Problemset: Git and GitHub (II)

Collaborative Workflows

In this excercise, we practice collaboration via a **shared central repository** using the **feature branch** workflow. Do this exercise in a team of two (or three) persons. We call them person A and person B. To get the most out of this exercise, share your screen with each other!

1 Setup

First, you need to create a shared central repository on Github:

- One person creates a new repository on Github, and invites the other person as collaborator. (In Github navigate to Settings > Manage Access > Invite a collaborator)
- The other person accepts the invitation (received by email and Github notificiation)
- Both collaborators clone the central repo to their local machines.

2 Working with Branches and Pull Requests

- In this exercise, A works on a new feature in a dedicated branch, pushes this branch to Github, and opens a pull request.
- B reviews the pull request: Do this for the sake of practicing as a thorough discussion. E.g. person B may requests some code adjustments. Also B may mark lines in the code and ask for explanations from A. A addresses the comments through further commits.
- Finally, B merges the changes into the main branch on Github. A and B pull the current status of the project to their local systems. The feature branch may be deleted locally and on Github.

Code hints:

```
# Make sure that main is up to date
git checkout main
git pull

# Create and checkout feature branch
git branch feature
git checkout feature

# Edit, add, commit and push to Github
git add .
git commit -m "your commit message"
git push --set-upstream origin feature

# Update local main branch
git checkout main
```

```
git pull

# Delete local feature branch
git branch -d feature
```

3 Merge conflict

In this exercise you are provoking a merge conflict. A and B work on separate features, editing the same file. The changes from A, who finishes first, are merged via a Pull Request into the main code base. What happens if B wants to merge her changes into main as well? How can B deal with this situation?

Code hints

```
# Pull latest changes from Github into main
git checkout main
git pull

# Option 1: Rebase commits in feature branch on latest commits in main
git checkout featureB
git rebase main
(git rebase --continue)

# Option 2: Merge latest commits in main into feature branch
git checkout featureB
git merge main
```