LARSON—INFO 790–CLASSROOM WORKSHEET 05
Using CONJECTURING on Athena

1. Log in to VCU's Athena cluster.

   The following directions assume you have an Athena account, that you have set up Sage, and that you have set up (using `make`) the CONJECTURING program.

   (a) Start the Chrome browser.

   (b) If you are off-campus, you'll need to connect to the VPN first.

   (c) Then go to `https://athena3.hprc.vcu.edu`

   (d) Login using your VCU EID as your username, and your corresponding VCU password.

   (e) Click the Apps button and a `Sage` session. The default options are fine. This will take a couple of minutes.

   (f) Click the Apps button and start an "athena shell access" session (this will give you a terminal window, where we can issue commands).

   (g) Your Sage session will first say "Queued", then "Starting". When it is ready you will see a button that says, "Connect to Sage". Click that.

   (h) You should then get an "untitled" interactive-Python notebook (ipynb), or the last file you had open the previous time you used Athena.

   (i) When your notebook opens look on the upper-right to make sure the SageMath kernel is running (if it isn't you can change the *kernel*).

   In each case, for each experiment, we will make a folder in your root directory; we will need a copy of the "expressions" compiled executable in that folder; and we will use an .ipynb located in that notebook. When we call the CONJECTURING program we will use the version in the ∼/`conjecturing` folder downloaded from github (what you did with the github command; if there are ever new files on github, using the command `git pull` will update your files).

   Real Estate!

2. In the terminal window, make a folder called `Real_estate` in your Home directory (use `mkdir Real_estate`).

3. Copy the *expressions* executable to your Real_estate folder:
   `cp ∼/conjecturing/c/build/expressions Real_estate/`

4. Copy the 3 files we'll need (plus other things) into this folder with:
   `cp ∼/conjecturing/examples/Learning-from-Data-2020/* Real_estate/`

5. Then "cd" (change directory) to that directory: `cd Real_estate`

6. Enter `ls` (the "list" directory command) to see what's there. You should see some "real estate" python files and some "csv" data files.

7. Go back to your Sage notebook session. Enter and evaluate `pwd` to see what directory you are currently pointed at. If it is not your Real_estate directory, enter:
   `cd Real_estate`.

8. You should rename your .ipynb notebook to **790-c05** (so it will be "790-c05.ipynb", and will remain on the Athena server for your future use/access/reference).

9. We will always run this command to run the latest version of the CONJECTURING program: `~/conjecturing/sage/conjecturing.py`. If you ever run a script and get an error that the `conjecturing.py` file is missing, change the line calling that program to `~/conjecturing/sage/conjecturing.py`.

10. Now lets try to run the existing real estate investigation script: `load("real_estate_python3.py")`.

11. If this command crashes with the error that there is no "conjecturing.py" file, you'll need to change this line to: `~/conjecturing/sage/conjecturing.py`.

12. You should be running a script (you should see various outputs appearing in your .ipynb file and, if it is still running, an hour-glass tab in your browser).

13. After the script is run, you will see various "property conjectures" (of both sufficient-condition and necessary condition types).

14. We'll want to better understand these—in order for you to do your own future investigations. The script may take a while. While it is running, double-click on `real_estate_python3.py`.

    Among other things, the conjecture-objects have names, the house-objects have names, and we'll want to know how to check a conjecture against house objects, for instance, to evaluate a conjecture against testing data:

    (a) The CONJECTURING program needs "objects" as inputs. What are these called?

    (b) The conjectures are given names in the script. Where are these? What are they called?

    (c) How can you give a name to a single "object" for your own testing purposes?

    (d) How can you give a name to a single "conjecture" for your own testing purposes?

15. Now lets test how the one produced "above" conjecture works for the "test" data.

    (a) Read in the test data: `test_data_df = pd.read_csv("testData.csv")`

    (b) Convert this raw data into *House* objects:
    `test_houses=[House(test_data_df, i) for i in range(len(test_data_df))]`

    (c) How many "test" houses are there? Run: `len(test_houses)`

    (d) Call the one produced "above" conjecture "conj0". Run: `conj0=above_conjs[0]`.

    (e) Let's see how good the conjecture is on the reserved data. Run:

```
count=0
for house in test_houses:
    if conj0.evaluate(house):
        count = count + 1
print(count)
```