# Task 2

Group members: Lance Fenicle, Joshua White

Overview:

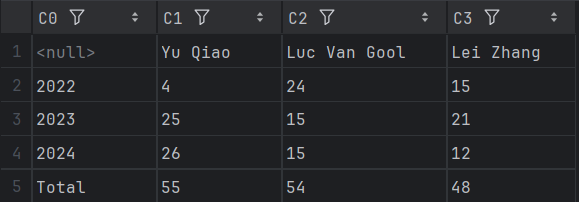
In this task, we went to 3 different web pages representing lists of scientific papers published in different years (2022, 2023, and 2024).

On this page, there were paper titles, and underneath, the names of the authors who contributed to the papers. For this task, we webscraped the 3 web pages, found every author and how many papers they had contributed to for each year, found out how much they contributed total, and made an Excel spreadsheet containing the 3 top contributors, how much they contributed among each of the 3 years, and how much they contributed total.

Table of functions

| **Function Prototype** | **Description** |
| --- | --- |
| fill\_contrib\_dict(papers\_list, dictionary) | This function was made specifically for this program. It takes the list of authors’ names that I had webscraped for each year, then removes newlines and commas, as that would sometimes bleed into the text data on the site. I caught the newlines initially, but it wasn’t until later in development that I had caught the commas. Once it gets that “true name”, it checks to see if that author is already in the dictionary. If not, it will add them, and sets the value (number of papers they contributed to) to 1. If they are, it increments the value |
| remove\_new\_lines(string) | A simple function that just removes any newline characters from a string. |
| remove\_commas(string) | A simple function that just removes any comma characters from a string |
| merge\_dicts(total\_dict, dict1, dict2) | merge\_dicts() takes 2 dictionaries, and for every instance where they share a key, the values of that key are combined, otherwise, keys unique to one of the dictionaries and its value are added in. It returns 1 dictionary of the two passed dictionaries combined. This is used because I have 3 dictionaries of authors for contributions of each year, and needed to be able to find total contributions among all 3 years. |
| get\_largest\_value\_dict(dictionary, largest) | get\_largest\_value() returns the greatest value among all keys in a dictionary. Because I needed this functionality, I also have the largest parameter. This function will return the greatest value less than largest, this allows me to get second and third largests. |
| def try\_for\_keyerror(dictionary, key) | As I was filling the spreadsheet, I came across an issue where sometimes it would look at a value in a key that wasn’t in the dictionary, so I made this function which is essentially just a try/except that I can call in a single line. |
| fill\_sheet\_top\_contrib(sheet) | This function is the one that fills the spreadsheet as specified. The logic is fairly complex, but in simple terms, it runs through each row and each column of that row. And based on which column and row it’s looking at, will put the needed information. For example, A:1 needed to be empty, so there’s a check for that. If the column was A, but the row wasn’t 1, the year needed to be added unless it was A:5, then ‘total’ needed to be printed. And it does that kind of logic for every group of row/column |

Screenshots:



Steps to run the task:

1. Run the Python file
2. Check the file made (note, the program needs the top\_contributors.xlsx file in the same directory as the task\_2.py file before it’s run. In the zip, I have provided it, so there should be no issues)

Conclusion:

I found the task a simple, yet fun exercise. The main difficulty came from the fill\_sheet\_top\_contrib(), not for any technical reasons, but logically trying to make the code as simple and efficient as possible.

Other than that, development went smoothly. I got a response from the 3 sites, made soups for each, investigated the html to find that author names are given the class “authsearch” and used that to get every one. I then, created dictionaries for each year, merged them, got the largest amounts and the names of the authors who contributed to them, and filled the spreadsheet.