XCS236 - Environment Setup

Git Setup

To setup Git on your local machine, follow the setup docs from Github. For general git information, please reference the provided handout. Once Git is set up on your machine, clone the problem set repository to get the relevant files on your machine.

> *Note: It is recommended to clone as opposed to downloading so you can pull any changes to main into your local branch if needed.*

Unset
```
# OPTION1: SSH
git clone git@github.com:scpd-proed/XCS236-PS1.git

# OPTION 2: HTTPS
git clone https://github.com/scpd-proed/XCS236-PS1.git
```

If you track your work using a remote version control, please **set your repository to private**.

Conda Environment

Please reference the Anaconda Setup handout for thorough instructions, and the Conda docs for a more general guide. For the rest of the Conda setup, I will assume the current working directory is `XCS236-PS1/src`.

The problem set repository provides an environment.yml (or enviorment_cuda.yml for CUDA enabled systems) which will be used to set up the environment. If you are using Apple silicon (M1/M2/M3) or don't have a discrete GPU, it is recommended to use the CPU environment.

Unset
```
# OPTION 1: CPU
conda env create --file environment.yml

# OPTION 2: GPU
conda env create --file environment_cuda.yml
```

This will take a few minutes to complete. Once done, you can activate the environment

```
Unset
conda activate XCS23
```

With the activated environment, Python will use the defined dependencies. When you are done with your assignment or need to exit the environment for any reason, you can deactivate

```
Unset
conda deactivate
```

To use the environment in a Jupyter Notebook, please reference section 3 of the Anaconda Setup handout.

Code Editor

You are welcome to use any code editor that you prefer. Many students use VS Code. This has extension support that can be used for debugging, jupyter notebooks, or multiple other customizations. Editors and IDEs such as Vim or PyCharm work as well. It is also recommended to use a debugger, such as PDB, to make resolving errors simpler.

Autograder

Once you have made progress in your assignment, you can run the program using

```
Unset
python main.py
```

This will first download any required files for the problem set, which may take a while. Then you can run the local grader using

```
Unset
python grader.py
```

Note that you may get errors if some parts are not implemented, these are expected. For more details on individual tests, the remote grader, and making submissions, please reference the PS 1 handout.

LaTeX / Overleaf

You may also wish to use LaTeX for your written assignments. The required files are provided as part of the assignment, and can be compiled locally or using Overleaf. Both approaches are detailed in the README or PS1. Of course, you are also welcome to handwrite and scan your assignments instead. If you choose to do so, submissions must be clear and legible.