Task Description

Your task is to write a Python program to fetch research papers based on a user-specified query. The program must identify papers with at least one author affiliated with a pharmaceutical or biotech company and return the results as a CSV file.

Problem Details

1. **Source of Papers**
   - Fetch papers using the **PubMed** API
   - The program should support **PubMed's full query syntax** for flexibility.
2. **Output Requirements**
   - Return the results as a CSV file with the following columns:
     - **PubmedID**: Unique identifier for the paper.
     - **Title**: Title of the paper.
     - **Publication Date**: Date the paper was published.
     - **Non-academic Author(s)**: Names of authors affiliated with non-academic institutions.
     - **Company Affiliation(s)**: Names of pharmaceutical/biotech companies.
     - **Corresponding Author Email**: Email address of the corresponding author.
3. **Command-line Program Features**
   - Accept the query as a command-line argument.
   - Provide the following options:
     - `-h` or `--help`: Display usage instructions.
     - `-d` or `--debug`: Print debug information during execution.
     - `-f` or `--file`: Specify the filename to save the results. If this option is not provided, print the output to the console.
4. **Code Organization and Environment**
   - **Version Control**:
     - Use Git for version control. The code must be hosted on **GitHub**.
   - **Dependencies and Setup**:
     - Use **Poetry** for dependency management and packaging.
     - Ensure that running `poetry install` sets up all dependencies.
   - **Execution**:
     - Provide an executable command named `get-papers-list` via Poetry.
5. **Documentation**
   - Include a `README.md` file with the following details:
     - How the code is organized.

        - Instructions on how to install dependencies and execute the program.
        - Mention any tools (e.g., LLMs or libraries) used to build the program, along with relevant links.

6. **Evaluation Criteria**

    - **Functional Requirements**:
        - Adherence to the problem statement.
        - Ability to fetch and filter results correctly.
    - **Non-functional Requirements**:
        - Typed python: Using types everywhere.
        - Performance: Efficiency of API calls and processing.
        - Readability: Clear and maintainable code with appropriate comments and docstrings.
        - Organization: Logical separation of concerns (e.g., modular functions and classes).
        - Robustness: Error handling for invalid queries, API failures, or missing data.

Bonus points

Each of these additional points

1. Break the program into two parts: a module and a command line program that uses the module.
2. Publish the module in test-pypi.

Notes

- You are free to use **LLM tools** or other resources to assist in development – please s
- Clearly document any external tools used in the `README.md`.
- Assume the program will be evaluated by automated scripts, so strict adherence to conventions is required.
- How to identify non-academic authors? You can apply any heuristics (email addresses, words like university, labs etc).