ESC190 LAB 0

Trees, Arbres, 木, Árbol

Due: January 19, 2020 23:59

BACKGROUND

The purpose of this lab is to reboot you into the semester, continuing from ESC180. You will be working with datasets from Statistics Canada[1] and the US Census[2] on mother tongues of the population. You will build self-balancing binary search trees to store this data.

You are given 3 starter code files:

- lab0_utilities.py contains the LanguageStat and Node classes. DO NOT modify this file.
- lab0.py contains skeleton code for you to complete. You should complete all the indicated class methods marked by #implement. DO NOT add any code outside of the given classes. You may create additional class method definitions if you find it helpful.
- tester_lab0.py contains some functions to help you test your code. This is NOT a complete tester. You should add any code you use to test here.

The only file that will be marked is lab0.py, and the marking will be done with the original version of lab0_utilities.py.

ACCESSING THE STARTER CODE

You will be tracking and submitting your files using Git, much like in ESC180. Ensure you follow these steps **very, very carefully**. Setting this up is a one-time process, and you will be working only with the essential git commands from then.

1. Make sure you read the entire instruction first before running the commands. Do not blindly follow along. The Lab TAs are available to answer your questions throughout the lab session.
2. Login to the ECF machine.
3. Open Terminal (Applications -> System Tools -> Terminal)
4. Navigate to your SSH folder by running the command

   cd ~/.ssh

5. Run the command ***ls*** to see the directory's contents.
6. If you see a file called "**<utorid>.conf**" (for example, a student with utorid "abc" would have "abc.conf"), change the name of the file to **config** by running the command

   mv <utorid>.conf config

   If you do not see the <utorid>.conf file, but see the file named config, DO NOT change anything else in the directory! If you do not see both files, contact one of the lab TAs as you will not be able to continue working on the lab.

[1] https://open.canada.ca/data/en/dataset/ef9cf970-6b27-4d14-9106-7e9ca691c2c1

[2] https://www.census.gov/topics/population/language-use/data.html

7. Run the ***ls*** command again and ensure you have the two files named **config** and **<utorid>**. You may ignore the other files for now. If you do not have these 2 files, contact the lab TAs.
8. Create a directory anywhere convenient (e.g. under Desktop/ or Documents/) named "ESC190_Labs" (or any name you want) by running

   mkdir ~/Desktop/ESC190_Labs

9. Navigate to this directory by running

   cd ~/Desktop/ESC190_Labs

10. Run

    git clone ssh://gitolite/<utorid>_esc190

    When it asks for your password, type your student number. This will allow you to access the assignment submission platform. For example, a student with the utorid **abc** would run the command **git clone ssh://gitolite/abc_esc190**.

11. Run

    cd <utorid>_esc190/lab0

    You will find the starter code in this directory. **DO NOT** rename or move the files anywhere else. It **MUST** live in the location with the same name that you cloned from git.

12. In order to enable Python 3.6.9, do **one** of the following in your ECF account:

    A. Navigate to Applications -> ECF -> Software Environment Configuration and check the "ESC190" checkbox.

    B. Type *ecf-sw-env -a ESC190* in the terminal.

    Both steps will enable the devtoolset-8 and rh-python36 software collections on your ECF accounts which include all of the software versions listed on the syllabus. Please note that these changes will remain on your accounts until you uncheck the checkbox from A, so please remember to do so after the course is over.

TASKS

You do not need to write docstrings for this lab. You are permitted to modify any of the provided methods (except those in lab0_utilities.py) as you see fit, but **do not** change the parameters of the methods.

CLASS LANGUAGES

You are required to complete the following methods:

```
build_trees_from_file(self, file_object)
```

[1] https://open.canada.ca/data/en/dataset/ef9cf970-6b27-4d14-9106-7e9ca691c2c1
[2] https://www.census.gov/topics/population/language-use/data.html

Takes as input a file object which will have the same structure as ca_languages.csv and updates the dictionary attribute `Languages.data_by_year`. `Languages.data_by_year` is a dictionary with keys which are the year of the language statistic (type `int`), and the values are balanced (AVL) binary search trees (instances of class `BalancingTree`). Return the `data_by_year` dictionary. This method should only modify the dictionary values using methods from the `BalancingTree` class. Tree insertions are based on the attribute node._val as described in the Node class under lab0_utilities.py. Note: insert elements into the tree in the order that you read the data files from top to bottom. For example, in ca_languages.csv, for the key 1931 insert in the following order: English, French, Lithuanian, etc.

Return type: `{int: BalancingTree}`

`query_by_name(self, language_name)`

`language_name` is a string representing a language name. Return a dictionary with keys that are the year, and values which are the count for the language `language_name` based on the current `data_by_year` dictionary. If `language_name` is not found in the `data_by_year` dictionary, return an empty dictionary.

Return type: `{int: int}`

`query_by_count(self, threshold)`

Return a new dictionary with the keys being the year, and the values being a list of language names which have a count which exceeds the type `int` threshold.

Return type: `{int: List[str]}`

CLASS BALANCINGTREE

You are given the code for a simple insert to the binary tree and one of the balancing rotations. You may assume that the inserted node has no left or right children. You are to implement the following methods:

`balance_tree(self, node)`

Modify the binary search tree such that it is balanced. Use node (which is the inserted node, `class Node`) as a starting point to calculate balance factors.

`right_rotate(self, z)`

Perform a right rotation around node z. You may use the method left_rotate for reference.

`find_balance_factor(self, node)`

Returns the balance factor of the indicated node.

`is_balanced(self)`

Return a boolean value: `True` if the tree is balance, otherwise, `False`.

[1] https://open.canada.ca/data/en/dataset/ef9cf970-6b27-4d14-9106-7e9ca691c2c1

[2] https://www.census.gov/topics/population/language-use/data.html

SUBMISSION INSTRUCTIONS

The same instructions from ESC180 can be followed for submitting your code. However, we will provide you with the instructions for this lab to refresh your memory. You will not find this section in future labs. As programmers you are expected to become comfortable with git commands, and you may reference the git manual for detailed instructions.

1. You must first tell Git to track (or stage) a file when you create a new file, make changes to it or delete it. This is done with **git add <filename1>.** If you have multiple files, use **git add <file1> <file2>**.
2. Once you have made significant progress in your labs, you can tell Git to save the current state in the **local machine** (the ECF computer) by running **git commit –m "<commit message>"**. Note that you must provide a commit message.
3. Finally, once you have committed for the last time, you can push your code to the remote repository by running **git push origin master** or simply **git push**.

Some points to keep in mind:

1. You **must** stage a file before being able to commit it. You **must** commit the file before being able to push it to the remote repository.
2. You can commit as many times as you'd like. We encourage you to commit and commit often, as it would make it easier for you and the lab TAs if you need to revert to a previous state if something goes wrong.
3. We **will not** receive your files unless you do a **git push.**
4. Running **git log** will show you a history of your commits with your commit messages.
5. Running **git status** will show you the current status of your directory. It may include information about which files are not tracked, which files are tracked but not committed, and which files are ready to be pushed. Ensure you stage, commit and push only the relevant files.
6. The git manual has detailed instructions on the most used git commands and the only commands you will need to use throughout the labs for ESC190.

[1] https://open.canada.ca/data/en/dataset/ef9cf970-6b27-4d14-9106-7e9ca691c2c1

[2] https://www.census.gov/topics/population/language-use/data.html