DS 210 Discussion 2

1 Git Overview

Technically: "*Git* is a **distributed version control** software system that is capable of managing versions of source code or data. It is often used to control source code by programmers who are developing software collaboratively. - Wikipedia"

In short, Git is a fast and arranged way to control the changes of your codes, so you do not need to do "v1, v2, ..." or "(1) (2) (3)". Basically, you create **branches** (local and remote) for your codes, **add** changes made to your local branches, **commit** them, **pull** from and **push** to remote. Sometimes you will also need to **merge** between branches.

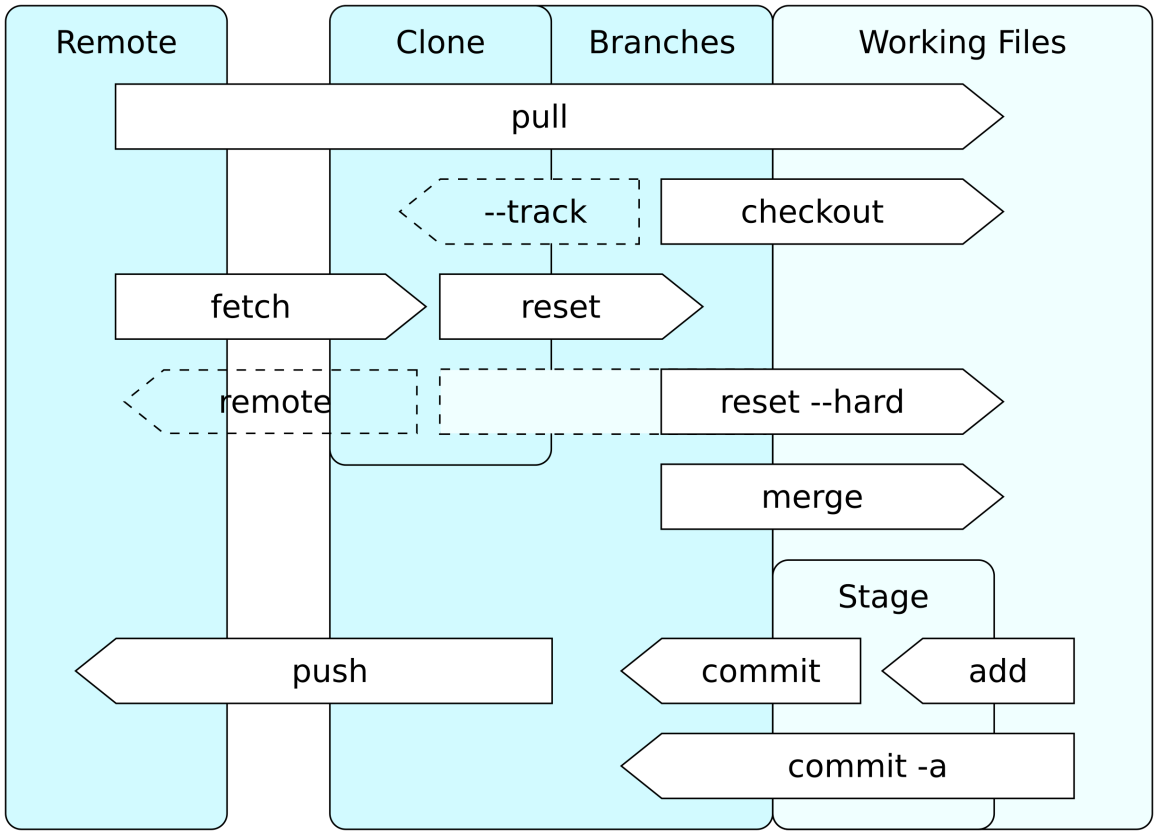
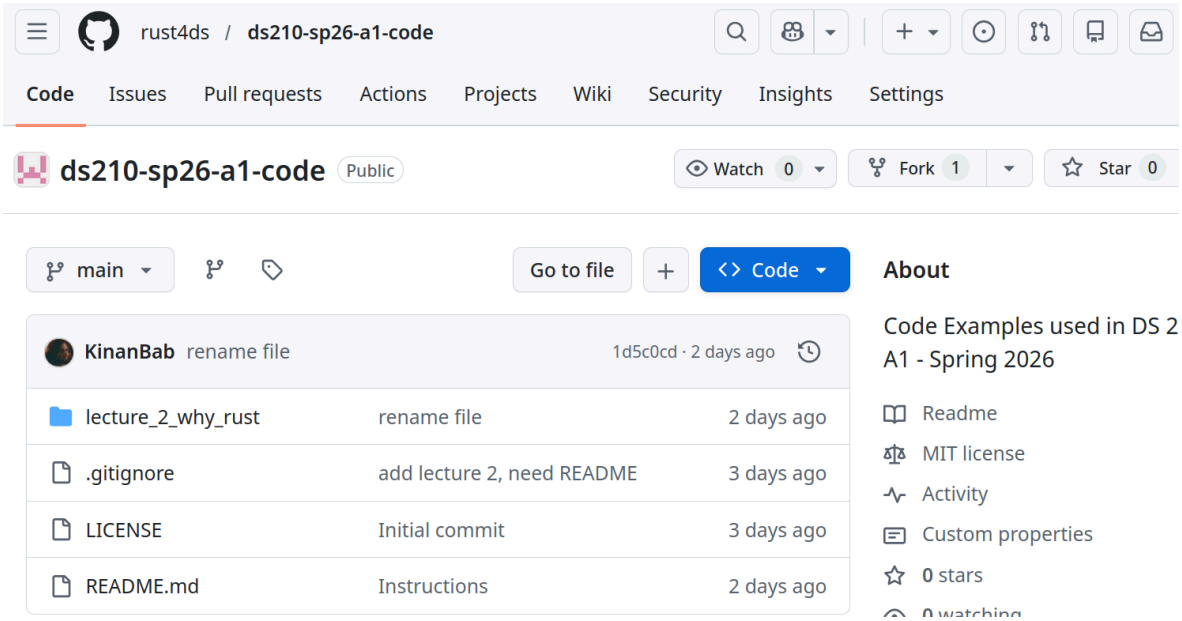


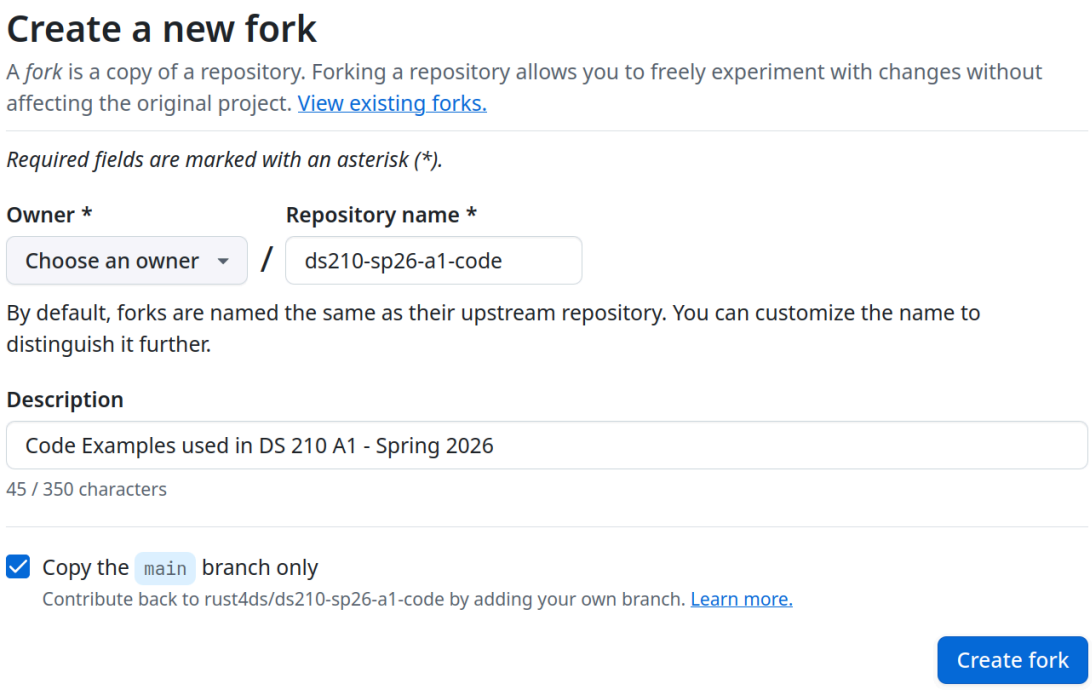
Figure 1: A chart of git operations. (from Wikipedia)

2 Fork the sample repository

1. Open our code repository https://github.com/rust4ds/ds210-sp26-a1-code in your browser:



2. Click the **fork** button on top right to fork your own copy of this repo to your account.



3 Clone the forked repo

Open your terminal, inside of your course folder (or anywhere you like), clone the repo you just cloned to your local computer.

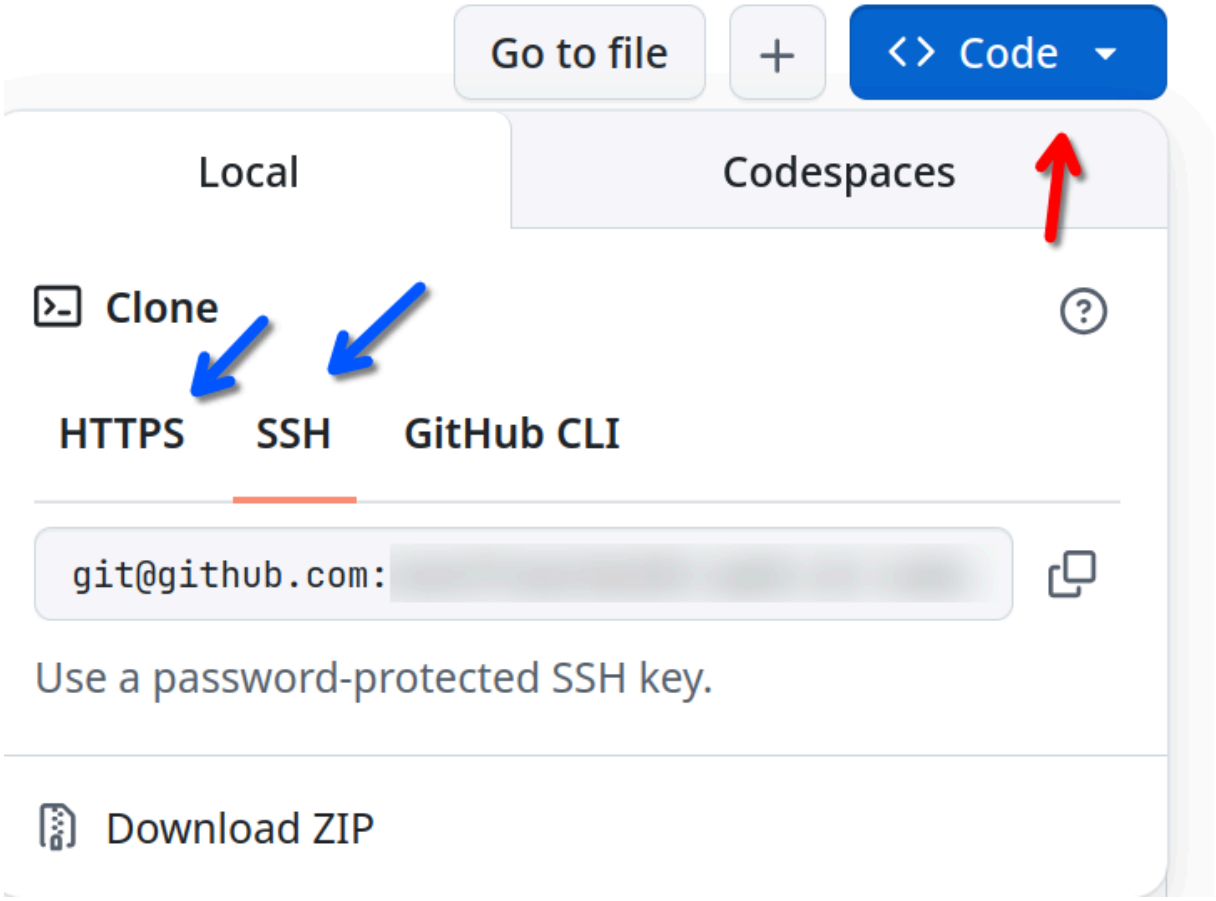
1. By `https`:

```
git clone https://github.com/[your_username]/ds210-sp26-a1-code.git
```

2. Or by `ssh`:

```
git clone git@github.com:[your_username]/ds210-sp26-a1-code
```

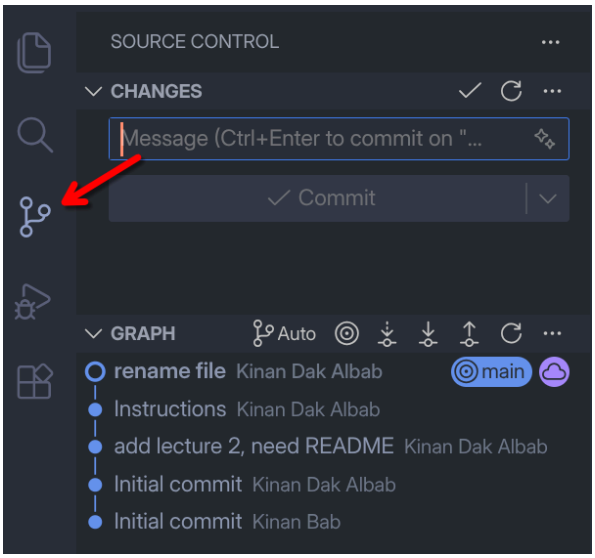
You can also find links to copy in the repo page **of your own forked repo**. (Don't clone the original repo from `rust4ds`!)



1. You may need to configure your creditional on your computer. For now, you could use `https` and type your username and password there.
2. Though, we recommend you to create an `ssh` key, config your name and email in Git, and use the second approach which is more secure and set-and-forget.
   - But remember, **never share or leak your ssh key in public!!** That gives anyone who has it the access to your GitHub account and precious projects.)
   - A good tutorial of how to do that: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

4 Add remote repo (Work in groups)

1. Now open the cloned repo folder in your VSCode. In the left bar you should see a git tab, and click on it will show you the a visualization of branches, commits, merges, and so on of this repo:



2. Work with your classmates in group of two, add the other's forked repo to the remote:

   ```
   git remote add other git@github.com:[other_username]/ds210-sp26-a1-code
   ```

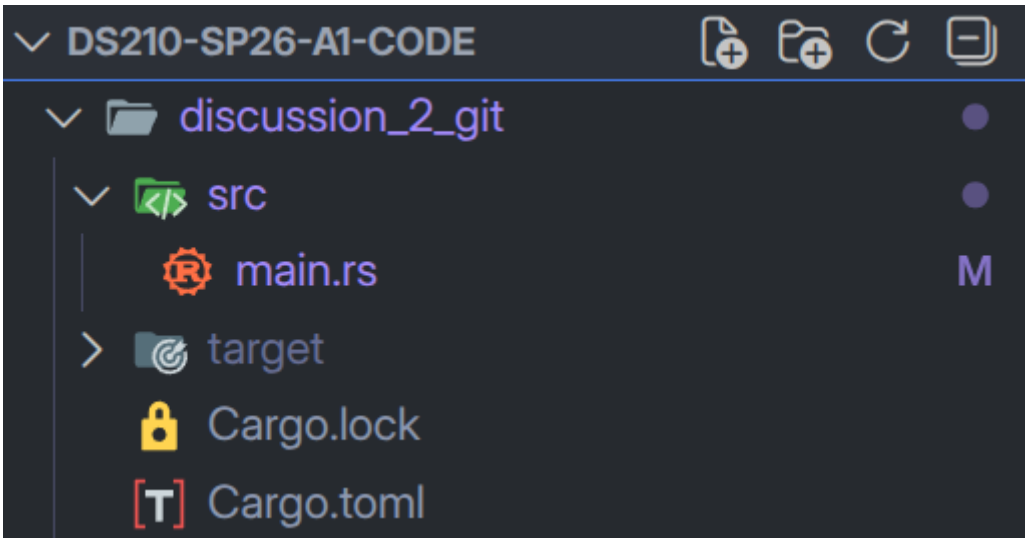
   or

   ```
   git remote add other https://github.com/[other_username]/ds210-sp26-a1-code.git
   ```

5 Complete the short code

1. Locate the file `discussion_2_git/src/main.rs`. In your terminal, try to use cargo to run the codes to test your `rust` runs well.

   ```
   cd discussion_2_git
   cargo run
   ```



2. For the two functions, each one select **one** function and complete it in your VSCode.
   - **One of you** does

```
fn parity(x: i64) → i64 {
    return x % 2; // Change this line
}
```

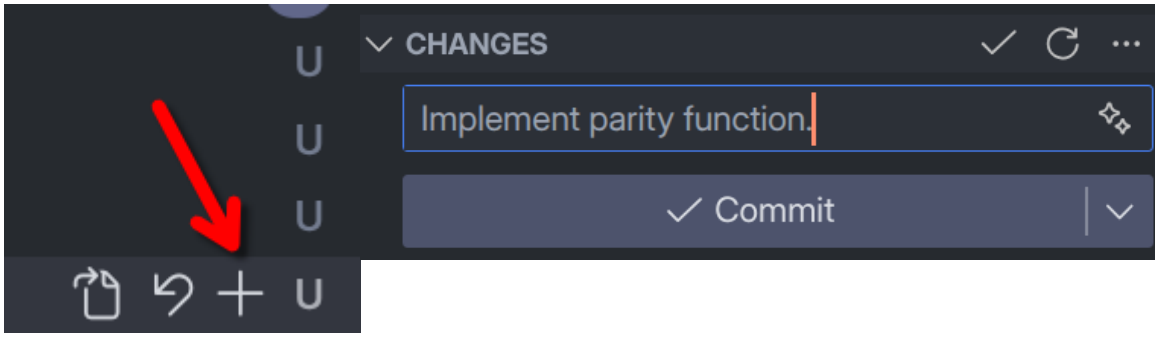
- **The other** does

```
fn add(a: i64, b: i64) → i64{
    return a + b; // Change this line
}
```

3. **Commit** the changes you make with a reasonable message (like "Implement parity function.").
   - You can do that in your terminal (recommended):

     ```
     # Stage all changes you make
     git add .
     # Commit the staged changes
     git commit -m "[message you want to write]"
     ```

   - Or in VSCode GUI:



6 Push to the remote

In this part, we will use command line to have a better understanding of what's happening here. These can be also done in GUIs but we want you to understand the process.

1. First, let's try to push into your own remote repo `origin/main`:

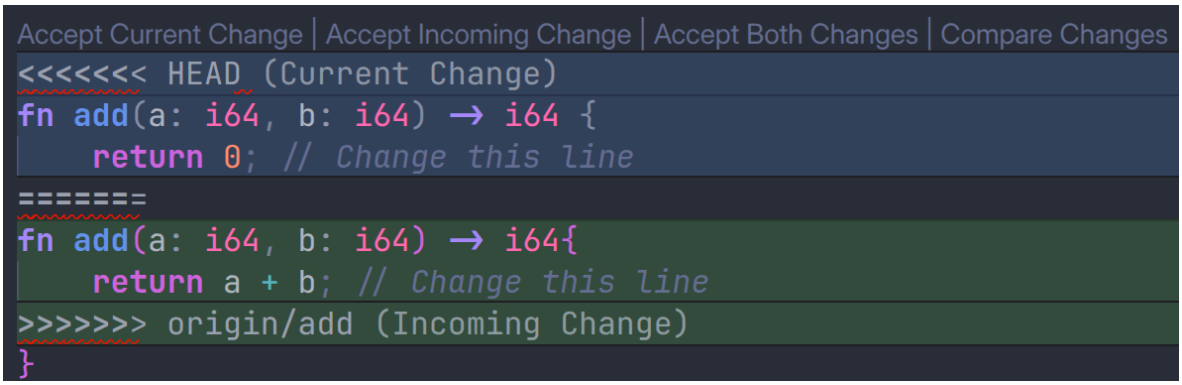
   ```
   git push # This will push your local 'main' branch to the remote 'origin'
   ```

   - Since you cloned from your own repo, your *local* `main` branch is tracked with your *remote* `origin/main`. Thus, the `git push` will automatically push your commits to the tracking `origin/main`.

2. **After both of you make the push**, try pull the other's changes into your local branch:

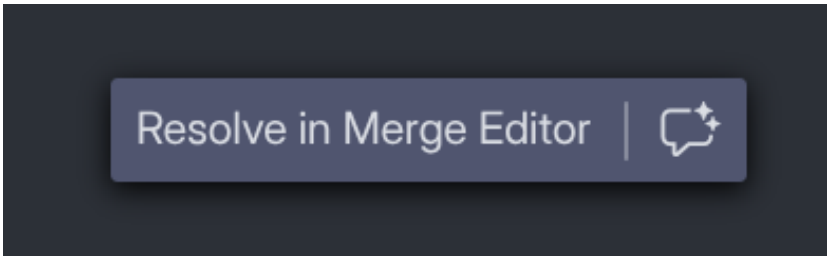
   ```
   git fetch --all # Fetch all changes from remotes
   git merge other/main # Merge the changes from the other's repo
   ```

3. There is a conflict happening since your codes have different implementation of functions. No worries! Now we will solve and merge them.



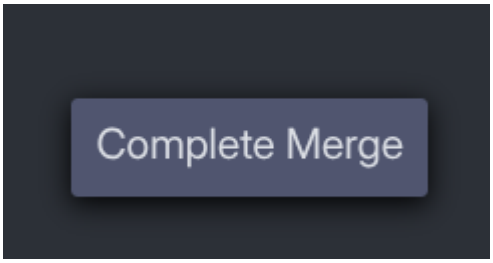
7 Resolve Conflicts and Merge

1. Click the toast shown in the bottom right:

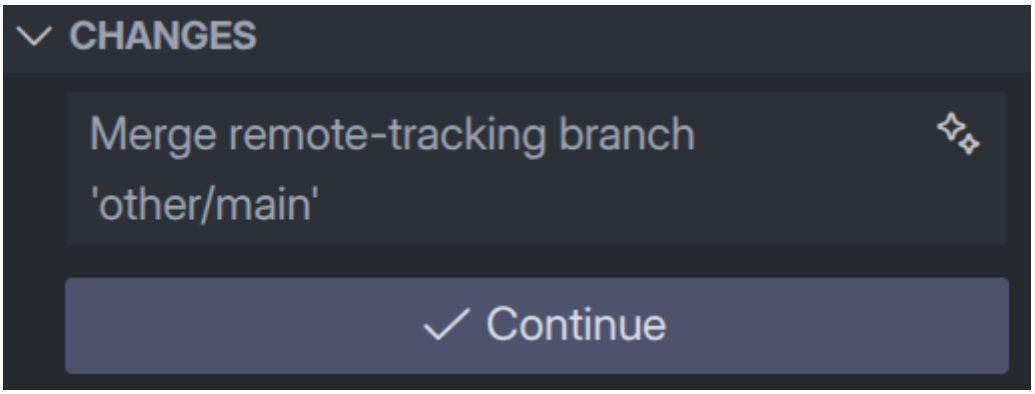


2. Now you should see something like

3. Select “Accept Incoming” from changes of `other/main`, which is what we want to get.
   - Or, you can modify the codes in the bottom window manually. This allows more flexible edits and merges.
4. Click the toast:



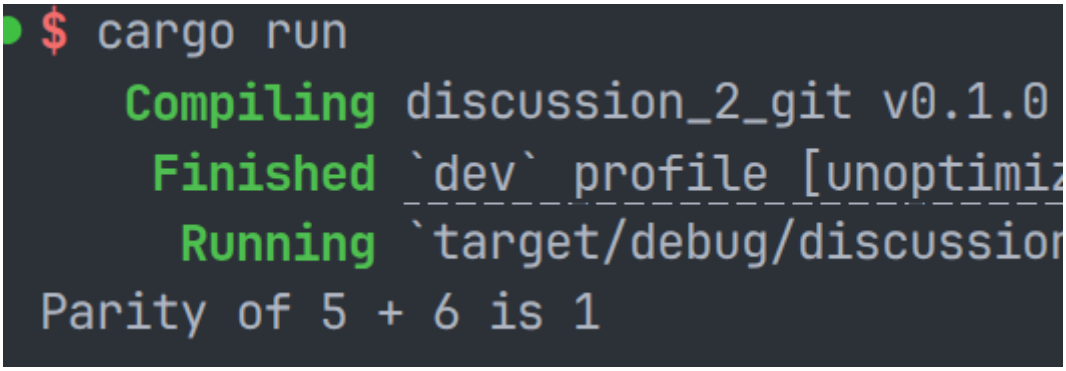
5. Remember to make the commit!



6. Now you finshed merging the codes from the other's repo! Let's try running:

   ```
   cargo run
   ```

   It should give output:



7. Lastly, push your merged local branch to your remote:

   ```
   git push
   ```

8 One more thing...

Now you know how to make changes to your repo locally, commit them, push them to remote (which is GitHub here), and merge other's changes into your own codebase.

But how to actively *contribute your codes to original/other forked repo*? You will need to make a **pull request (PR)**. This is not a git feature but GitHub, and you may want to use this approach for collaboration works in course projects (fork a repo of your own, did your part in your group, make PR to main repo, discuss and merge the PR).

Due to time constriants we cannot do a demo here, but we strongly recommend you to see Github's tutorial of "Pull Requests" https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests. You should already be equipped with enough knowledge to understand how it works.

- Please do not make random PR to `rust4ds` (or any other community projects on GitHub)! You can try to create your own repo and play with it as you want.