1. Install CyberDuck

https://cyberduck.io
- Simplest way to access files on the CS lab server
- Enter configuration on right and double click icon below to access server

****For Windows users:
- Need to download mobaXterm
- https://mobaxterm.mobatek.net
- Or any similar program
2. Open Terminal and SSH into CS Lab Server

`ssh <netID>@login.cs.duke.edu`  (example: `ssh abc35@login.cs.duke.edu`)

- In terminal, run above command to ssh with CS Lab account credentials (will probably need 2-factor authentication) and should see something like this:

```plaintext
*******************************************************************************
  * Welcome to the Computer Science department's Login Servers. *
  * These computers are made available to the members and *
  * guests of the CS department as general work resources. *
  * We ask that applications that can be run on a user's *
  * desktop, e.g., chrome, xemacs, etc., not be run on the *
  * Login Servers. Please remember that these are shared *
  * resources, and are not intended to be compute servers. *
  * Please use desktop machines or other public machines as *
  * compute servers. For CPU-intensive research applications, *
  * dedicated Compute Servers are available. For more information *
  * on the cluster, please see: *
  * https://www.cs.duke.edu/csl/facilities/cluster *
  * https://www.cs.duke.edu/csl/faqs/slurm *
*******************************************************************************
```
3. Make and Activate Python Virtual Environment

After SSHing into the login server:
- `mkdir Documents`
- `cd Documents`
- `mkdir OspreyMain`
- `cd OspreyMain`
- `python3 -m venv Osprey`
- `source Osprey/bin/activate`

After SSHing into the CS server, you should be at `/home/users/<netID>` (check with "pwd")

From here I make a "Documents" folder, and move inside. Then I make a folder called OspreyMain and move into it. In my "OspreyMain" directory, I run the above Python3 command, which generates “Osprey.” Inside this folder is the activation script. Running "source" on this script inside the terminal will activate the Python virtual environment.
4. Clone Github Repo

While within the “OspreyMain” directory do:
- Git clone [copied URL]

This will copy the Main branch of OSPREY3 into this directory. You will see the folder below in your Documents folder (my URL is different). If it doesn’t show up, press refresh on Cyberduck.

Press this to copy URL

We will work out of here to install OSPREY
5. Edit Build Script

Change Directory into the downloaded OSPREY3 directory:
- cd OSPREY3

We want to edit “build.gradle.kts” to make compatible with virtual environment
- Can use Vim or Cyberduck to move the file onto your computer for editing
- Delete 4 instances of “--user” (the highlighted spots)
- Save changes (drag back onto the server using Cyberduck)

```kotlin
val python2InstallScripts by creating {
    group = "build",
    description = "Make install scripts for python 2 distribution"
    doLast {
        val pythonDir = pythonBuildDir.resolve("python2")
        pythonDir.copyRecursively(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
    }
}
```

```kotlin
val python3InstallScripts by creating {
    group = "build",
    description = "Make install scripts for python 3 distribution"
    doLast {
        val pythonDir = pythonBuildDir.resolve("python3")
        pythonDir.copyRecursively(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
        pythonDir.copy(pythonBuildDir)
    }
}
```
6. Install

1. Need to set to latest Java 17
   - **Check**: `java --version`
   - **Run**: `export JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64`
     - Will need to run this every time unless you set `bash_profile` (can google how to do this)

2. **Run**: `./gradlew assemble`
   - Will get some errors here, we can ignore some modules
   - **Run**: `./gradlew assemble --x makeDoc --x testDistZip --x python2Install --x python2Uninstall`
     - These aren’t necessary for the class
   - If still have errors:
     - **Run**: `./gradlew assemble --x makeDoc --x testDistZip --x python2Install --x python2Uninstall --x python3Wheel`

3. **Run**: `./gradlew pythonDevelop` (setup Python environment)
7. Running Examples

Three Steps:
1. Prepare OMOL file from structure (PDB of bound complex)
2. Prepare conformation space
3. Run Design Algorithm of choice (we will use Kstar)

On the server, all jobs must be submitted to the cluster via Slurm (sample scripts provided):
- These will be submitted using:
  - `sbatch <slurm-script>.sh`
You can check the status of your jobs with:
- `squeue -u $USER`
To cancel job:
- `scancel <jobID number>`

Can edit the SBATCH parameters to fit your scripts, for the purposes of the class: `nodes=1`
Summary – Running OSPREY

1. SSH into CS server
   - `ssh <netID>@login.cs.duke.edu`

2. Activate Python3 virtual environment
   - `cd /Documents/OspreyMain/` (change into the OspreyMain directory)
   - `source Osprey/bin/activate`

3. Set Java version to 17
   - `export JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64`

4. Modify example scripts for your needs:
   1. Prepare OMOL molecule file from PDB
   2. Prepare conformation space from OMOL file
   3. Run Kstar
   - Submit jobs to cluster using `sbatch`!
Other Useful Resources

   - How to set up SSH keys from the Duke CS Lab website (so you don’t have to enter password/2fac every time you SSH in terminal or use cyberduck)

   - Duke slurm FAQ for submitting jobs to cluster

3. Osprey@cs.duke.edu
   - Help email that will go to the lab – for questions/bugs while you’re using Osprey : )
   - Mention you’re a student and send logs/scripts you’re using so we can see what’s happening