CMSC 420 Fall 2024: Coding Project 3
Splay Trees (Variation 1)

1 Due Date and Time

Due to Gradescope by Sunday 20 October at 11:59pm. You can submit as many times as you wish before that.

2 Get Your Hands Dirty!

This document is intentionally brief and much of what is written here will be more clear once you start looking at the provided files and submitting.

3 Assignment

We have provided the template `splay.py` which you will need to complete. More specifically you will fill in the code details to manage insertion, deletion, and search for a Splay Forest, meaning a collection of Splay Trees.

We have implemented a `SplayForest` class as well as a `Node` class. The `SplayForest` class stores only an object `roots` whose keys are the names of Splay Trees and whose values are pointers to the roots of those Splay Trees.

Note that the `Node` class includes a parent pointer which you will need to maintain and the `dump` method prints out the parent key (except for the root).

Please look at this file as soon as possible.

4 Details

The functions should do the following:

- `def newtree(self,treename):`

  This is done for you. This adds an entry in `self.roots` whose key is `treename` and whose value is `None`.

- `def insert(self,treename:str,key:int):`

  Insert the `key` into the tree named `treename` using the first method discussed in class. The key is guaranteed not to be in the tree.

- `def delete(self,treename:str,key:int):`

  Delete the `key` from the tree named `treename` using the first method discussed in class and using the right subtree if neither subtree is empty. The key is guaranteed to be in the tree.

- `def search(self,treename:str,key:int):`

  Search for the `key` in the tree named `treename` using the method discussed in class. This should splay the tree but does not return or print anything. The key may or may not be in the tree.

5 Additional Functions

You will probably want some helper functions as well as `SplayForest` class methods. Up to you.

6 What to Submit

You should only submit your completed `splay.py` code to Gradescope for grading. We suggest that you begin by uploading it as-is (it will run!), before you make any changes, just to see how the autograder works and what the tests look like. Please submit this file as soon as possible.

7 Testing

This is tested via the construction and processing of tracefiles.

- Each non-final line in a tracefile is either `insert,treename,key` or `delete,treename,key`, or `search,treename,key`. All together these lines result in the creation of a Splay Forest.
- The final line will always be `dump`.

You can see some examples by submitting the `splay.py` file as-is.

8 Local Testing

We have provided the testing file `test_splay.py` which you can use to test your code locally. Simply put the lines from a tracefile (either from the autograder or just make one up) into a file `whatever` and then run:

```
python3 test_splay.py -tf whatever
```