# Research Paper Fetcher - Python Program

The following Python program fetches research papers based on a user-specified query, identifies papers
with at least one author affiliated with a pharmaceutical or biotech company, and returns 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.

Code:

```python
import argparse
import pandas as pd
from Bio import Entrez
import re

def fetch_papers(query: str, max_results: int = 20) -> list:
    Entrez.email = "your-email@example.com"  # Replace with your email address
    handle = Entrez.esearch(db="pubmed", term=query, retmax=max_results, usehistory="y")
    record = Entrez.read(handle)
    handle.close()
    return record["IdList"]

def fetch_paper_details(pubmed_ids: list) -> list:
```

```
    ids = ",".join(pubmed_ids)
    handle = Entrez.esummary(db="pubmed", id=ids)
    record = Entrez.read(handle)
    handle.close()
    return record

def filter_non_academic(authors: str) -> bool:
    non_academic_keywords = ["pharma", "biotech", "inc", "company", "corporation", "industry", "lab"]
    for keyword in non_academic_keywords:
        if re.search(rf" {keyword} ", authors, re.IGNORECASE):
            return True
    return False

def create_csv(papers: list, filename: str):
    df = pd.DataFrame(papers)
    df.to_csv(filename, index=False)
    print(f"Results saved to {filename}")

def main():
    parser = argparse.ArgumentParser(description="Fetch research papers from PubMed")
    parser.add_argument("query", help="Search query for PubMed")
    parser.add_argument("-f", "--file", help="Output CSV filename", default="papers.csv")
    parser.add_argument("-d", "--debug", help="Enable debug output", action="store_true")
    args = parser.parse_args()

    paper_ids = fetch_papers(args.query)
```

```
    if args.debug:
        print(f"Fetched {len(paper_ids)} papers from PubMed")

    papers = fetch_paper_details(paper_ids)
    filtered_papers = []
    for paper in papers:
        authors = paper.get("AuthorList", [])
        non_academic_authors = [author for author in authors if filter_non_academic(author)]
        if non_academic_authors:
            filtered_papers.append({
                "PubmedID": paper["Id"],
                "Title": paper["Title"],
                "Publication Date": paper["PubDate"],
                "Non-academic Author(s)": ", ".join(non_academic_authors),
                "Company Affiliation(s)": ", ".join(non_academic_authors),
                "Corresponding Author Email": paper.get("CorrespondingAuthor", "N/A"),
            })

    create_csv(filtered_papers, args.file)

if __name__ == "__main__":
    main()
```

Usage:

```bash
```

```
python research_paper_fetcher.py "machine learning in healthcare" -f output.csv -d
```

Explanation:

- **Command-line interface (CLI)**: Accepts a search query and allows optional arguments (`-f` for file output and `-d` for debugging).
- **Fetching Data**: Uses PubMed's API to search papers based on the query, and retrieves detailed data using the PubMed `esummary` API.
- **Non-academic Authors**: Checks if the authors are affiliated with pharmaceutical or biotech companies using keywords.
- **CSV Output**: Stores the filtered data in CSV format.

Output CSV sample:

```
PubmedID,Title,Publication Date,Non-academic Author(s),Company Affiliation(s),Corresponding Author Email
12345678,Investigating the Efficacy of Drug X for Cancer,2022-05-15,John Doe,BioPharma Co., jdoe@biopharmaco.com
87654321,Advances in Gene Therapy for Rare Diseases,2023-02-10,Richard Roe,GenTech Pharmaceuticals, rroe@genpharma.com
13579246,New Insights into Immunotherapy for Lung Cancer,2021-12-22,Emily White,PharmaGlobal,e.white@pharmaglobal.com
```

License:

MIT License