Steps to Automate Data Search

1. **Define Search Terms:**
   - A list of search terms is created and stored in a .txt file.
2. **Locate or Create Excel List:**
   - Shipping Lists https://www.fdlp.gov/collection-tools/shipping-lists
   - New Electronic Titles https://catalog.gpo.gov/F/?func=file&file_name=find-net&local_base=NEWTITLE
3. **Retrieve Matching Entries:**
   - Python searches through an Excel file, identifying rows that match the terms.
4. **Organize Results:**
   - Matching entries are extracted and placed into a new Excel file, streamlining the review process.

Ingredients Needed:

- **Criteria/Terms/Keywords:** Search terms stored in a .txt file.
- **Excel File of Records:** Spreadsheet containing the data to be searched.
- **Python Tool:** Python IDE, Pandas and Openpyxl libraries.

Tools & Technologies Used:

1. **PyCharm (or any Python IDE):**
   - Used for writing and running the Python code.
2. **Pandas Library:**
   - A Python module for data manipulation and analysis. Essential for reading and writing Excel files.
3. **Openpyxl Module:**
   - Required for Pandas to handle Excel file operations.

Setup Instructions:

1. **Run PyCharm in Administrator Mode:**
   - Right-click the PyCharm icon and select "Run as Administrator."
2. **Organize Files:**
   - Ensure all necessary files (code, Excel, search terms) are in the same folder. PyCharm typically generates a folder for the project.
3. **File Path Adjustments:**
   - Modify the file path to access the Excel file correctly by using raw strings (e.g., `r"file_path"`).
4. **Search Term Input:**
   - Input search terms using a .txt file. Ensure no extra spaces or lines are present.

Coding Iteration:

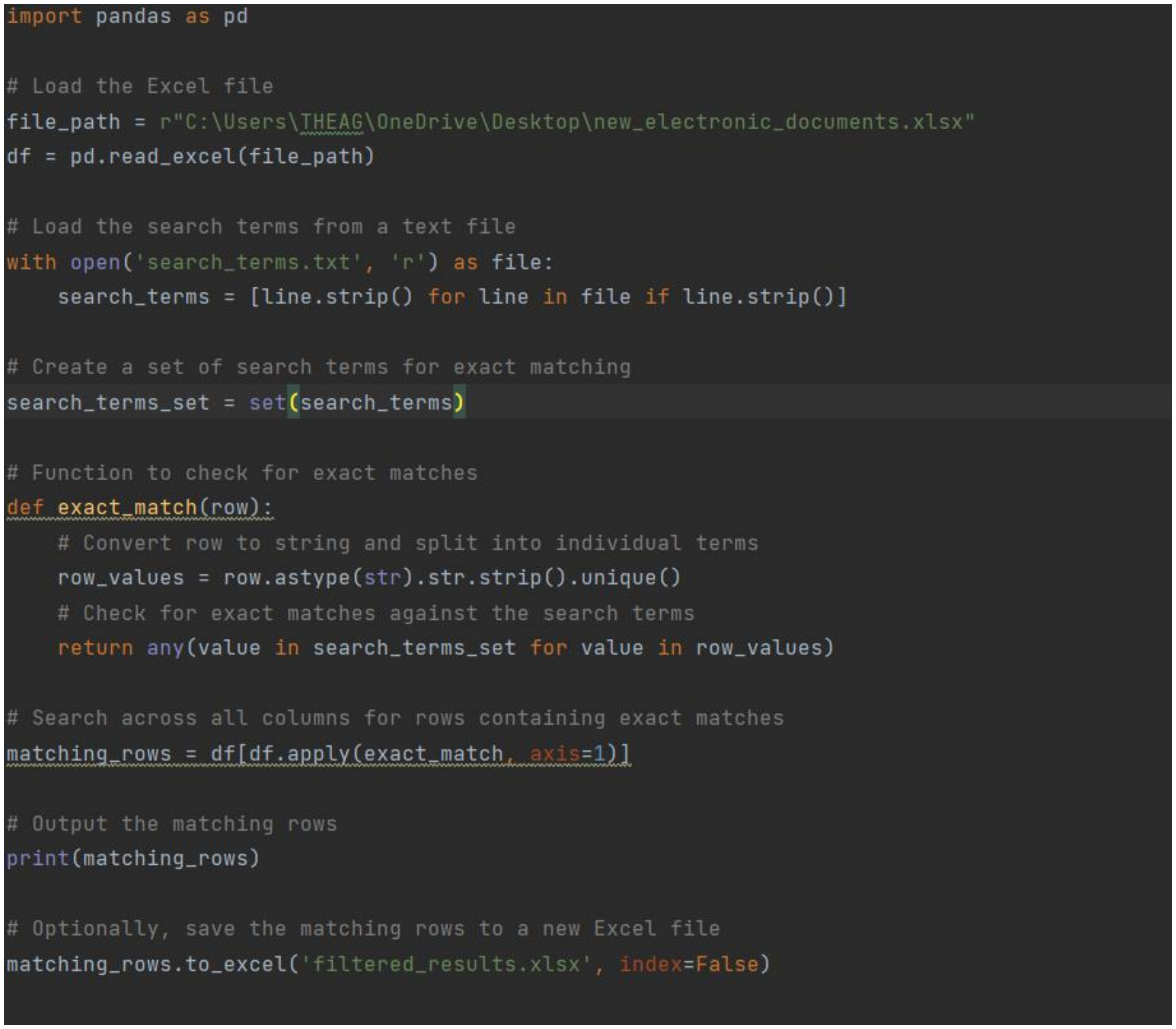
- **Efficiency Example:** Automating searches with a .txt file is faster than manual input.
- **Refinement:** Several iterations were required to refine a search based on unique depository selections.
- **Initial Issue:** Early attempts matched partial numbers (e.g., "0050-E-17" was matched because of "0050-E"). Code was modified to ensure exact matches only.

Pro Tip:

- For Search Criteria use Notepad, avoid spaces at the top of the .txt file. Search terms on the same line will be treated as a single term.

Code in Text format

```
import pandas as pd

# Load the Excel file
file_path = r"Insert your file path here"
df = pd.read_excel(file_path)

# Load the search terms from a text file
with open('search_terms.txt', 'r') as file:
    search_terms = [line.strip() for line in file if line.strip()]

# Create a set of search terms for exact matching
search_terms_set = set(search_terms)

# Function to check for exact matches
def exact_match(row):
    # Convert row to string and split into individual terms
    row_values = row.astype(str).str.strip().unique()
    # Check for exact matches against the search terms
    return any(value in search_terms_set for value in row_values)

# Search across all columns for rows containing exact matches
matching_rows = df[df.apply(exact_match, axis=1)]

# Output the matching rows
print(matching_rows)

# Optionally, save the matching rows to a new Excel file
matching_rows.to_excel('filtered_results.xlsx', index=False)
```



```python
import pandas as pd

# Load the Excel file
file_path = r"C:\Users\THEAG\OneDrive\Desktop\new_electronic_documents.xlsx"
df = pd.read_excel(file_path)

# Load the search terms from a text file
with open('search_terms.txt', 'r') as file:
    search_terms = [line.strip() for line in file if line.strip()]

# Create a set of search terms for exact matching
search_terms_set = set(search_terms)

# Function to check for exact matches
def exact_match(row):
    # Convert row to string and split into individual terms
    row_values = row.astype(str).str.strip().unique()
    # Check for exact matches against the search terms
    return any(value in search_terms_set for value in row_values)

# Search across all columns for rows containing exact matches
matching_rows = df[df.apply(exact_match, axis=1)]

# Output the matching rows
print(matching_rows)

# Optionally, save the matching rows to a new Excel file
matching_rows.to_excel('filtered_results.xlsx', index=False)
```