Exercises for Git

Introductory Programming 2020

1. Make sure that Git is installed on your system and retrieve the version of your Git installation by running the correct command in your terminal.

2. Let’s imagine that we start out a new project from scratch and we want version control. Open VSCode, create a folder somewhere, create a Java-file named Main.java in this folder and put in some simple code (the content doesn’t matter – just make sure that it can compile (you don’t get any errors when trying to run/debug the code)). Save your Java-file.

   The following steps should preferably be done by typing commands into your terminal

   a. Make sure that your terminal is currently in the desired working folder (type ‘ls’ and make sure that you can see your newly created Java-file.
   b. Now, initialize Git in this project .
   c. Manually add the java file you just created to your newly initialized Git repo.
   d. The next step is to commit your ‘changes’ to your repo.

3. Now, compile your java-file in your terminal. Observe that a new file is now created in your working folder called SOMETHING.class
   If we don’t want Git to track this compiled file(or simply, never track .class-files in general) what can we then do?
   (Hint: It has something to do with ignoring certain files/file types)

4. Add yet another .java-file to your working directory. Put some simple code inside and compile the file. It could be this:

```
public class Something {

        private int number;

        public Something() {

                this.number = 5;

        }

}
```

   Also, add a file called inp.txt, Add some content to it and save it.
   At this point, if you did #3 correctly, you can type one single Git-command which will add both the new .java-file and the inp.txt-file to your Git repo while NOT adding the .class-file(s).

(Hint: the command has something to do with adding files while ignoring certain other files.)

5. At this point we wish to save all our hard work somewhere remote just in case we accidentally drop our laptop down the toilet.
   Start by logging in to Github online. You can use github.itu.dk and your ITU-credentials. Create a new repository. There's a button for this somewhere. Give the repo a name. Make sure you don't initialize this repo with a README. After pressing 'Create Repository' you should be presented with instructions on how to push an existing repo to this newly created remote.

   Now, go back to your terminal, add the remote and push your files to the remote. If successful, you will see some confirmation in the terminal (several lines of code) and your files are now visible online in your Git-repo. (notice that locally, you have all the `.class`-files while they're not visible online)