1. Use GitHub API to collect information about GitHub repositories
    - Name
    - Owner
    - Description
    - Homepage
    - License
    - Number of forks
    - Watchers
    - Date the data was collected

When you request to print the object (my_repo) it should look something like this:
‘<owner>/<repository_name>: <description> (<watchers>)'

2. Relate each repository to a list of pull requests. For each pull request you need to keep:
    - Title
    - Number
    - Body
    - State
    - Date of creation (created_at)
    - Closing date (if the state is different than open)
    - User

Thus, for each repo, you need to collect the pull requests that are returned on the first page of a query like this (using repository https://github.com/JabRef/jabref as an example):
https://api.github.com/search/issues?q=is:pr+repo:jabref/jabrefLinks to an external site.

For the last 4 fields: number of commits, additions, deletions, and changed_files; You will need to make another query using the following format using the number of the pull requests you found before (using repository https://github.com/JabRef/jabref as an example):
https://api.github.com/repos/JabRef/jabref/pulls/5531

3. You are also required to *scrape* the following information from the user profile page on GitHub:
    - Number of Repositories
    - Number of Followers
    - Number of Following
    - Number of contributions in the last year

4. You must develop a function called `save_as_csv` that can be reused to convert any object to a csv entry (row).

The function receives the file name and the object to be converted. If the file does not exist, you need to create the file (with a header). If the file exists, you need to append a new line with the object in the CSV. To make it possible, you will need to have a method in each of your classes with the very same name, which will return a string with the data already structured as a CSV. Use this function to create/update the files as following (NO REPEATED ENTRIES): o when you collect data from a repositories, you need to add it to a CSV called `repositories.csv` o when you collect the pull requests of a repositories, you need

to store them in a file named after the owner and the name of repository(repos/owner-repo.csv) o when you collect data from users, you need to add it to a CSV called `users.csv`

5. You should have a menu for your app (Console is sufficient, GUI is allowed). The menu should allow its' user to:
    - Request the system to collect data for a specific repository (from GitHub). By providing the owner and repository name, your program needs to start the collection of data, such as:
        - Repository
        - Pull request
        - Users -- including scraped data
    - Show all repositories collected (with submenu of actions possible on each repo) ▪
        - Show all pull requests from a certain repository
        - Show the summary of a repository. Summary must contain:
            A. Number of pull requests in `open` state
            B. Number of pull requests in `closed` state
            C. Number of users
            D. Date of the oldest pull request
        - Create and store visual representation data about the repository (via pandas)
            A. A boxplot that compares closed vs. open pull requests in terms of number of commits
            B. A boxplot that compares closed vs. open pull requests in terms of additions and deletions
            C. A boxplot that compares the number of changed files grouped by the author association
            D. A scatterplot that shows the relationship between additions and deletions
        - Calculate the correlation between all the numeric data in the pull requests for a repository
    - Create and store visual representation data about all the repositories (via pandas):
        - A line graph showing the total number of pull requests per day
        - A line graph comparing number of open and closed pull requests per day
        - A bars plot comparing the number of users per repository
    - Calculate the correlation between the data collected for the users
        - Following
        - Followers
        - Number of pull requests
        - Number of contributions
        - Etc.

6. You must have 5 unit tests for your project.