Installing git

- On Windows, go to https://git-scm.com/download/win for the download to start automatically.

- On MacOS/Linux, first open a new Terminal Window.
    - On MacOS, try

      $ git --version

      This will check for the current version of git and prompt an install if it is not already installed.

    - On Linux, try

      $ sudo dnf install git-all
      or
      $ sudo apt install git-all

Creating a git repository

1. Create a new folder and name it NewRepo.

2. Navigate to this folder on command line by typing in

   $ cd NewRepo

3. To make this folder into a git repository, type in

   $ git init

   This creates a file called ".git" inside your folder. Deleting this file will cause the folder to lose its git repository status.

4. To create a readme file, type in

   $ touch readme.md

Making Changes to Files

1. To check the current status of your git repository, type in

   $ git status

   This shows the following message:

```
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        readme.md
```

   Files under ‘untracked’ are those that are not added to the git repository yet.

2. To add this new file, type in

   $ git add readme.md

   $ git status

   Now we get the following message:

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   readme.md
```

3. “Adding” the file means that git can now track changes to the file, but these changes have not been finalized or “committed”. To do this, type in

   $ git commit -m “added readme file”

   The -m flag refers to a brief message describing the commit. Once more, we try

   $ git status

```
On branch master
nothing to commit, working tree clean
```

Branches

Creating a branch makes a copy of the git repository in its current form. This allows you to keep working on the repository while also storing a copy of its current form. Branches can be merged into each other in the future. This is especially useful if a project has multiple collaborators.

1. We will first add a new file to our git repository.

   $ touch hello_world.txt
   Open this file in a text editor and type in “Hello world!” in the first line. Now commit this new file and its contents.

   $ git add hello_world.txt
   $ git commit -m “added hello world file”

2. Now we will create a new branch.
   $ git branch new_edits
   This creates a branch named “new_edits”

   The main branch is usually called “master” or “main”.

   $ git checkout new_edits

   This changes the current working branch from “master” to “new_edits”

3. In this new branch, we will make changes to the hello_world file. Open this file in a text editor and add another sentence: “It’s a beautiful day!”

   $ git add hello_world.txt
   $ git commit -m “modfied hello world file”

   This commits the new changes to the new_edits branch.

4. Now if we switch back to branch master and check status,
   $ git checkout master
   $ git status

   ```
   On branch master
   nothing to commit, working tree clean
   ```

From the pov of the master branch, the changes to the hello_world file are not yet part of the git repository. If you open the file in a text editor now, you can see that only the sentence “Hello world!” is visible.

We can also see this with the git diff command:
$ git diff master new_edits

```
diff --git a/hello_world.txt b/hello_world.txt
index c57eff5..fab8e71 100644
--- a/hello_world.txt
+++ b/hello_world.txt
@@ -1 +1 @@
-Hello World!
\ No newline at end of file
+Hello World! It's a beautiful day!
\ No newline at end of file
```

5. To integrate the changes, type in
   $ git merge new_edits

   Assuming there are no conflicts, this will add the new sentence to the master hello_world.txt file.

   If we try to merge two branches where the differences in files don’t amount to simple additions/subtractions, we will have to manually resolve the conflicts in the two versions by opening them on a text editor and then merge.

Remote Repositories

1. We can also “clone” remote git repositories (usually hosted on github) into local copies, and merge changes in this local copy back into the remote server. In the process, we also have to make sure we integrate real-time changes in the remote copy into our local copy.

To clone an existing remote repository, navigate to the directory where you want the new repository to exist and type in

$ git clone https://github.com/MalavikaMukundan/sample_repo
$ cd sample_repo

2. Add a new file called test.txt to your local copy of sample_repo, and commit this change.
   $ touch test.txt
   $ git add test.txt

$ git commit -m “added test file”
$ git status

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

3. Next, add this file to the remote repository by typing in
   $ git push

4. If the remote copy has had more modifications than your local copy, to update your local copy, type in
   $ git pull