GIT Hands On

What is a Version Control System?

- It keeps track of changes made to a project by the members of the same team.
- At any point in time any version of the code can be recovered.

What is GIT?

- It is a distributed version cotrol system (as opposed to centralized ones).
- The main advantage is that there is no need to be in constant contact with a central repository.
  - A **repository** is the collection of files and folders associated with a project along with each file's revision history.

Install GIT

- Git - Downloads

Basic Commands

See Git Handbook · GitHub Guides for more details.

- `git init` Initiate a new repository and start tracking the current folder.
- `git clone` Creates a local copy of an already existing remote repository.
- `git add` Stages changes to be committed.
- `git commit` Saves the staged changes to the project history.
  - Tracking file history is a two step process in GIT.
  - Staging takes a snapshot to be included in the project history.
- `git status` Shows the status of changes as untracked, modified or staged.
- `git branch` Shows the branches being worked on.
- `git merge` Used to combine together changes made in different branches.
- `git pull` Updates the local copy of the project with the changes present on the remote one.
- `git push` Updatest the remote copy of the project with the changes present in the local one.

Merge Conflicts

Practical example of the workflow

- Open the GIT Bash shell

```
$ git clone <repo address>
$ cd <repo-dir>
$ mkdir <name>-test
$ echo "Some content to play with" > merge.txt
$ git add merge.txt
$ git commit -am "Initial commit"
# The first time it will ask you to setup some iformation
```

We have a repository with the **master** branch containing one file which is committed. Let us now create a new branch.

```
$ git checkout -b merge_branch
$ echo "New content to be merged with the master branch" > merge.txt
$ git commit -am "Edited content of merge.txt to create conflict."
```

We now have a new branch **new_merge_branch** with a commit that overrides the content of merge.txt. Let us now go back to work on the master branch.

```
$ git checkout master
$ echo "New content to append" >> merge.txt
$ git commit -am "Added content to merge.txt"
```

At this point we have a repository with two commits, one in the master branch and the other in the merge_branch. Let us try to merge these two and see what happens.

```
$ git merge merge_branch
```

And we see we have a conflict to resolve. Let us see how to proceed from here.

```
$ git status
$ cat merge.txt
```