Getting info - extraction of desired material - better (faster, more accurate) than removal of each unwanted part to get to desired material

Note:

* “jq” is like “sed” for JSON data, with “sed” being the command that removes unwanted wording
* Git Bash doesn’t come with the “jq” function, so we need to download it to use it
* To test out if something worked without looking at the entire file, add “| head” at the end to only look at the first 10 lines

***For Non-Developers - step 2 - columns:***

* Open Git Bash and put in:

“cd ~

mkdir bin

cd bin

explorer .”

This allows us to create a special file called ‘bin’ that allows any .exe file to run (a.k.a. “reference”) if located in it (perfect for what we’ll be downloading next)

* Download “jq”: <https://stedolan.github.io/jq/>
* add the file to the “bin” file just created (it’ll be located in one’s computer account file, like C:\Users\username (username is the account name on one’s computer (this location is called the “home directory”)), and rename the file to “jq.exe” before or after downloading to use that name for the function.

In Git Bash:

* Optional step: type in “jq --version” to double check that everything works. A successful result would show the version that it’s at.
* ‘cd “absolute file path”’
	+ absolute file path - location where the dump file is in
* cat series.txt | cut -f5 | jq --raw-output '[.work\_title[], .authors[].key, .works[].key, .series[]] | @csv' > series.csv
	+ cat - a command to give the computer a function to work with, plus it shows the progress when running the code
		- disregard when it says “jq: error (at <stdin>:line #): Cannot iterate over null (null)”, as the lines that should appear still do in the finished file
		- check progress by single clicking on the file and viewing it on the preview pane in Windows Explorer. It also shows the file size at the bottom
	+ series.txt - use name of desired dump file on computer to use
	+ cut -f5 - is used because the info is only in the 5th row, so it takes less time (also, the code might not be able to work if they are not removed)
	+ Note:
		- .work\_title[], .authors[].key, .works[].key, .series[] - can be changed to any of the fields desired in the output listed in <https://openlibrary.org/type/edition>. If using a different dump file, go back to <https://openlibrary.org/type/> and choose a different one (like /type/work)
		- Be mindful of square brackets (a type of “array symbol”) that words may be inside, as they would need to be included and accounted for to get the info out from it. Curly brackets (<https://onlinewritingtraining.com.au/brackets-parentheses-square-angle-curly-brackets/>) are not a worry.
			* Examples: so if the words are covered with [], [] would need to be written next to the field. For instance, with the publisher query, add [] after, like: publisher[]. If what’s needed is inside the brackets, add the field next to the bracket. With authors, it would say authors[].key. Ones without brackets in the answer would not need that, like the title field.
		- editions.csv - could be assigned any name
		- To figure out how jq would be added into other functions, use <https://jqplay.org/>
			* Ex: “head dumpfile.txt | cut -f5 | clip”
				+ dumpfile.txt is the edited dump file from “step 1 - rows”
				+ cut - like cut/paste
				+ f5 is the column being cut out
				+ clip is copy version in copy/paste in Git Bash
			* So now column 5 is copied to be pasted, we’ll paste it online at <https://jqplay.org/> where it says “JSON”
			* Type into “filter”: [.title, .number\_of\_pages] | @csv
				+ .title, .number\_of\_pages - optional: choose your own fields
			* Copy what appears at the “command line” section into Git Bash, but add “| clip” at the end

uniq - next