Enough Python to Do Something Useful With

Chris Woodley

Session 2 – Numpy and Pandas

Scope of this Session

1. Solution of Previous Exercise
2. Installation and setup for this session
3. Data i/o in python
4. Numpy
5. Pandas
6. Self Guided Python skills session
   - Text i/o, Numpy and Pandas examples
   - Numpy/Pandas exercises

1. Molecular Weight Calculator

```
def calc_molecular_Weight(formula):
    atoms = []
    numbers = []

    for i in range(len(formula)):

        if formula[i].isupper():
            current_atom = formula[i]
            if i + 1 < len(formula):
                if formula[i+1].islower():
                    current_atom += formula[i+1]
                    if i + 2 < len(formula):
                        if formula[i+2].isupper():
                            numbers.append(1)
                    else:
                        numbers.append(1)
                elif formula[i+1].isupper():
                    numbers.append(1)
            else:
                numbers.append(1)

            atoms.append(current_atom)

        if formula[i].isdigit() and formula[i-1].isalpha():
            current_number = formula[i]
            if i+1 < len(formula):
                if formula[i+1].isdigit():
                    current_number += formula[i+1]
            numbers.append(int(current_number))

    molecular_weight = 0.
    for i in range(len(atoms)):
        molecular_weight += atomic_weights[atoms[i]]*numbers[i]

    return molecular_weight
```

Function declaration

Initialise list

Parse formula

Calculate molecular weight

Calculate molecular weight

- Exercise 4 from previous session
- “Write a function to parse a string of a molecular formula containing two-letter elements”
- Compared to previous exercises this requires parsing of the characters around the current character

- Like previous answers we split the solution into two parts: collecting atoms and numbers, and calculating the molecular weight

1. Molecular Weight Calculator

```
atoms = []
numbers = []

for i in range(len(formula)):

    if formula[i].isupper():
        current_atom = formula[i]
        if i + 1 < len(formula):
            if formula[i+1].islower():
                current_atom += formula[i+1]
                if i + 2 < len(formula):
                    if formula[i+2].isupper():
                        numbers.append(1)
                else:
                    numbers.append(1)
            elif formula[i+1].isupper():
                numbers.append(1)
        else:
            numbers.append(1)

        atoms.append(current_atom)

    elif formula[i].isdigit() and formula[i-1].isalpha():
        current_number = formula[i]
        if i+1 < len(formula):
            if formula[i+1].isdigit():
                current_number += formula[i+1]
        numbers.append(int(current_number))
```

Must be the start of an element so initialise a new atom

If next character is lowercase, this is a two-letter element

If next-next character is uppercase, there is only one of this element

If next character is uppercase, then single atom in molecule

Finished parsing atom

If character is a number, and the previous character is alpha – new number of elements

Check for two digit number

2. Anaconda Setup

- We need packages for today's notebooks
- Delete last weeks environment (not necessary every time)

conda env remove –n my_conda_env

- Create a new python environment

conda create –n new_conda_env

- Activate the environment

conda activate new_conda_env

- Install numpy, pandas and matplotlib

conda install numpy pandas matplotlib ipykernel

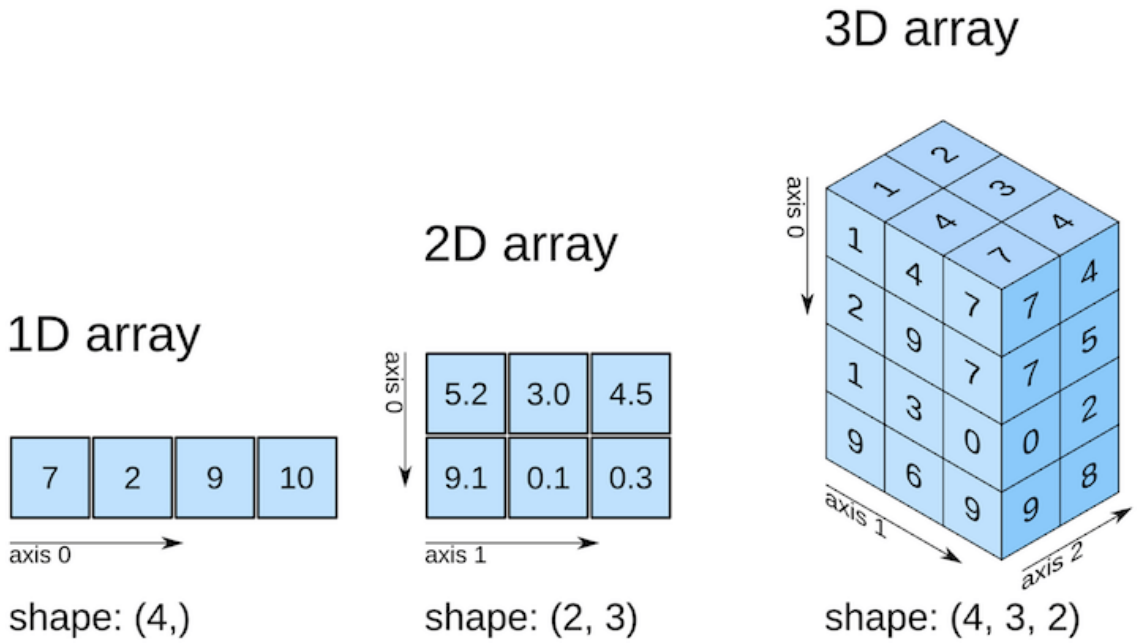
3. Data I/O in Python

- **Data I/O:** Essential for automating tasks, processing data, and saving results in various file formats.
  - Need efficient ways to get our data into python
  - We can use python to process data
  - Need efficient ways to get data out of python

- **Text Files:** Widely used for configuration files, logs, and plain data storage.
  - Why?:
    - Simple: Human-readable, easily edited and shared.
    - Flexible: Suitable for many formats (e.g., .txt, .csv, .pdb).
    - Portable: Can be used across platforms without special software.

4. Numpy

- **NumPy:** The Core Library for Numerical Computing in Python
  - Efficient Array Operations: Handles large multi-dimensional arrays and matrices.
  - Fast Mathematical Computations: Optimized for performance with vectorized operations.
- **What is a NumPy array?**
  - A **NumPy array** is like a **list of lists**, but:
    - More **efficient**.
    - Designed for **numerical data** and fast computations.
    - Supports multi-dimensional data (e.g., matrices, 3D arrays).
- **Key Functions of NumPy:**
  - Array Creation
  - Array Manipulation
  - Mathematical Operations
- **Things that can be stored in an array:**
  - Image data (2D or 3D array)
  - Financial data (1D array, timeseries)
  - Molecular data (2D array for coordinates)



5. Pandas

- **Pandas: The Core Library for Data Manipulation in Python**
  - **Data Handling and Analysis**: Pandas excels at managing structured data (e.g., tables, time series).
  - **Efficient DataFrames:** Optimized for working with large datasets in rows and columns.
- **What is a Pandas Dataframe?**
  - A **DataFrame** is like a **spreadsheet**:
    - Rows and columns of labeled data.
    - Indexing for easy access and manipulation.
    - Can handle different types in the same table.
- **Key Functions of Pandas:**
  - **Data Import/Export**: Read and write data from/to CSV, Excel, SQL, etc.
  - **Data Cleaning**: Handle missing values, filter rows, and modify columns.
  - **Data Aggregation**: Perform group operations, summarize data, and compute statistics.
- **Things that can be stored in a DataFrame:**
  - **Survey or Experiment Data**: Rows are participants/samples, columns are variables/measurements.
  - **Time Series Data**: Indexed rows by date, columns for different metrics.
  - **Financial Data**: Stock prices, trade volumes, or performance indicators across companies.

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no
e	dog	5.0	2	no
f	cat	2.0	3	no
g	snake	4.5	1	no
i	dog	7.0	2	no
j	dog	3.0	1	no

5. Troubleshooting/ Further Reading

- Official Python Documentation:
  - Comprehensive guide to Python's built-in functions, including file handling and I/O.
  - https://docs.python.org/3/tutorial/inputoutput.html
- NumPy Official Documentation:
  - Detailed reference for array creation, manipulation, and mathematical functions.
  - https://numpy.org/devdocs/user/
- Pandas Official Documentation:
  - A go-to resource for everything related to DataFrames, data manipulation, and importing/exporting data.
  - https://pandas.pydata.org/docs/user_guide/index.html
- Google your issues – find a relevant stack overflow post.
- ChatGPT can find the solution to the exercises:
  - **Try not to use this –** manually troubleshooting is a good way to learn python