eSim2022 BTAP Workshop Exercise 2 Instructions

Chris Kirney

2022-06-16

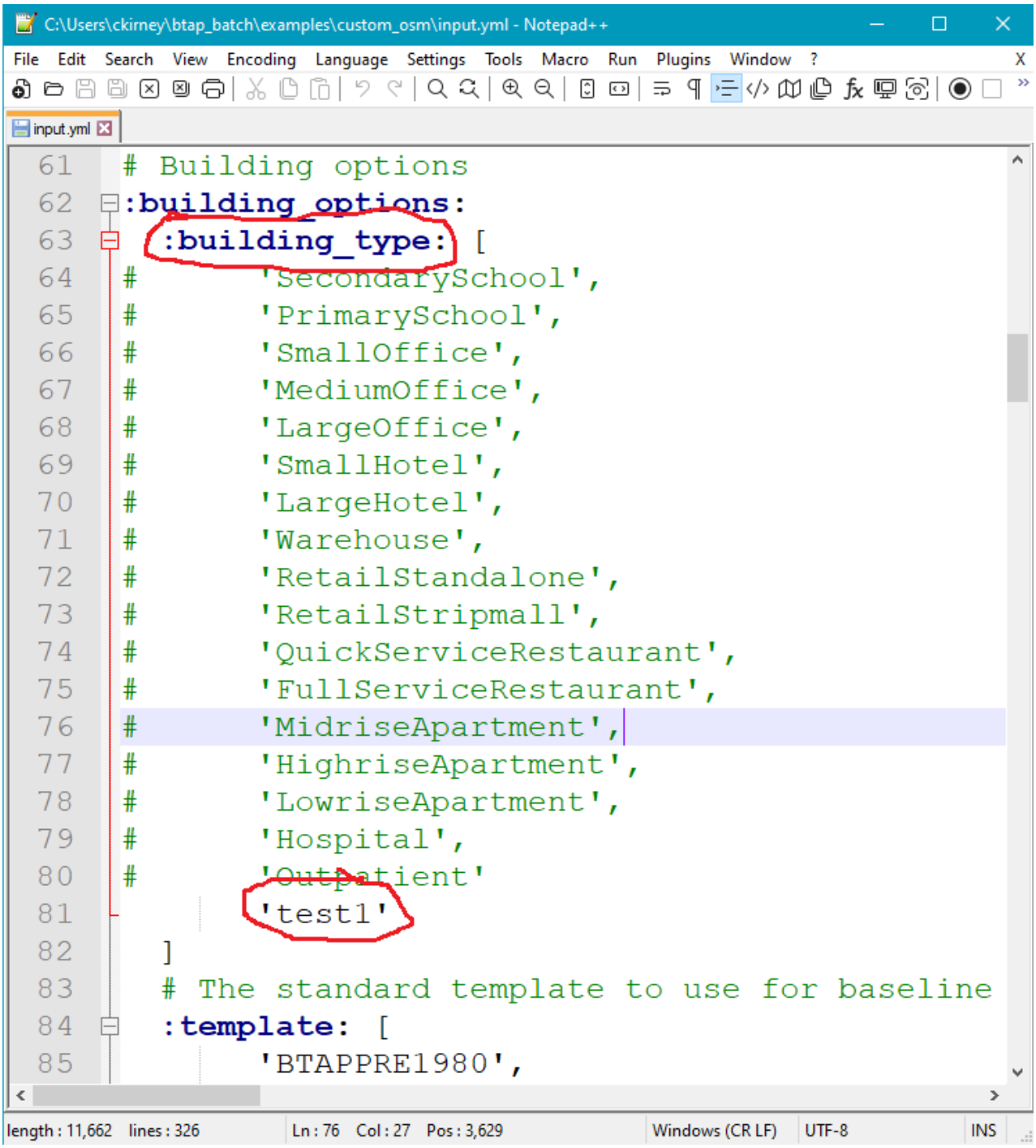
Exercise 2:

In this exercise we will run btap_batch with the custom geometry file we modified in exercise 1. If you didn't complete exercise 1, or you had trouble with your file, you can use the 'btap_batch/esim2022/exercise2.osm' file. This exercise assumes that you have already set-up everything you need to run btap_batch and have already tried a parametric analysis locally as per the README.md instructions here:

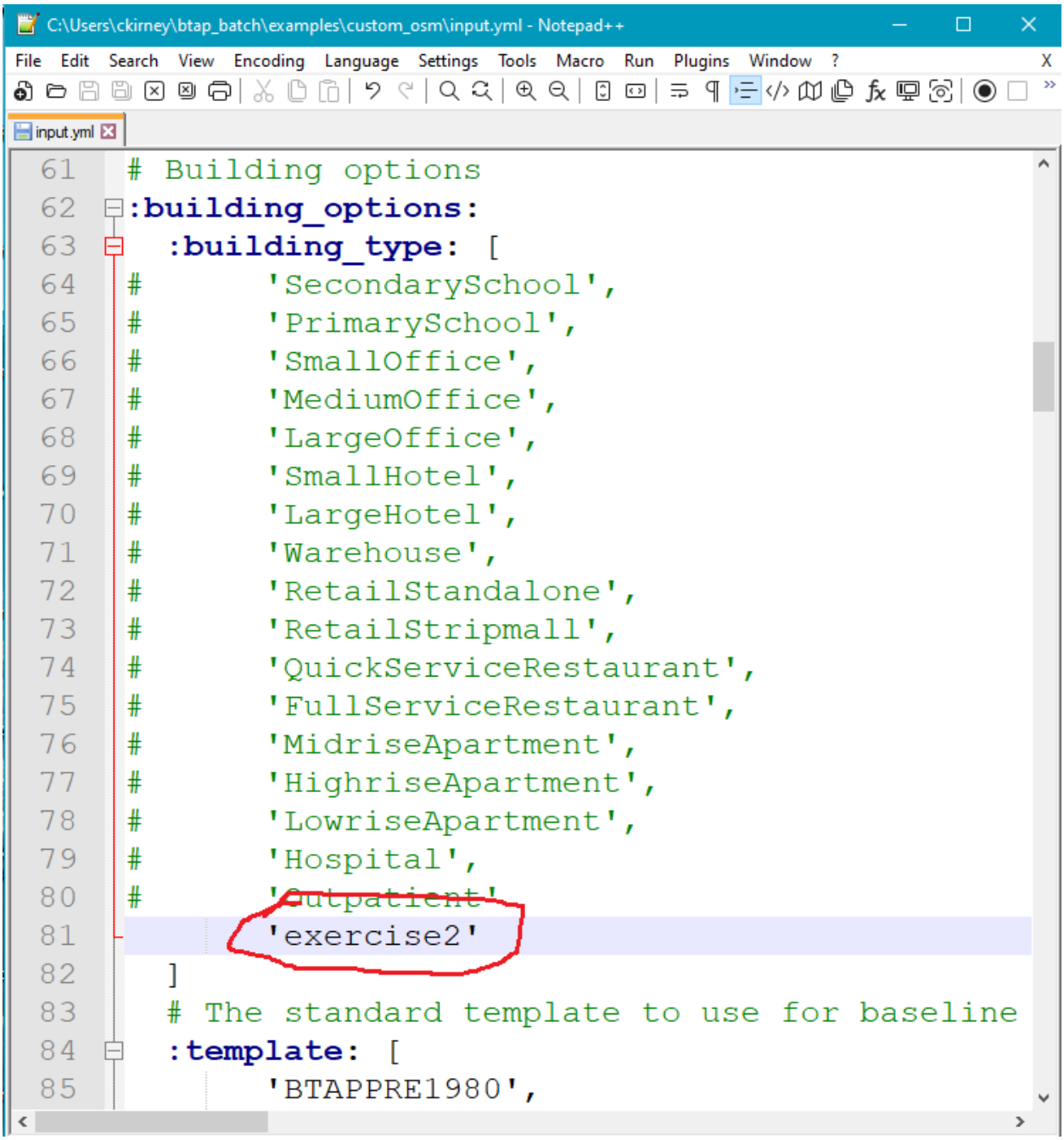
https://github.com/canmet-energy/btap_batch/blob/esim_mod/README.md

To continue your computer should have internet access.

1. Copy the file the completed in exercise 1, or the 'btap_batch/esim2022/exercise2.osm' file, to the 'btap_batch/examples/custom_osm/osm_folder/' folder.
2. Use a text editor (e.g. Notepad, Notepad++, Sublime Text, etc.) to open the 'btap_batch/examples/custom_osm/input.yml' file.
3. Look for 'test1' in the ':building_type:' array (see below):



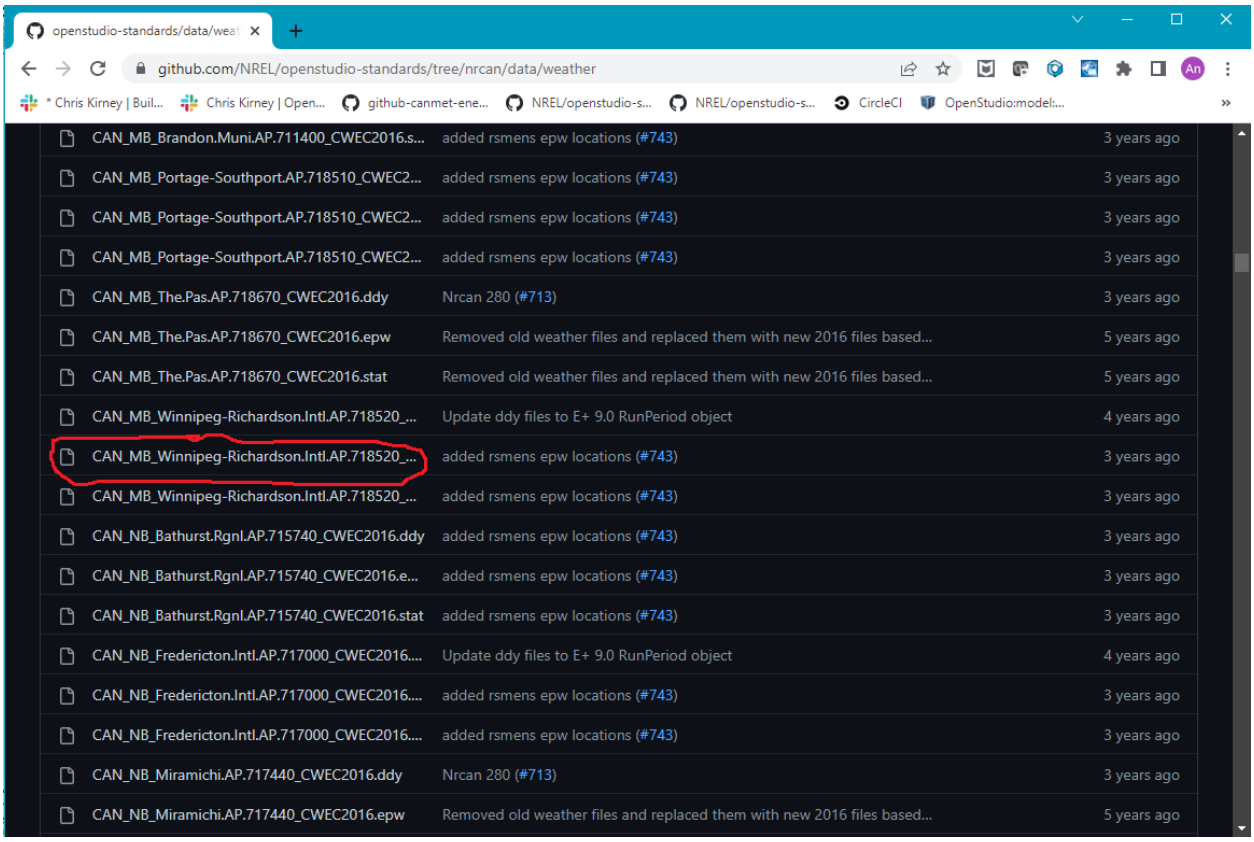
4. Replace ‘test1’ text with ‘exercise2’ or whatever is the name of the file (without the ‘.osm’ extension) you put in the ‘osm_folder’ (see below):



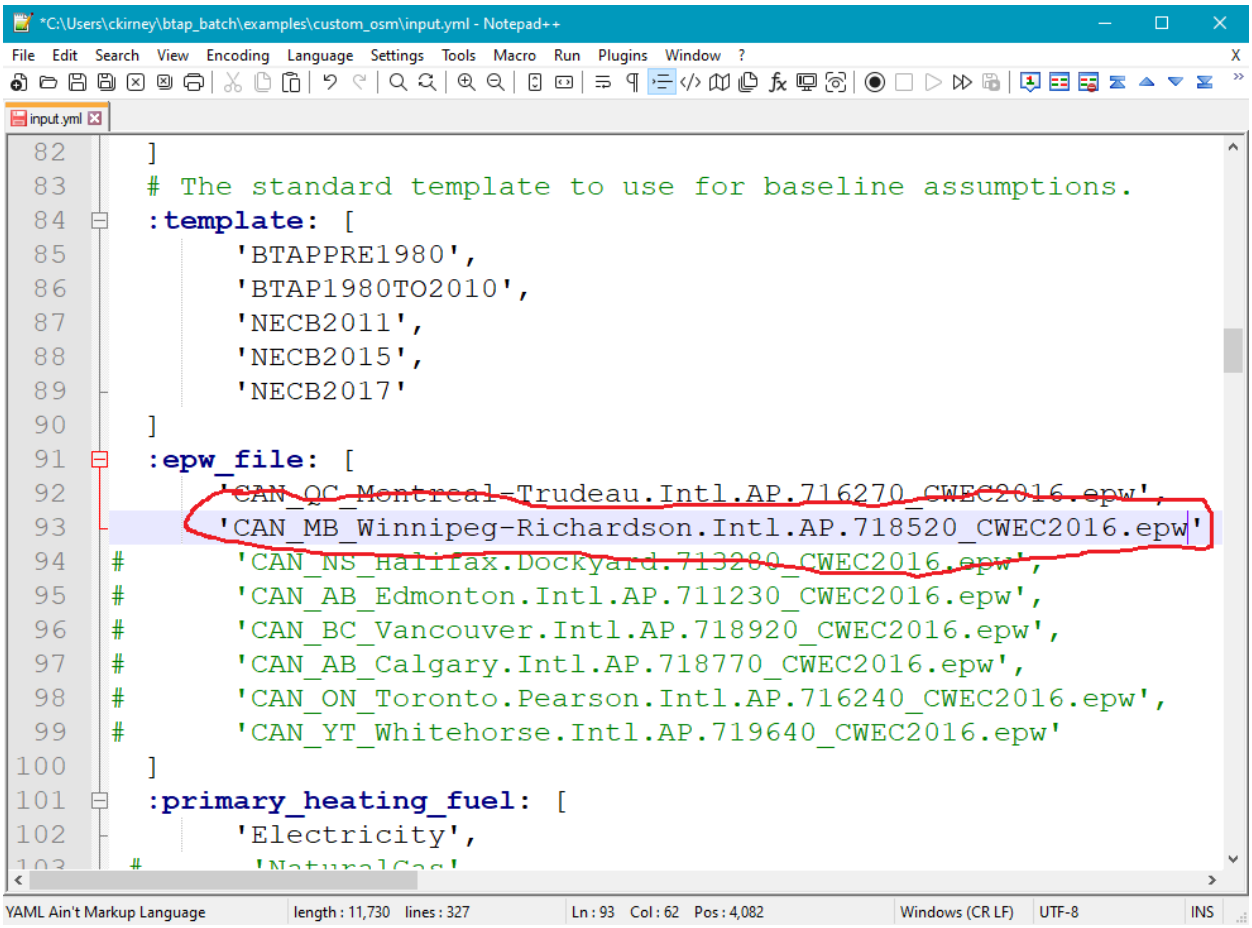
5. We will also use a new weather location. Use the link below to go to where the weather files are stored on the 'openstudio-standards' repository:

   https://github.com/NREL/openstudio-standards/tree/nrcan/data/weather

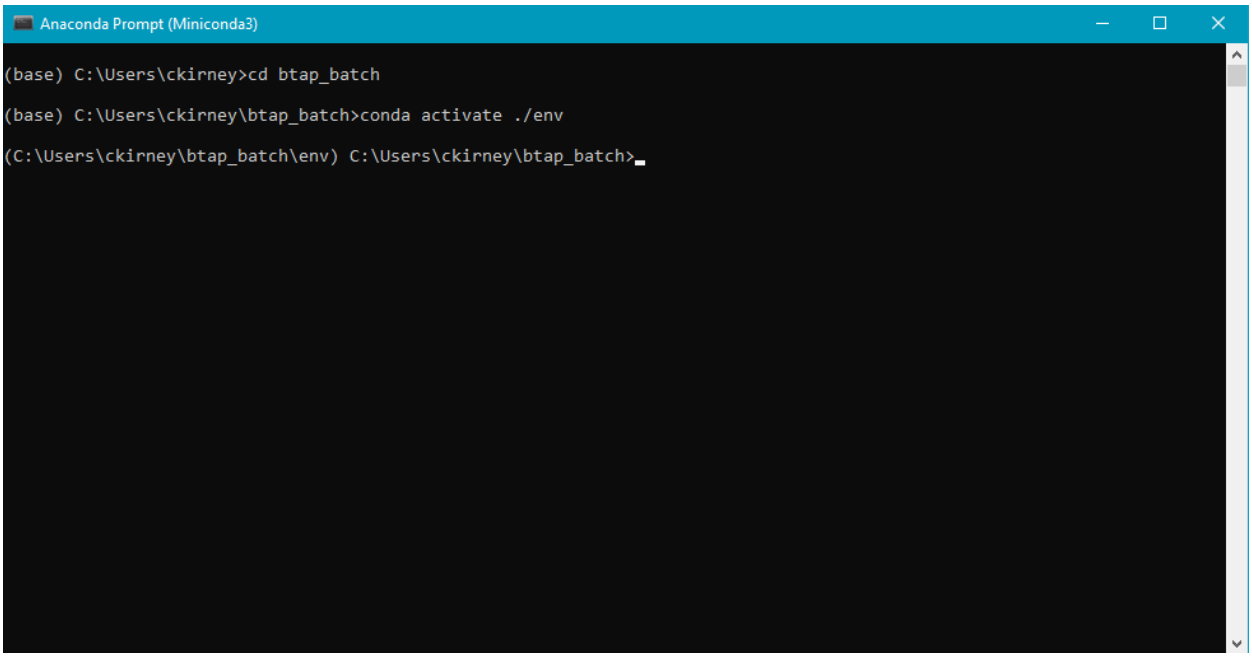
6. You can use whichever weather location there that you want as long as it has a '.epw', '.stat', and '.ddy' file associated with it. For this exercise we'll use Winnipeg (see below):



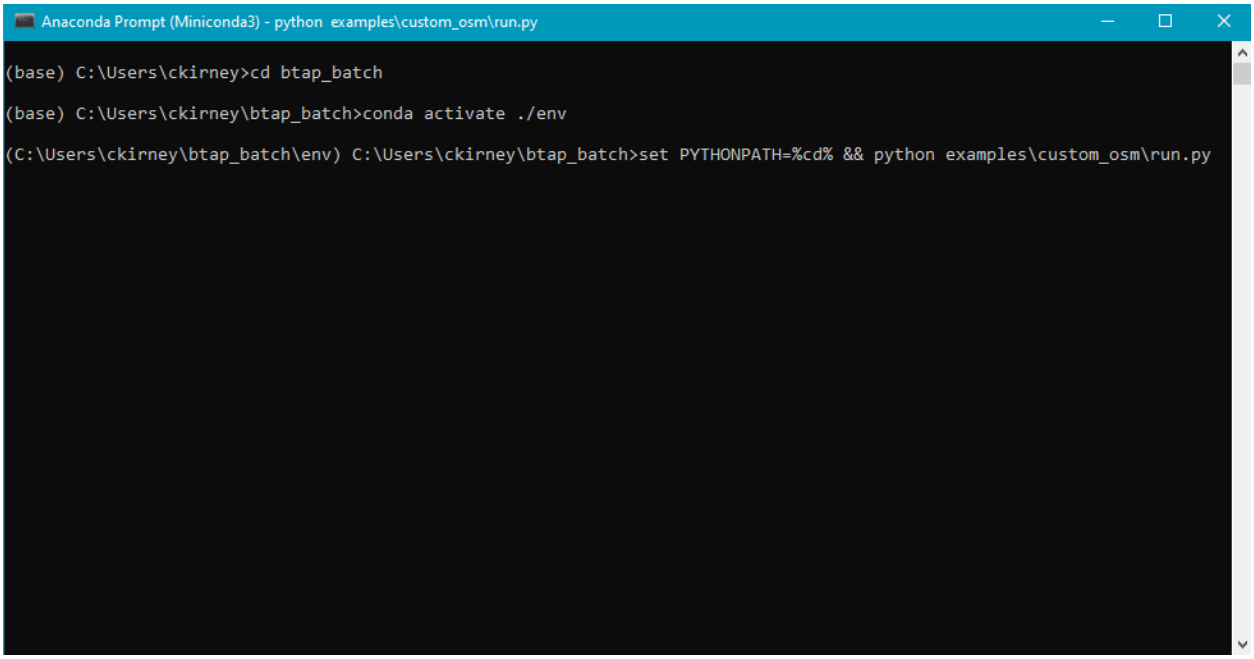
7. In the 'btap_batch/examples/custom_osm/input.yml' file add a line under CAN_QC_Montreal-Trudeau.Intl.AP.716270_CWEC2016.epw' in the :epw_file:' array. Put 'CAN_MB_Winnipeg-Richardson.Intl.AP.718520_CWEC2016.epw' in the space you just created (see below):



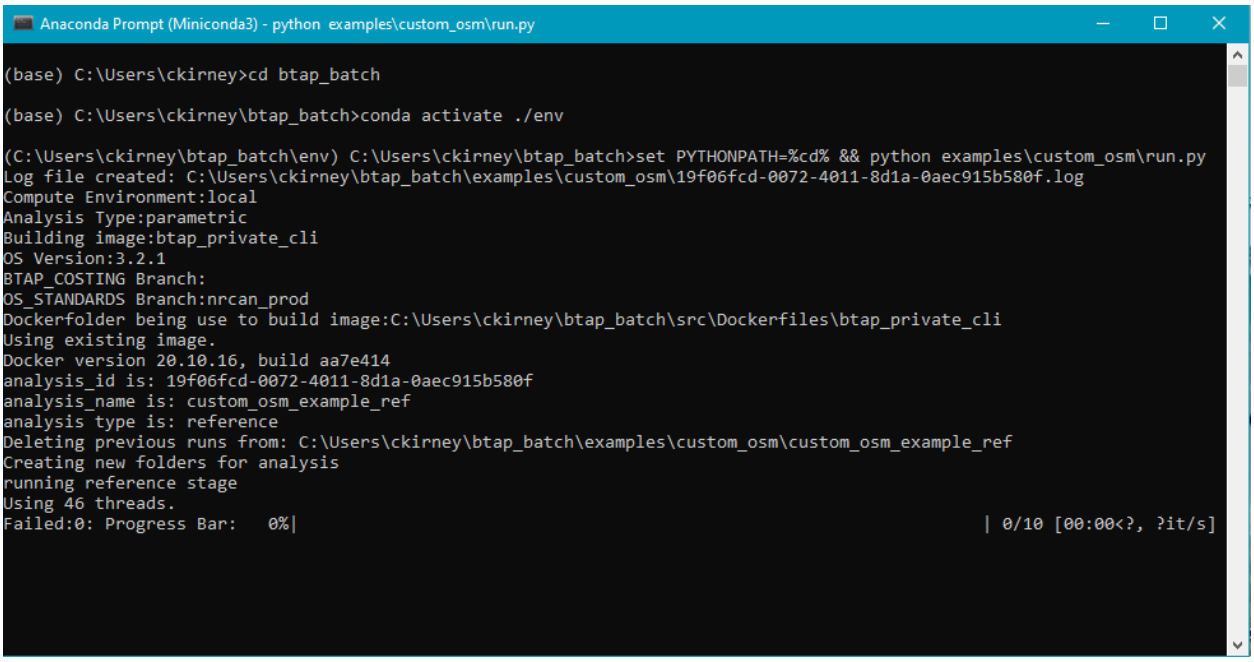
8. Save and close the 'btap_batch/examples/custom_osm/input.yml'.
9. Make sure Docker Desktop is running and open the 'Anaconda Prompt (Miniconda3)'.
10. In the 'Anaconda Prompt (Minicond3)' window navigate to 'btap_batch'.
11. Assuming you created the Anaconda environment as per the btap_batch README.md, activate the Anaconda environment by typing 'conda activate ./env' (see below):



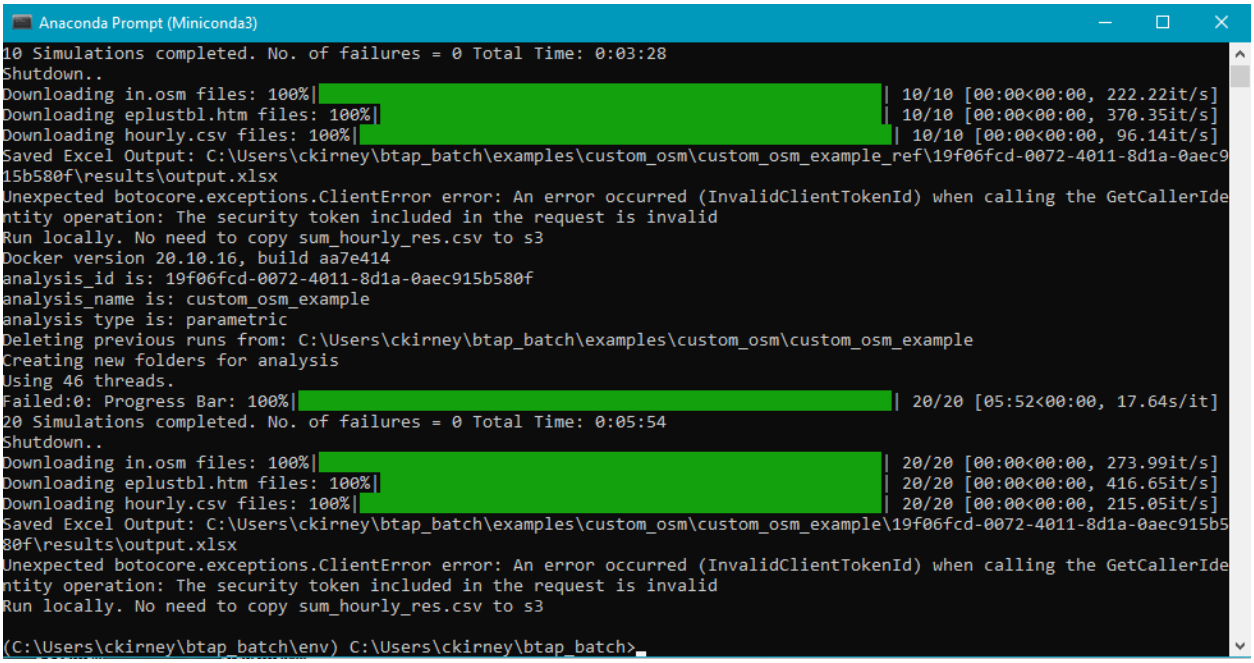
12. We will now start our btap_batch run by typing 'set PYTHONPATH=%cd% && python examples\custom_osm\run.py' (see below):



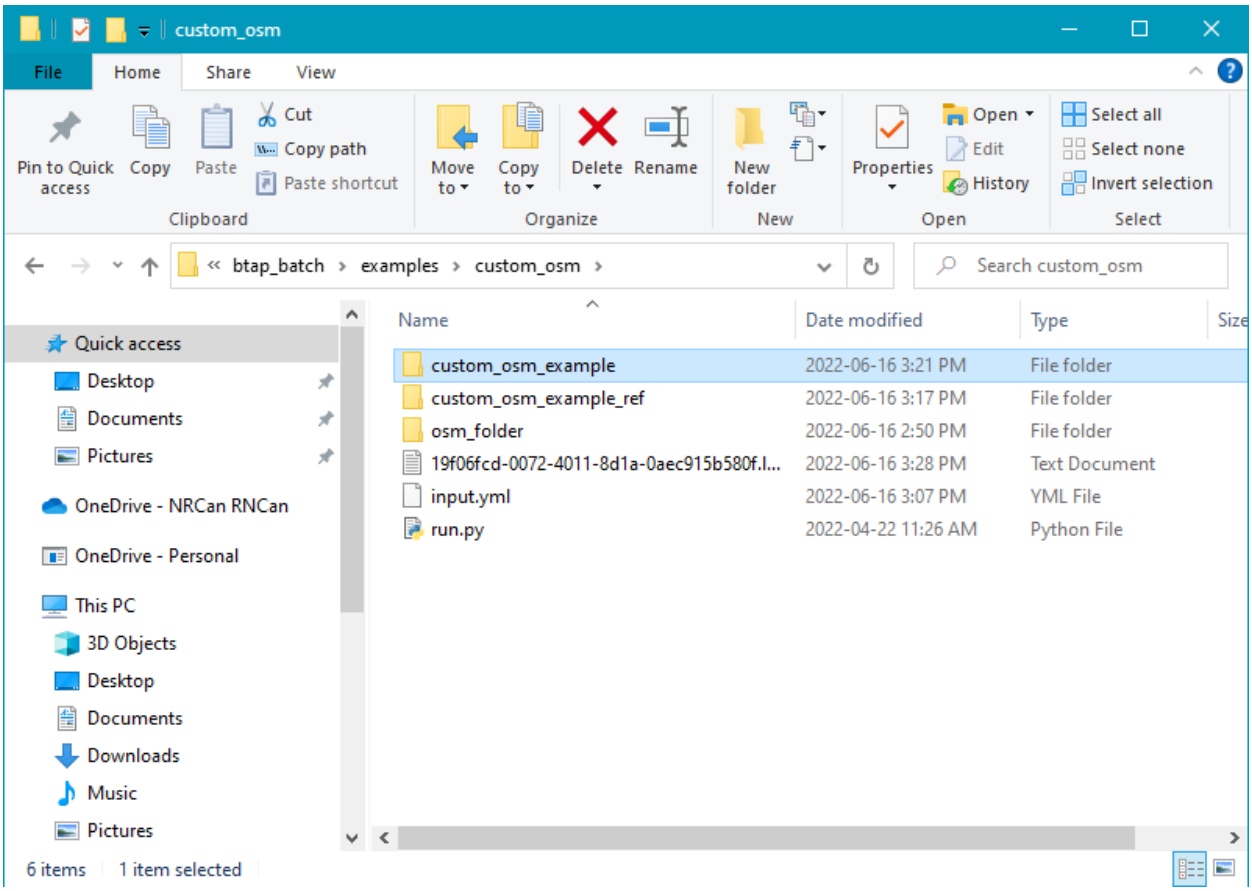
13. The analysis should start after a minute or two and you should see something like the following:



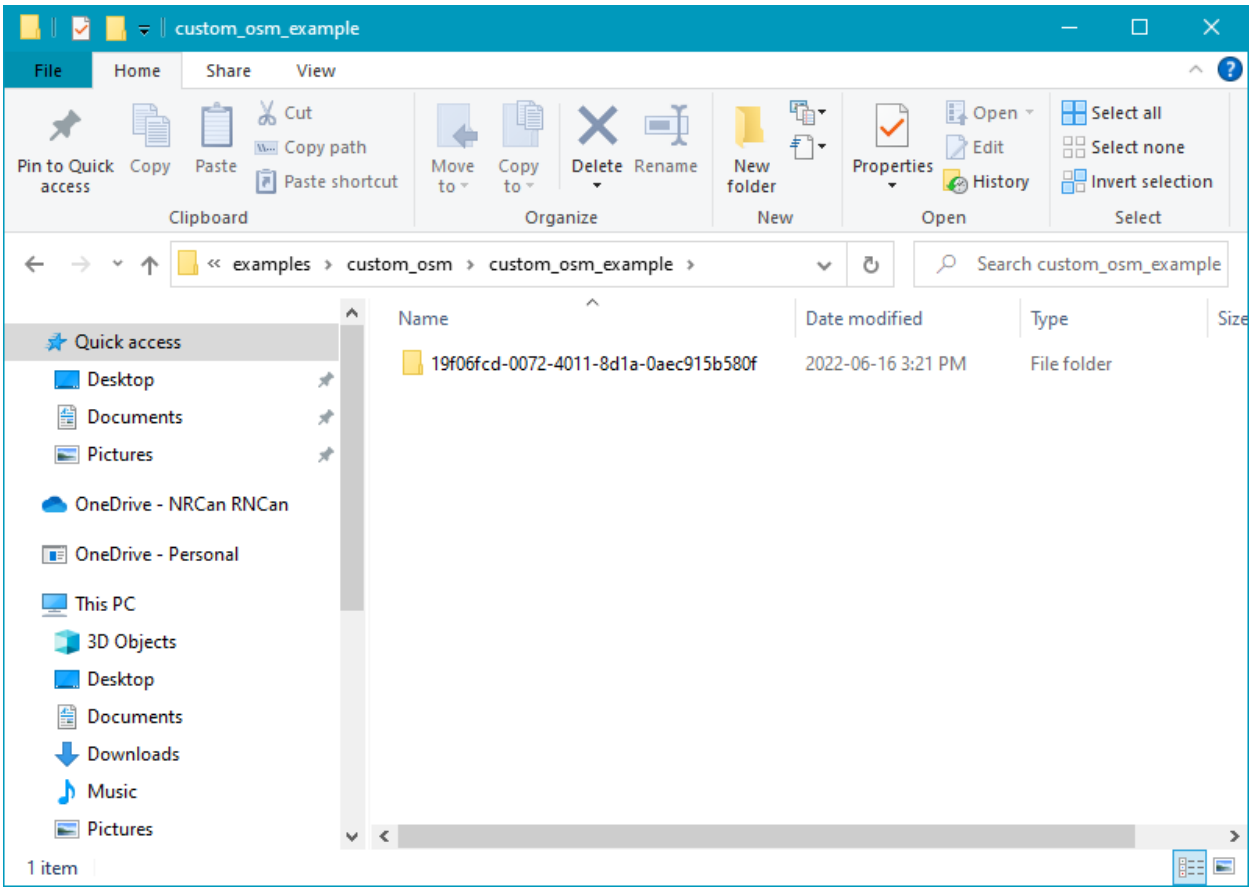
14. Wait until the analysis is complete. Assuming all went well, you should see something like the following:



15. Congratulations! Your simulations are complete (and hopefully successful). Close the 'Anaconda Prompt (Miniconda3) and open the Windows File Explorer and navigate to your btap_batch folder and then into 'btap_batch/examples/custom_osm'. There should be a 'custom_osm_example' folder and a 'custom_osm_example_ref' folder (see below):



16. The ‘custom_osm_example_ref’ folder contains the results of a reference analysis. This is essentially the same as the analysis we did except that all building options (other than building type, template, weather, and primary heating fuel) are defaulted. Open the ‘custom_osm_example’ folder and you should see another folder with a long string of numbers and characters (see below, note that your folder should have a different string of numbers and characters than what is in the screenshot):



17. This is the folder that contains the analysis you ran. The long string of numbers and characters (called a hash from now on) is what btap_batch has named your analysis. Open that folder and you should find an 'input' folder, an 'output' folder, and a 'results' folder.

    The 'input' folder should contain a number of folders with hash names. Each of these folders represents a specific simulation and contain the OpenStudio file used in the simulation and a 'run_optinos.yml' file that contain the specific btap_batch characteristics of that particular simulation.

    As with 'input', the output folder contains several folders with hash names. As with 'input', each folder represents a simulation. In this case the folders contain the outputs from the simulation including the OpenStudio and EnergyPlus results and some output files created by BTAP. Two in particular are 'btap_data.json' and 'qaqc.json'. Both files contain summarized results for the simulation. They both contain similar information. However, 'qaqc.json' has more information but is presented in a way that is more difficult to input into data analysis software or scripts.

    The results folder contain 'database', 'eplustbl.htm', 'failures', 'hourly.csv', and 'in.osm' folders and an output.xlsx file.

The ‘database’ folder contains ‘.csv’ files named with the hash of the simulation they refer too. Each ‘.csv’ contains summarized results for that simulation. The information in all ‘.csv’ files are collected together and saved in the ‘output.xlsx’ file.

The ‘eplustbl.htm’ file contains the EnergyPlus results ‘.htm’ report for each simulation (they are named after their corresponding simulation).

The ‘failures’ folder contains any error outputs from failed simulations. It should be empty.

If you chose to produce hourly results for some output variables in the ‘input.yml’, those results are stored in the ‘hourly.csv’ folder. In this exercise we did, so the folder should contain many ‘.csv’ files named after their corresponding simulation and containing the simulation’s hourly results.

The ‘in.osm’ folder contains the OpenStudio models run for the analysis. Each ‘.osm’ file is named after their corresponding simulation.

The output.xlsx file contains the summarized inputs and outputs for the analysis.