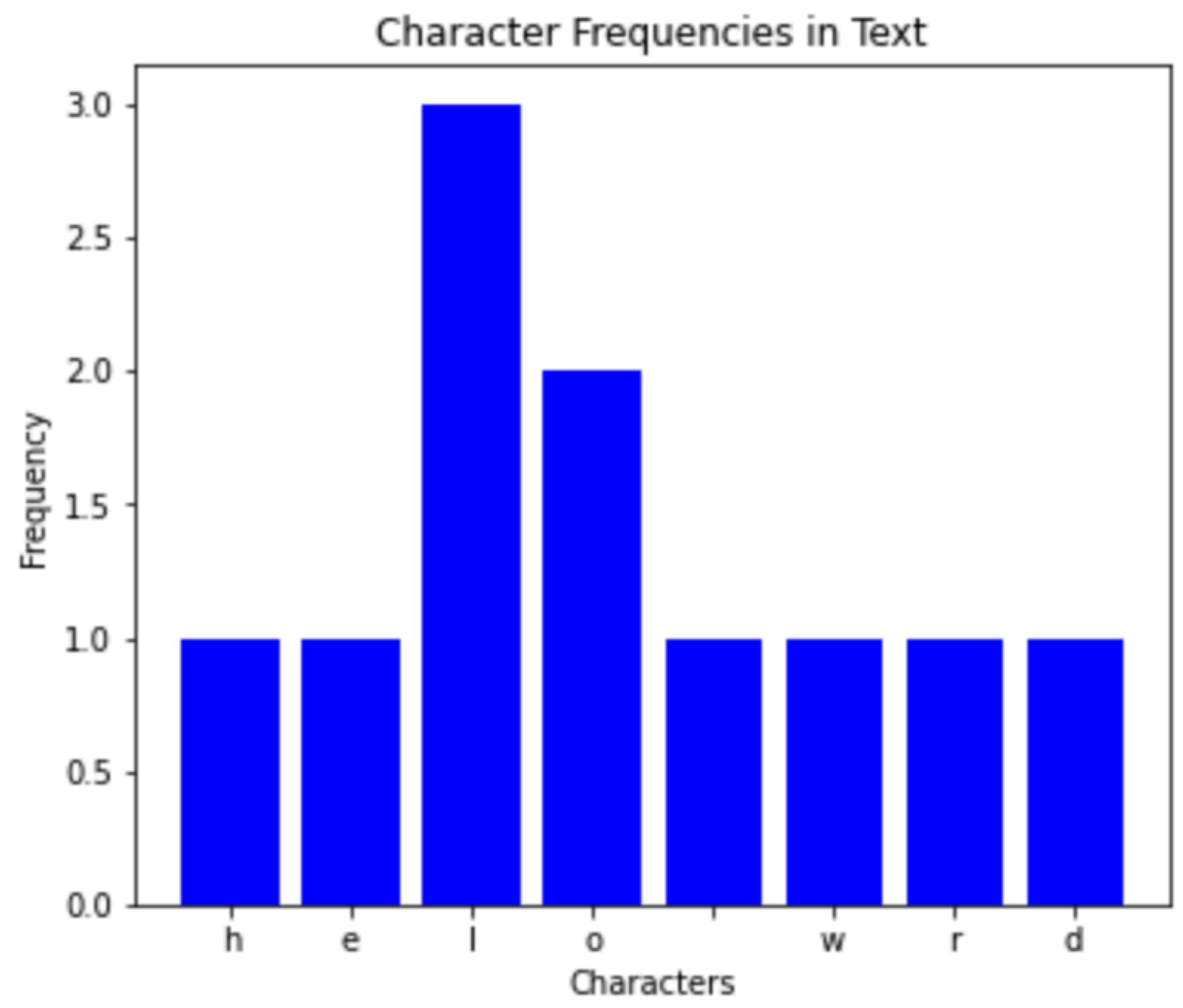
Plotting Characters Frequencies



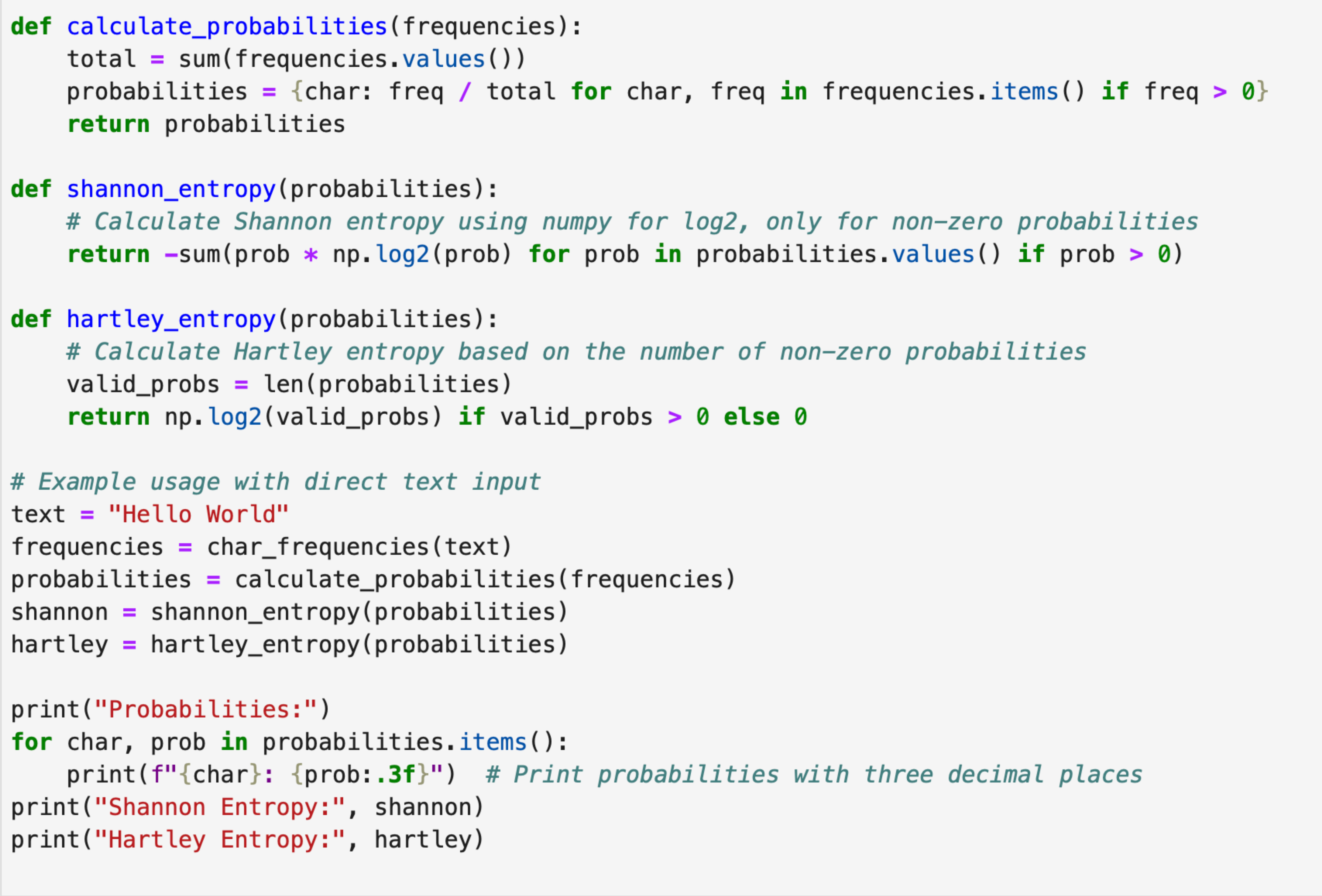
```python
def plot_bar_chart(frequencies):
    # Prepare the data for plotting
    characters = list(frequencies.keys())
    counts = list(frequencies.values())

    # Create a bar chart
    plt.figure(figsize=(6, 5))  # Set the figure size
    plt.bar(characters, counts, color='blue')  # Specify bar color as blue
    plt.xlabel('Characters')  # Label for the x-axis
    plt.ylabel('Frequency')  # Label for the y-axis
    plt.title('Character Frequencies in Text')  # Title of the bar chart
    plt.show()

# Example usage
text = "Hello World"
frequencies = char_frequencies(text)
plot_bar_chart(frequencies)  # Correct the function name here
```



Entropy Calculation



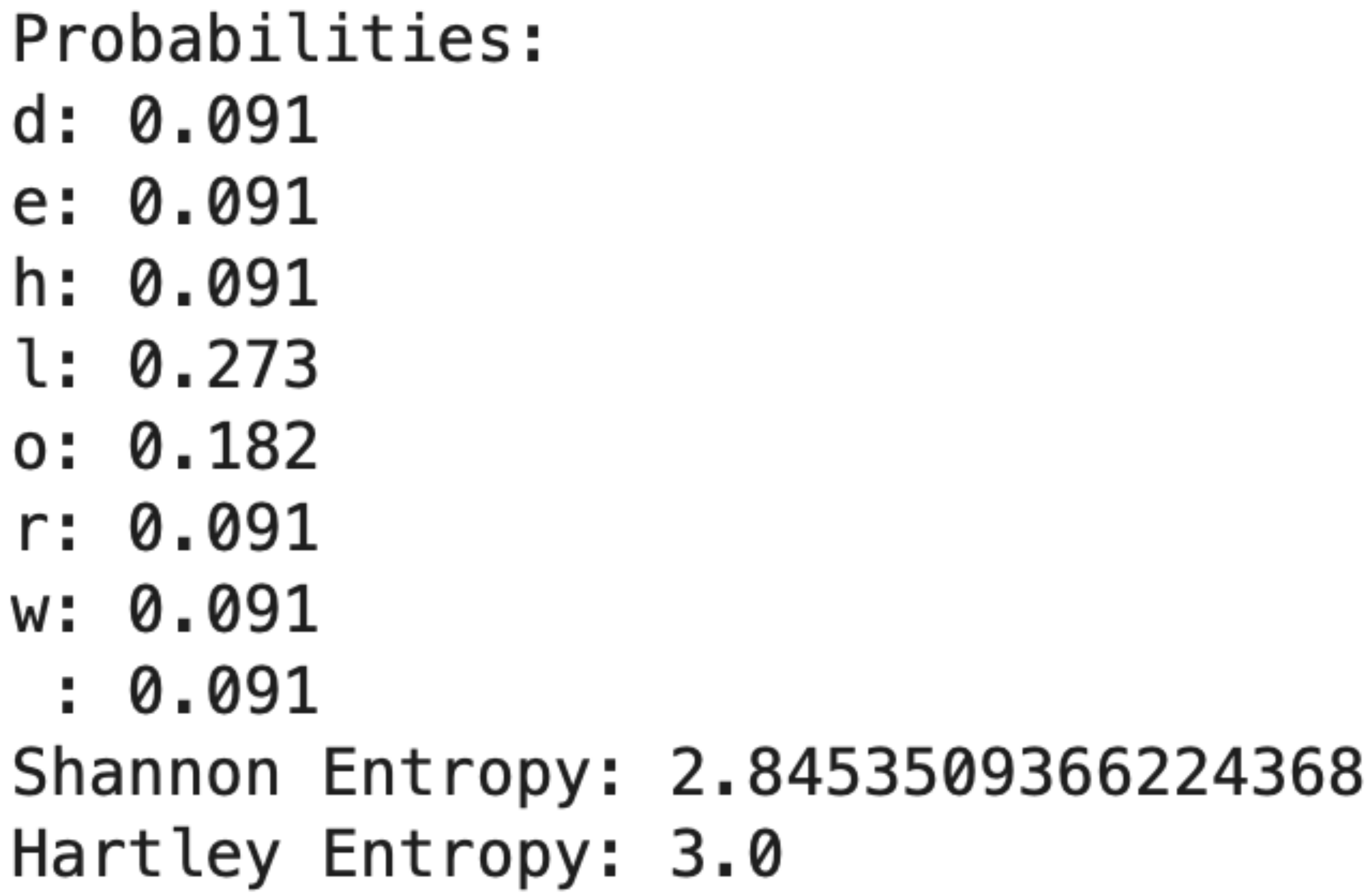
```python
def calculate_probabilities(frequencies):
    total = sum(frequencies.values())
    probabilities = {char: freq / total for char, freq in frequencies.items() if freq > 0}
    return probabilities

def shannon_entropy(probabilities):
    # Calculate Shannon entropy using numpy for log2, only for non-zero probabilities
    return -sum(prob * np.log2(prob) for prob in probabilities.values() if prob > 0)

def hartley_entropy(probabilities):
    # Calculate Hartley entropy based on the number of non-zero probabilities
    valid_probs = len(probabilities)
    return np.log2(valid_probs) if valid_probs > 0 else 0

# Example usage with direct text input
text = "Hello World"
frequencies = char_frequencies(text)
probabilities = calculate_probabilities(frequencies)
shannon = shannon_entropy(probabilities)
hartley = hartley_entropy(probabilities)

print("Probabilities:")
for char, prob in probabilities.items():
    print(f"{char}: {prob:.3f}")  # Print probabilities with three decimal places
print("Shannon Entropy:", shannon)
print("Hartley Entropy:", hartley)
```



```
Probabilities:
d: 0.091
e: 0.091
h: 0.091
l: 0.273
o: 0.182
r: 0.091
w: 0.091
 : 0.091
Shannon Entropy: 2.84535093662243768
Hartley Entropy: 3.0
```

Implementation Guidelines

Use of Data Structures: The examples in the notebook used **dictionaries**, but the lab instructions required **arrays** instead.

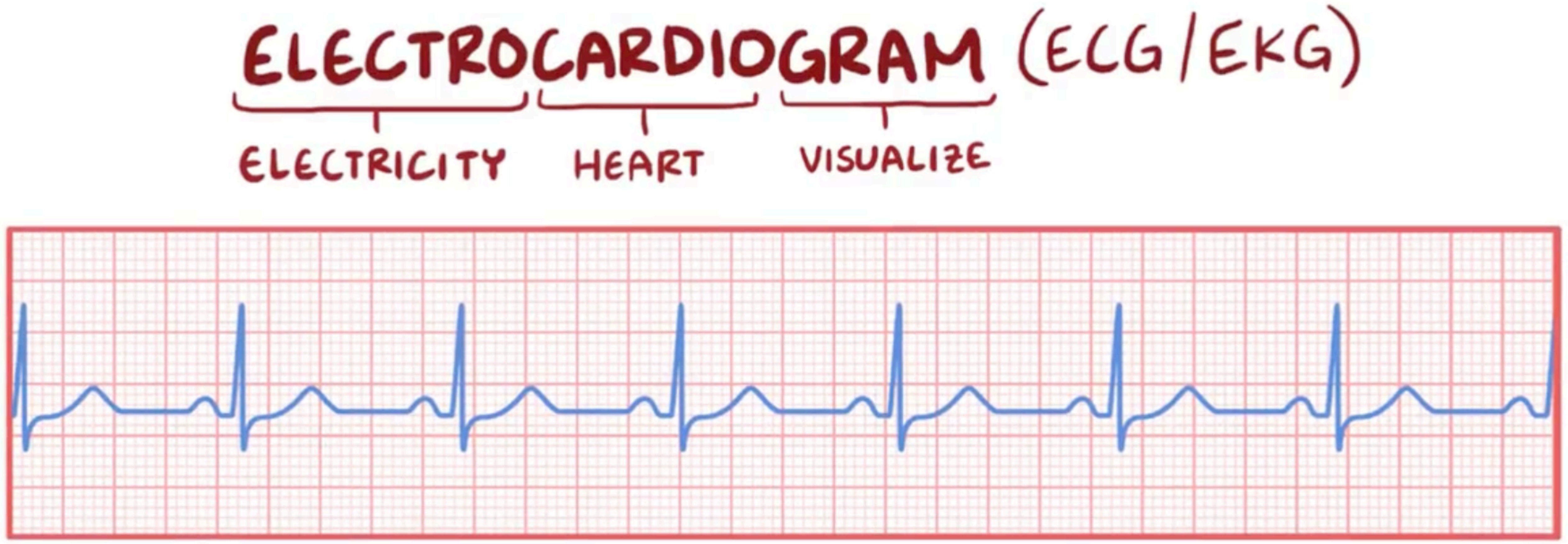
Character Restrictions: Only **26 English letters plus space** should be considered; all other characters should be ignored in the text.

Questions Summary

3. If you scramble the letters of the meaningful text and then measure its Shannon entropy, it will be the same as the original. Yet, the scrambled text is gibberish. How you would extend the standard Shannon entropy formula so that it may distinguish meaningful text from its letter-shuffled versions.

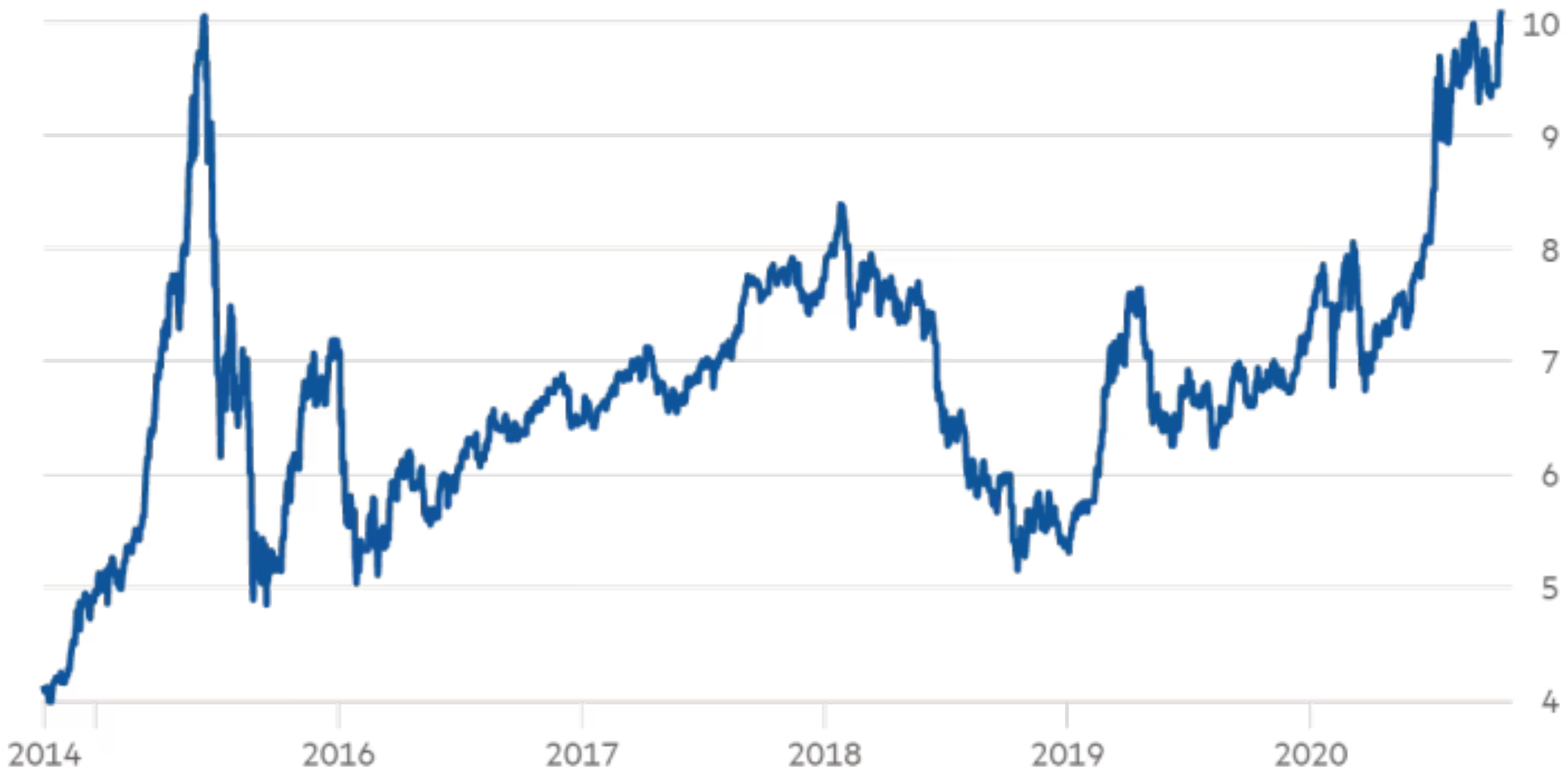
4. Calculate Shannon entropy for an infant who knows four alphabet symbols ({a, b, c, d}) with specific probabilities ('a' at 1/2, 'b' at 1/4, and 'c' and 'd' each at 1/8). Use this entropy to design an optimal sequence of yes-no questions for the infant's father, who can't hear her but wants to guess the symbol she's using by asking the mother. Report the sequence and explain the rationale behind its efficiency in matching the calculated entropy.

Applications



Using Shannon entropy for **anomaly detection in ECG data** is a promising approach, particularly useful in health monitoring systems for early detection of cardiac anomalies. It provides a quick, effective way to signal deviations from normal heart activity, potentially aiding in timely medical interventions.

Applications



Monitoring Entropy Over Time: By tracking how entropy values change over time, analysts can identify periods where the market behaves unpredictably or deviates from typical patterns. A significant increase in entropy might indicate a market transition or the beginning of a new trend. However, like all analytical tools, it should be used in conjunction with other methods to form a well-rounded trading strategy.

BINGHAMTON UNIVERSITY | THOMAS J. WATSON COLLEGE OF ENGINEERING AND APPLIED SCIENCE

SSIE 483/583 Lab 1

MEASURING INFORMATION (UNCERTAINTY-BASED)

Shayan Esfarayeni
sesfarayeni@binghamton.edu