Bioinformatics institute 2024/2025
Python programming, Nikita Vaulin

Git practice

This text is to read
This text is to copy
This text if to edit (if necessary)

Additional sources

- git book
- Learning git branching

Glossary

Repository (repo)	A directory which is tracked by git (including all sub-directories)
Commit	A snapshot of the repository
Branch	An ordered series of commits
HEAD	A commit that you are currently on

Git practice commands

1. New repo

Create a new empty directory

```
mkdir test_project && cd test_project
```

Check the directory is empty

```
ls -al
```

Initialize (start) git repository in this folder. Now git will track the changes in this directory

```
git init
```

See the message "**Initialized empty Git repository in …**"

Check the directory content once again. What's new?

```
ls -al
```

2. Working with file

Let's create new non-empty file

```
touch file
```

Add file to the index (tell git to track the file)

```
git add file
```

Coomit file - save it's current version in git

```
git commit
```

1) Now git will ask you to write the commit message (description). Just write "*Initial commit*' and exit editor (`Ctrl+X, Y, Enter` for **nano**; `Esc, :wq, Enter` for **vim**)
2) For newly installed git you may need to add some configurations:

```
git config --global user.email your@email.com
git config --global user.name "Your Name"
```

Edit the file:

```
echo "Hello world!" >> file
git add file
git commit -m "Edit file"
```

3. Git branching

Move to the very first commit

```
git log
git checkout <hash of the first commit>
```

See the message "`You are in 'detached HEAD' state`"

Create new branch and switch to it

```
git checkout -b <new branch name>
```

Do some work

```
echo "New file" > file2
echo "And another new file" > file3
git add file2 file3
git commit -m "Add files 2 and 3"

echo "Edit file3" > file3
git add file3
git commit -m "Edit file3"
```

4. Merge branches

You always merge some branch** **<u>into</u>** **the branch you are currently on. So to merge your new branch *into* main/master branch, first - switch to the main/master branch.

```
git checkout main
```

Do merge

```
git merge <new branch name>
```

5. See the branches

stackoverflow.com/questions/1057564/pretty-git-branch-graphs

```
git log --graph --oneline --decorate --all
```

Beautiful, isn't it?

Git and GitHub

1. Make your PC and GitHub trust each other

```
ls -al ~/.ssh
ssh-keygen -f ~/.ssh/github_key -t ed25519 -C "your@email.com"
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/github_key
cat ~/.ssh/github_key.pub
```

Copy all the content of *.pub* file and paste into the github.com/settings/keys (→ `New ssh key`).
Add the meaningful *Title* (like “WSL2 key” or “Ubuntu key”, etc)

2. Two ways to link your local repo with remote GitHub repo

1) Easy. Open remote repo on GitHub, (→ `Code` → `SSH` → `copy`)
   Download (fetch) this repo

   ```
   git clone <SSH-URL>
   ```

   Now you can work with it (don’t forget to enter the directory)

2) Harder. Do some work in local repo. Then create an empty repo on GitHub, copy its ssh-url. Then in terminal:

   ```
   git remote add origin <SSH-URL>
   ```

3. Exchanging data between local and remote repos

Download data from remote to local

```
git pull
```

Upload data from local to remote

```
git push
```