Assignment 1: GitHub Classroom

Objectives

Start getting familiar with `git`, GitHub, GitHub Classroom, and Python.

Tasks

1. Accept the GitHub Classroom assignment at
2. Clone the repository
3. Make a branch with a meaningful/useful name
4. In `math_lib.py`, fix the issue with the `division` function (hint: what happens when the second input/argument is zero?)
5. Add a "Changelog" section to `README.md`. In the changelog, add a bulleted list that summarizes your changes
6. Commit your changes to the branch with a useful commit messages
7. Push your local branch to your remote repository. Create a pull request and merge the pull request.
8. In a new branch (don't forget to move your local repository back to main and pull the new changes)
    a. add a function called `add` that adds to numbers to `math_lib.py`
    b. create new file `calculate.py` that uses both methods (see the Hints section below)
    c. create a new file `run.sh` that runs `calculate.py`
9. Like in steps 5-7, update the changelog with your changes (don't forget to describe what `run.sh` does and how to run it!), commit your changes, push your changes, and create and merge a pull request.
10. OPTIONAL: in a new branch, do the extra credit described below. Update the changelog, commit your changes, push your changes, and create and merge a pull request.
11. Create a release from `main` tagged as v1.0

(Fairly) Straightforward extra credit + command line practice

The repo contains a FASTA file `example.fasta`. Add code to `run.sh` so that, in addition to the running `calculate.py`, the script

1. accepts a FASTA file (you could also add functionality that makes the FASTA file optional, i.e., if a FASTA file is provided, process it; if not, ignore/skip the FASTA file logic)
2. extracts and prints all the instances of "><string>". For example, on `example.fasta`, the script would print:

```
>crab_anapl
>crab_bovin
>crab_chick
>crab_human
>crab_mesau
>crab_mouse
>crab_rabit
>crab_rat
>crab_squac
```

Hints

In order to pull your file `math_lib.py` into the `calculate.py` file, you can use "`import`" followed by the file name (no extension) at the top of your python file. You can choose any name as your import as, here I have used the abbreviation "`ml`".

```
import math_lib as ml
```

Then in order to use your functions, you will just call them from the math_lib module using your extension, for example:

```
x3 = ml.add(3,2)
x4 = ml.div(34,2)
```