Assignment #1:

Converting a Data Wrangling Task into a Snakemake Workflow

Objective:

Transform a current data wrangling notebook or example into a reusable Snakemake workflow that can be applied to any input data. This task will focus on creating an efficient and modular pipeline that automates your data processing.

Instructions:

1. **Select your task:**
    - Take a current data wrangling task you've worked on, either from your rotation project or an example exercise. The task should involve manipulating or processing data.
    - Your goal is to transform this into a Snakemake workflow, making it applicable to any relevant input data.
2. **Design the pipeline:**
    - Break down your current notebook into discrete steps that can be implemented as rules in Snakemake. Think about how to structure the pipeline:
        1. Identify the input files.
        2. Decide on the key steps that manipulate or analyze the data.
        3. Define the output that you want to produce from the input data.
    - *You may choose to implement ALL steps of an existing data processing task, but we are just looking for you to implement one step.*
3. **Implement the workflow:**
    - Write a `Snakefile` that automates your pipeline. This should include:
        1. Multiple rules for the key step in your data wrangling process.
        2. [Optional] The use of wildcards to handle input/output file names flexibly.
        3. [Optional] Appropriate input/output declarations in each rule to create dependencies between tasks.
4. **Tools and libraries:**
    - You can use any command-line tools or R/Python packages relevant to your workflow.
    - Remember to integrate necessary dependencies for your workflow, such as specific packages or tools for data wrangling (e.g., `dplyr`, `pandas`).
5. **Resource management:**
    - Define resource management parameters where relevant in your rules:
        1. Use `threads` to parallelize steps where applicable.
        2. If your pipeline is likely to handle large files, specify memory and compute requirements where necessary.
6. **Local Execution:**

    - Run the pipeline locally on your machine using Snakemake. Ensure that all steps work correctly from start to finish when applied to your input data.
7. **Deliverables:**
    - **Snakefile**: Your complete Snakemake workflow script.
    - **Command line**: The command line argument to run your workflow
    - **Brief report (½ to 1 page)**: Include the following details:
        1. **Input/output description**: Explain the input data structure and what the expected outputs are.
        2. **Pipeline purpose**: Describe what your pipeline is doing—what problem is it solving, or how is it transforming the input data?
        3. **Tools used**: List the command-line tools, R/Python libraries, or other dependencies implemented in your workflow.

Tips:

- Think about generalizing the workflow to handle similar types of input data beyond your own, so it can be applied in various contexts.
- Make use of Snakemake's powerful wildcard and rule features to structure your pipeline efficiently.
- Consider the real-world utility of your pipeline. What benefit does automating this process provide?

Resources:

- Snakemake documentation