📂 Archive Data Extractor

This Python tool automates the process of extracting, cleaning, and organizing date-related information and metadata from structured Excel files. It is designed for use in archival, library, or historical inventory records where content like folder dates, location codes, and descriptions must be structured for digital use.

🚀 Features

- Extracts a wide variety of **date formats** using advanced regex
- Detects and preserves **bold subtitles and section titles**
- Merges continuation rows for descriptions
- Separates **scope indicators** like `[3 folders]` or `2 copies`
- Assigns and propagates **6-digit Location Numbers**
- Ensures **logical Box–Folder–Location** hierarchy
- Saves a fully cleaned and organized `.xlsx` output file

📁 Output File Columns

Column	Description
`Box`	Box number (auto-filled if missing)
`Folder`	Folder number, marker, or section label
`Description`	Cleaned content, with date removed
`Date_And_After`	Extracted date string
`Scope`	Quantity descriptors (e.g., "3 folders", "2 copies")
`LocationNumber`	6-digit archive location number assigned and filled down if applicable

---

🛠️ Usage Instructions

1. **Prepare your Excel file**
   - The default `data.xlsx` file must have three columns labeled exactly as `A`, `B`, and `C` in the first row.

These represent:
    - **A** → Box
    - **B** → Folder
    - **C** → Description
  - Paste your data **directly below** these headers.
    Do not rename or rearrange the columns — the script identifies them based on fixed order.
2. **Place your file in the working directory**
  - To locate your current working directory:
    - **Windows**:
      Right-click inside the folder where your script is → **Shift + Right Click** → "Copy as Path"
    - **macOS**:
      Open Terminal, type `cd` (with a space), then drag and drop the folder into the Terminal window.
  - Once you're in the correct folder, run:

```
cd /path/to/your/folder
python3 extract_dates.py
```

3. **Review your output**
  - The cleaned Excel file will be saved as:
    `processed_file.xlsx` in the same folder.

⚠ Important Notes

- ✅ **Paste using 'Match Destination Formatting'**
  Avoid Excel's auto-formatting that turns `04-25` into `Apr-25`. This preserves numeric or identifier formatting.
- ✅ **Remove any completely blank rows**
  These often appear when pasting from online sources and can interfere with the script.
- ✅ **Review special cases manually**
  For example: `Troost 4747, 1979–1980` may extract `4747` as a date. The script attempts to avoid this, but edge cases still need visual review.
- ✅ **Date Range Validation**
  - The script extracts dates **only between 1800 and 2099**.
  - To change this:
    - Open the `extract_dates.py` script.
    - Locate the `extract_text()` function (around line 240+).
    - Inside, find this condition:

```
if not (1800 <= year <= 2099):
```

- Update the lower/upper limits as needed.

🧪 Future Enhancements (Optional)

- Add unit tests for regex and logic validation
- Modularize into a CLI tool or installable Python package
- Develop a GUI for drag-and-drop Excel processing
- Integrate with cloud-based file storage for batch runs

👨‍💼 Author & Credits

Project Lead: Nithin
Tool Name: Archive Data Extractor
Tech Stack: Python, Pandas, OpenPyXL, Regex
