MATH40006: An Introduction To Computation
Git and GitHub

How To Use Git and GitHub to do Problem Sheets

Short summary; skip to the next subsection for a fuller walk-through

First, make sure you're on a machine with Anaconda and GitHub Desktop installed, and that you have an invitation to accept the current assignment, complete with a Web link. Then:

- Copy the Web link into the address field of your favourite browser.
- **The first time only**, choose your short form user name from the list you see.
- Click on **Accept this assignment**. Wait for a minute or so, then refresh the page.
- Follow the link you now see.
- Pull down the menu from the **Code** button and choose **Open with GitHub Desktop**.
- Click on the button labelled **Clone**.
- In the GitHub Desktop window, choose **Show in Explorer**.
- Launch Anaconda and Jupyter Notebook, and within Jupyter Notebook, navigate to the correct folder; click on the `.ipynb` file.
- Do the problem sheet! On a regular basis, head back to the GitHub Desktop menu, type in a summary of where you're at and hit **Commit to main**.
- When you're done, and you're *sure* you're done, head back to GitHub Desktop for one last commit, following which you should hit **Push origin**.

GitHub? What's a GitHub?

In British English, some computing terms that come from America can land a little oddly (we could cite some genuinely arresting examples!) But this isn't actually an example of that. GitHub is based on Git, and the developer of Git, Linus Torvalds, gave it that name in full knowledge of what Git means on this island. It's a software engineer's gag; and unfortunately, not the last we'll come across on this module.

But what is it? Well, for a fuller answer to that question you'll need to wait till later in the module. The short version is, first, that Git enables us to keep tabs on how the files we're working on evolve over time, and to have access to earlier or alternative versions in an orderly way. This is called **version control**, and more about it in a few weeks. Secondly, GitHub moves this version control online, into the Cloud, which enables all sorts of things, such as collaboration on coding projects–or, as here, downloading coding exercises and submitting your answers to the teaching team.

The software you'll need

If you're already into Git and GitHub, then you may wish to fly solo, using whatever approaches, and whatever software applications, work for you. But if you're not familiar with Git or GitHub, or if you're not 100 per cent confident that what you've experienced so far will work in this context, then here's what we recommend.

You should download an application called GitHub Desktop, which is available at

`https://desktop.github.com/`

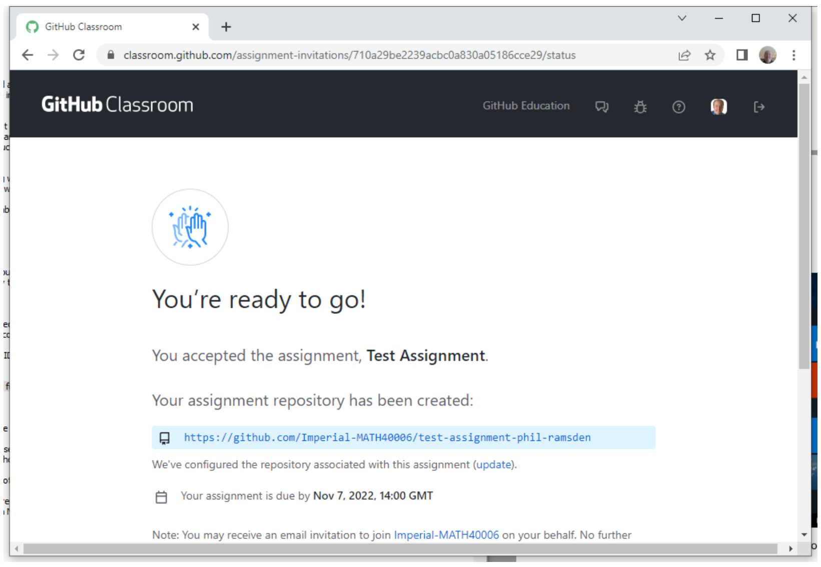
And that's it! That's all the GitHub-related software you'll need. (Of course, you'll also need Anaconda to actually do the tasks.) Before you attempt a problem sheet, make sure you have GitHub Desktop and Anaconda downloaded and installed on whatever machine you're using. (If you use a Departmental desktop machine, you'll find they're already there.)

Accepting an assignment

Let's imagine you're working on a machine that has GitHub Desktop installed, and that you've received, via a Blackboard announcement, an invitation to accept an assignment. This will come with a *link* to the assignment; a web address. The first thing to do is to copy that web address into the address field of a Web browser; Chrome and Firefox both work quite well.

The first time you do this, you'll then be asked to choose your User ID from a long list. It's really important that you go through this step, though you'll only have to do it for the first assignment; from then on, this step won't arise.

After clicking on the button labelled **Accept this assignment**, waiting for a minute or so, and then refreshing the page, you should end up at a page that looks something like this:


GitHub Classroom
classroom.github.com/assignment-invitations/710a29be2239acbc0a830a05186cce29/status
GitHub Classroom
GitHub Education

You're ready to go!

You accepted the assignment, **Test Assignment**.

Your assignment repository has been created:

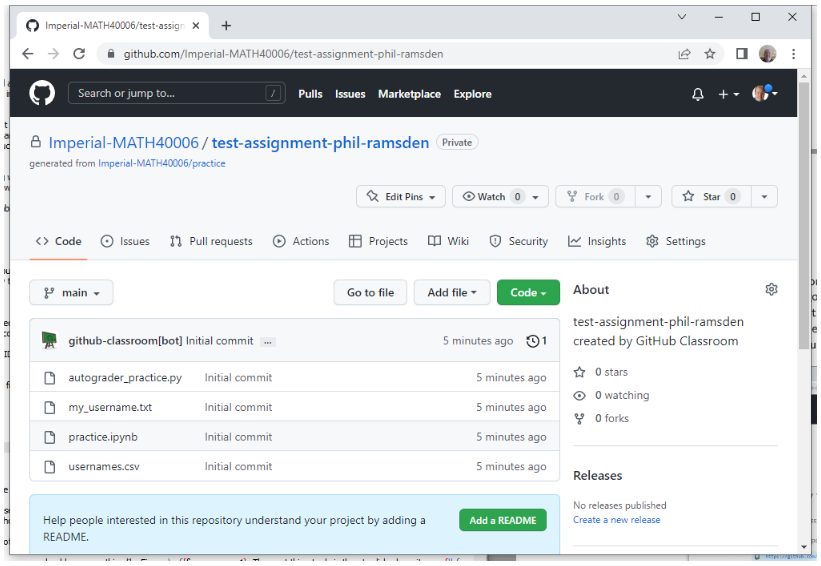
`https://github.com/Imperial-MATH40006/test-assignment-phil-ramsden`

We've configured the repository associated with this assignment (update).

Your assignment is due by **Nov 7, 2022, 14:00 GMT**

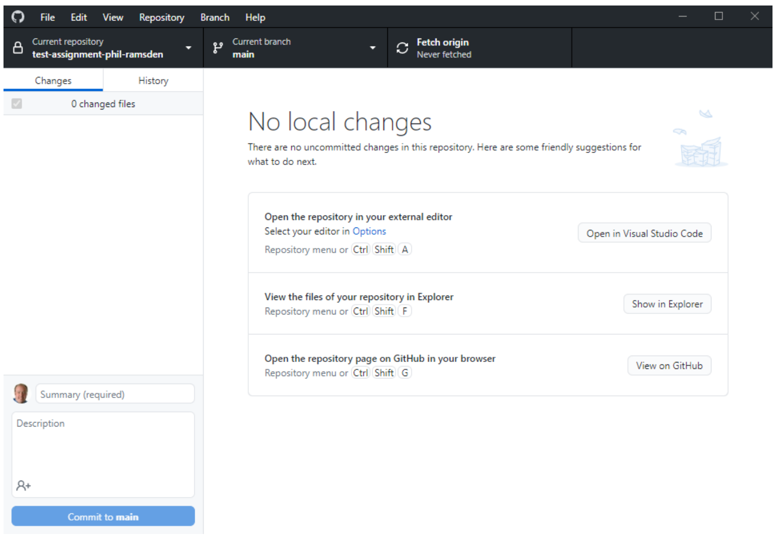
Note: You may receive an email invitation to join Imperial-MATH40006 on your behalf. No further

Follow the link, and the page will look like this:



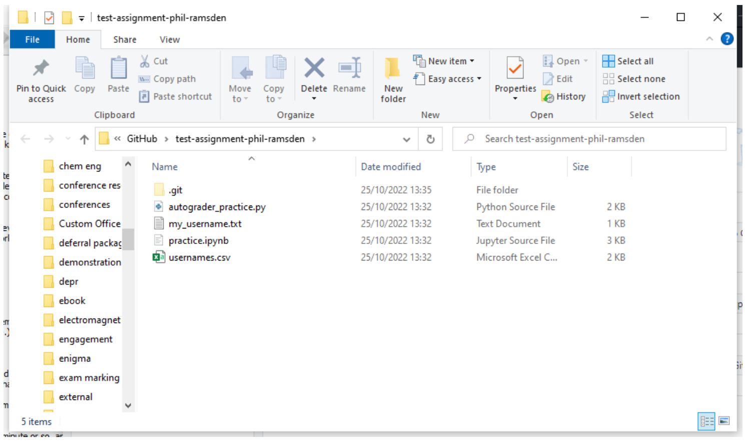
Opening in GitHub Desktop

The green button labelled **Code** opens a drop-down menu. You should select the option **Open with GitHub Desktop**. When you do that, it should open the GitHub Desktop application, and you'll be invited to **Clone**; you should do that. Then the window will look something like this:


File
Edit
View
Repository
Branch
Help
Current repository
test-assignment-phil-ramsden
Current branch
main
Fetch origin
Never fetched
Changes
History
0 changed files
No local changes
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.
Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A
Open in Visual Studio Code
View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer
Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub
Summary (required)
Description
Commit to main

You should click on the button labelled **Show in Explorer** (if you're using a Mac, this will say something like **Show in Finder**).

This will open a folder *on your local machine* containing all the files you'll need to do the problem sheet:

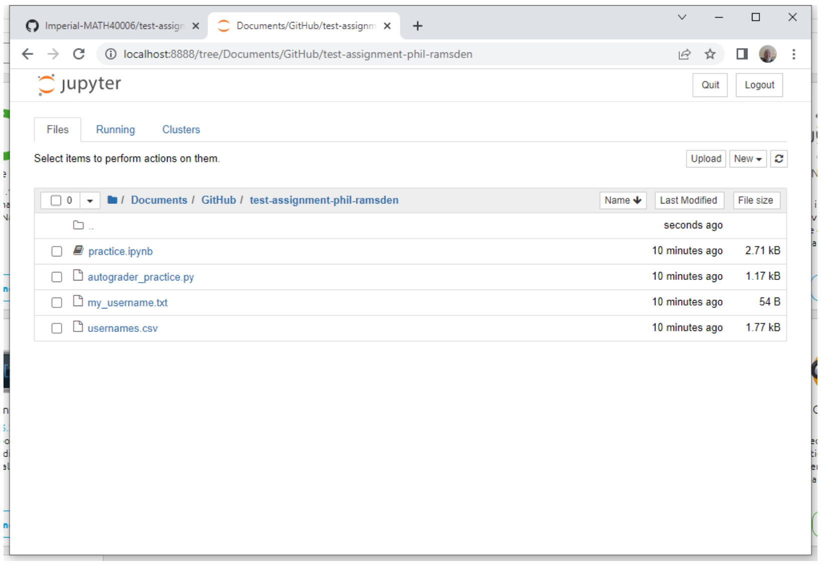


Name	Date modified	Type	Size
.git	25/10/2022 13:35	File folder	
autograder_practice.py	25/10/2022 13:32	Python Source File	2 KB
my_username.txt	25/10/2022 13:32	Text Document	1 KB
practice.ipynb	25/10/2022 13:32	Jupyter Source File	3 KB
usernames.csv	25/10/2022 13:32	Microsoft Excel C...	2 KB

You're ready to go!

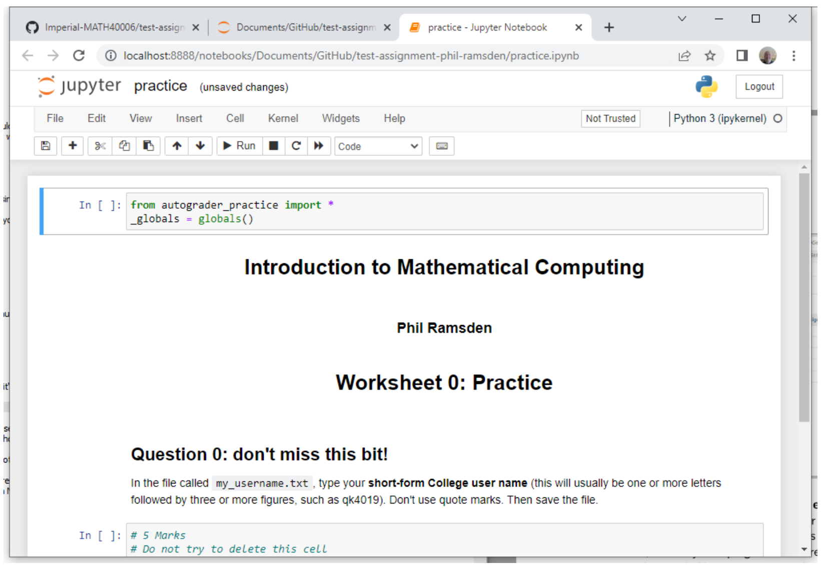
Starting to do the problem sheet

Now you know where the problem sheet is on your machine, it's time to launch Anaconda and then click on **Jupyter Notebook**. Then you should navigate to the correct folder:


Imperial-MATH40006/test-assig
Documents/GitHub/test-assignm
localhost:8888/tree/Documents/GitHub/test-assignment-phil-ramsden
jupyter
Quit
Logout
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/ Documents / GitHub / test-assignment-phil-ramsden
Name
Last Modified
File size

Name	Last Modified	File size
..	seconds ago	
practice.ipynb	10 minutes ago	2.71 kB
autograder_practice.py	10 minutes ago	1.17 kB
my_username.txt	10 minutes ago	54 B
usernames.csv	10 minutes ago	1.77 kB

The problem sheet is the file whose extension is `ipynb`; in this case, it's `practice.ipynb`. If we click on that, it looks like this:


Imperial-MATH40006/test-assign
Documents/GitHub/test-assignm
practice - Jupyter Notebook
localhost:8888/notebooks/Documents/GitHub/test-assignment-phil-ramsden/practice.ipynb
jupyter practice (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [ ]: from autograder_practice import *
        _globals = globals()
```

Introduction to Mathematical Computing

Phil Ramsden

Worksheet 0: Practice

Question 0: don't miss this bit!

In the file called `my_username.txt`, type your **short-form College user name** (this will usually be one or more letters followed by three or more figures, such as qk4019). Don't use quote marks. Then save the file.

```
In [ ]: # 5 Marks
        # Do not try to delete this cell
```

At this point, the task is simply to **do the problem sheet**. This practice sheet won't take us much time, because all it contains is one question, labelled **Question 0**, worth five marks. Every problem sheet will include a Question 0, and they'll all be identical!

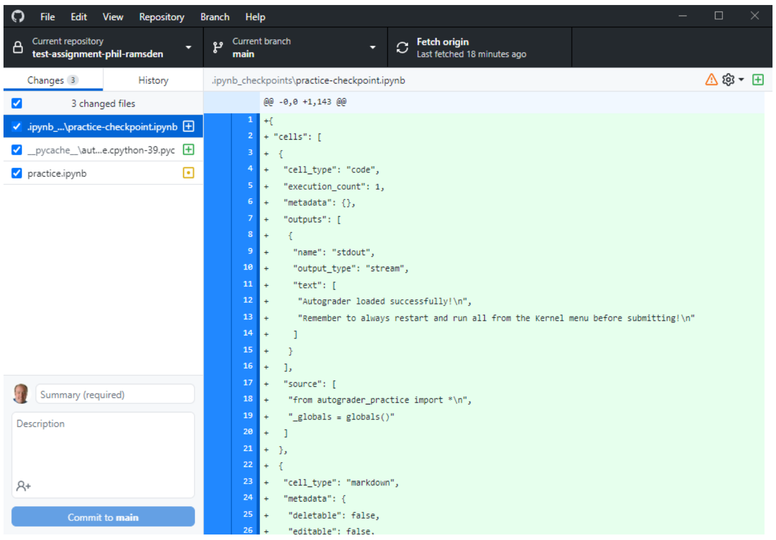
First, we click in the very first cell, hold down Shift and press Enter. You should get some output saying something like

```
Autograder loaded successfully!
Remember to always restart and run all from the Kernel menu before submitting!
```

Next, if we navigate to the next code cell and do "Shift-Enter", we get a discouraging message:

```
You don't seem to have changed the contents of the file.

0 out of 5 marks
```

You may already have worked out what to do to fix this, but let's stop at this point, and save this notebook (by clicking on the Save icon). If you now return to the GitHub Desktop window, it should look something like this:



.ipynb_checkpoints\practice-checkpoint.ipynb

```
@@ -0,0 +1,143 @@
+{
+ "cells": [
+  {
+   "cell_type": "code",
+   "execution_count": 1,
+   "metadata": {},
+   "outputs": [
+    {
+     "name": "stdout",
+     "output_type": "stream",
+     "text": [
+      "Autograder loaded successfully!\n",
+      "Remember to always restart and run all from the Kernel menu before submitting!\n"
+     ]
+    }
+   ],
+   "source": [
+    "from autograder_practice import *\n",
+    "_globals = globals()"
+   ]
+  },
+  {
+   "cell_type": "markdown",
+   "metadata": {
+    "deletable": false,
+    "editable": false,
```

The commit cycle

We'll explore version control in a little more depth later in the module, but it's good to get into good habits now. Whenever you've done a chunk of work on a problem sheet, you should go to the GitHub Desktop window and **commit** your changes. The minimum requirement is that you type a short description of what you've done where it says **Summary (required)**; we could type something like

Autograder imported; first question attempted incorrectly

Then you should hit the button labelled **Commit to main**.

Every time you make changes to any of your files, it's possible to commit those changes with a brief description like this. It's a good habit to get into doing this fairly frequently; and you should *always* do it before ending your session.

Completing the exercise and submitting

Question 0 is fairly easy to complete. All you need to do us open the file `myusername.txt` in a **text editor** (such as Notepad on a Windows machine), and replace the text that's there with your **short form College ID**, which will be a combination of letters and digits, like `xa1014`.

Then, when you go back to the Jupyter notebook and "Shift-Enter" on the question cell, you should get the following response:

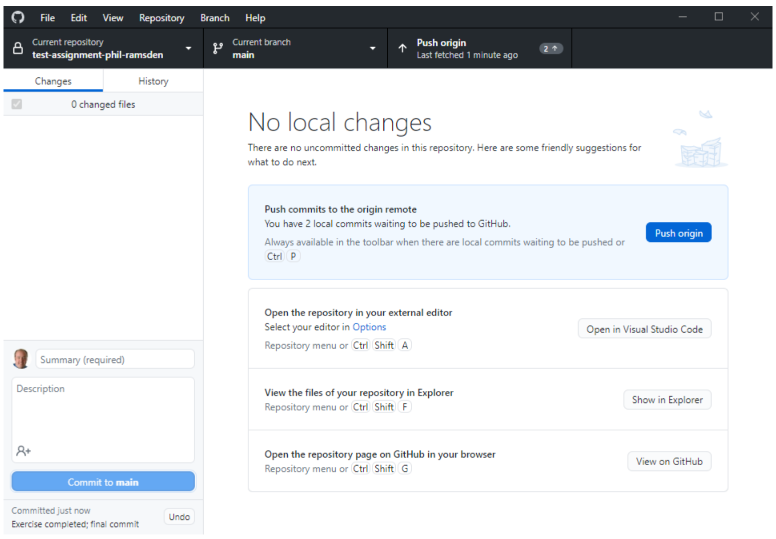
```
You've changed the contents of the file; thank you!
Your username has been recognised; thank you!

5 out of 5 marks
```

(If you don't, please contact Phil or Sam!)

That's this problem sheet completed (obviously, the others will take longer). Save the notebook, then go back to GitHub Desktop and do one last commit; we could use the description

Exercise completed; final commit

The GitHub Desktop window will now look like this:



Current repository
test-assignment-phil-ramsden

Current branch
main

Push origin
Last fetched 1 minute ago
2 ↑

0 changed files

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push commits to the origin remote
You have 2 local commits waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or Ctrl P

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

Summary (required)
Description

Committed just now
Exercise completed; final commit

There's just one more thing to do; but this is a really important step, so don't miss it out. You need to click on the blue button labelled **Push origin**. (If this has disappeared—which happens if you leave it too long—the **Push** item from the **Repository** menu has the same effect, as does "Ctrl-P".)

And you're done: you've completed a problem sheet and submitted it for us to have a look at. Well done, and thank you!