Week 11

Introduction to Programming and Numerical Analysis

Ebener, Luca

UNIVERSITY OF COPENHAGEN

Overview

- Git introduction
- Git demonstration
- Work on inaugural project

What is Git?

- Git is a version control system used for tracking changes in source code.
- It allows multiple developers to collaborate on projects efficiently.
- Git tracks changes, allows branching, merging, and reverting changes easily.

Key Terms in Git

- **Repository**: A folder or directory where your project resides, managed by Git.
- **Commit**: A snapshot of the current version of the repository.
- **Three states**:
  - **Modified**: You have changed the file but have not committed it yet.
  - **Staged**: You have marked a modified file to go into your next commit.
  - **Committed**: The data is safely stored in your local database.
- **Branch**: A parallel version of the repository, allowing for experimentation and isolation of features.
- **Merge**: Combining changes from different branches into one.

Basic Git Commands

- **git init**: Initialize a new Git repository.
- **git clone <repository_URL>**: Clone an existing repository from a URL.
- **git add <filename>**: Add changes to the staging area.
- **git commit -m ”message”**: Commit staged changes with a descriptive message.
- **git push**: Upload local repository changes to a remote repository.
- **git pull**: Fetch and merge changes from a remote repository to your local repository.

What is GitHub?

- GitHub is a web-based hosting service for version control using Git.
- It provides a platform for collaboration, code review, and project management.
- GitHub allows users to host public or private repositories.
- It offers features like issue tracking, wikis, and project boards.

Git Branching Strategies

- **Feature Branching**: Each new feature or task is developed in a dedicated branch, keeping the main branch clean.
- **Git Flow**: A branching model that defines specific branches for different stages of development (e.g., feature, release, hotfix).
- **Trunk-Based Development**: All developers work on a single branch (usually main or master), promoting frequent integration and smaller, iterative changes.

Rebasing vs. Merging

- **Merge**: Combines changes from one branch into another, creating a new commit that has two parent commits.
- **Rebase**: Moves the entire branch to begin on the tip of another branch, resulting in a linear history without merge commits.
- **When to use**: Use merging for collaborative branches, use rebasing for keeping a clean and linear history, especially for feature branches.

Demonstration

- Lets try out these concepts.
- Resources:
  - Cheatsheet:
    - https://education.github.com/git-cheat-sheet-education.pdf
  - VSCode Git-introduction:
    - https://www.youtube.com/watch?v=i_23KUAEtUM
    - https://www.youtube.com/watch?v=HosPml1qkrg