Git branches and merges

- When we need our code to diverge into two different versions, we start a new branch of the current repo (git branch *newbranchname*)
- it starts as a duplicate, but changes are now made seperately (e.g. a windows branch and a linux branch)
- Can have multiple branches, which can in turn branch out again
- We switch to working in a new branch using checkout (git checkout *branchname*)

Basic branch commands

- git branch *name* (creates new branch)
- git branch -D *name* (deletes branch)
- git checkout *name* (switch to new branch, make sure you’ve added/committed or stashed all changes first)
- git branch -m *oldname newname* (rename branch)
- git branch (shows list of all branch names)

Merging branches

- We can also take two existing branches and merge them together into a single branch
- Somewhere they have a common ancestor, where they diverged into different branches
- Files that have changes since then in one branch but not the other use the changed version
- Files that haven’t changed in either branch (of course) stay the same
- Files that have changed in both branches cause a conflict...

Merge conflicts

- When we have two conflicting versions of a file to merge together, git warns you of a merge conflict and doesn’t complete the merge until you resolve the conflict
- If you open the conflicted file, you’ll find the sections with conflicting changes shown like

```
Stuff that’s the same
<<<<<<
version from one file
=======
Version from other file
>>>>>>
```

Resolving conflicts

- Edit the conflicted file to keep the parts you want and delete the parts you don’t want, make sure you delete the lines of ==== >>>> <<<<
- Save and do a git add for the file
- The merge will complete when the last conflict is resolved and you do a git commit
- Now you have a single branch

Visualizing the branches

- You can see the branches visually by running programs like *gitk* when you are in the repository, but those won’t run in a simple text ssh window
- You can get an ascii map of the branches, merges, and commits using the following command

```
git log --graph --branches --pretty
```