1. The command line and bash scripting

We would like you to use the command line to build a file tree within your homework repository as shown below.

```
s1
|---s3
|   |---conf.txt
|---s2
    |---text_chunk1.txt
    |---Advanced
         |---text_chunk2.txt
```

- Record your commands in a text file `answers.txt` .
- Commit and push the file structure and the answers.txt to your remote homework repository.

- **conf.txt** should contain the sentence:

```
"I love bash scripting." on the first line.
```

- **text_chunk1.txt** Should say somthing else .
    - For example:

```
"A whole new world"
"A new fantastic point of view"
```

- **text_chunk2.txt** Should be a copy of text_chunk1.txt with extra text appended.

For example:

```
"A whole new world"
"A new fantastic point of view"
"Do you want to build a snowman?"
```

1a. Writing a Bash script

Shell scripts are just files with lines of Shell commands. They are designed to be run at the command line terminal and it's actually a full featured language (although no one programs completely in shell scripts). Shell

scripts are an automated way of programming tedious tasks, repeated sets of commands that need to be performed.

For example, you might write one to run a series of python programs every morning or when you deploy a website. You might write one to automatically create a file structure for a project (like a project template).

- Since you are on different systems choose either the Mac/UNIX instructions **or** the Windows instructions below

Mac/Unix

We'd like you to write a bash script that generates the above file tree and save it as make_tree.sh. To do so, you can simply save the commands you use to generate the file tree in a .sh file. Note that you cannot use the "cd" command.

You do not need to execute this bash file as we will read it manually in order to grade it! There are several ways of executing bash files. Most simply, you can execute them via a command line call like sh make_tree.sh.

In order to make this file executable. You should:

- add a "shebang" to the top of your file, which points to your bash implementation.

```
#!/bin/bash
```

in this case bash is in /bin/bash which I can see by calling: **which bash** or **type -a bash**

- make the file executatable.

```
chmod +x make_tree.sh
```

Windows

We'd like you to write a bat script that generates the above file tree and save it as make_tree.bat. To do so, you can simply save the commands you use to generate the file tree in a .bat file. Note that you cannot use the "cd" command.

You do not need to execute this bat file as we will read it manually in order to grade it! There are several ways of executing bat files. Firstly you can execute them via a command line call by simply typing make_tree.bat. You can also just double-click on the file from a Windows folder.

In order to print text to a file from a .bat script, you must type the command slightly differently than we did in the command line.

Instead of:
"this is some text to output" >> output_file.txt

You will need to use:
@echo this is some text to output >> output_file.txt

2. Playground

Please make at least two commits to the github playground. They must be seperated by at least two days and can be small but they must edit a single file called `edit.txt`. This is to show you how to deal with merge conflicts and multiple editors [ie people editing the repository]. *If this file does not exist, please go ahead and be the first to create it.*

A couple of notes:

To clone into the github playground, you do not need to set up a "three way" repository like you have for your homework repository. You can instead use the default git command, "git clone" to clone the repository. You can then use just "git pull" and "git push" to pull from and push to the repository.

You may find these instructions helpful to clone into the repository (note, that the repository name will not have your username but instead will have the course group name): https://help.github.com/articles/cloning-a-repository/

You may also run into a "merge conflict" if you try submitting your changes to `edit.txt` at the same moment in time as somebody else. These conflicts occur only when two people edited the same lines of code at the same time. Git does not know how to resolve the conflict, so it asks the user to help.

Some merges are easy and will be done automatically however some are more complicated and have to be done manually. This resource can help you resolve difficult merge conflicts:

https://help.github.com/articles/resolving-a-merge-conflict-from-the-command-line/

Remember - when in doubt, use "git status". Good luck!

Checklist.

Please be sure you submit the following:

1. Your folder structure, as discussed in part 1.
2. Answers.txt, as discussed in part 1.
3. Your bash script, as discussed in part 1.

4. Two github-playground commits, separated by at least a day, as discussed in part 2.