Homework 2

Due 11:59 PM on September 20, 2018

There are two parts to this assignment.

Part 1: Implementing a Bloom filter

Please complete the `add_elem` and `check_membership` methods in the `bloom_filter.py` file.

Important: Your Bloom filter implementation must use the three hash functions that we have provided (`cs5112_hash1`, `cs5112_hash2`, and `cs5112_hash3`).

Do not use Python's built-in hash function! You must use ours.

The comments in `bloom_filter.py` say more about what we expect for inputs and outputs.

Grading Note: In addition to testing the output of your functions, we will also be inspecting how you're storing data in the underlying array. Therefore, be sure to implement the algorithms to spec.

Part 2: Implementing a hash table, two different ways

First, implement a hash table using linear probing. To do so, complete the specified methods in the `hashtable_linear_probing.py` file. Please read the comments in the file carefully to understand our expectations for inputs and outputs. You ***must*** use linear probing to handle collisions in this implementation. Note that you also need to resize the underlying array when appropriate to ensure that there is always room to insert a new mapping. An array data structure is provided, as is a hashing function.

Do not use Python's built-in hash function! You must use ours (`cs5112_hash1`).

You must use the provided `Fixed_Size_Array` type to implement your hash table.

Grading Note: In addition to testing the output of your functions, we will also be inspecting how you're storing data in the underlying array. Therefore, be sure to implement the algorithms to spec. For your linear probing hash table, you must make sure to insert elements at the next available cell (looping around to the beginning of the array if need be).

Second, implement a hash table using chaining (with a linked list). To do so, complete the specified methods in the `hashtable_chaining.py` file. Please read the comments in the file carefully to understand our expectations for inputs and outputs. You ***must*** use chaining with a linked list to handle collisions in this implementation. You will be expected to implement array resizing for this hash table as well. The linked list data structure is provided to you, as is an array data structure and a hashing function.

Do not use Python's built-in hash function! You must use ours (`cs5112_hash1`).

You must use the provided `Fixed_Size_Array` type to implement your hash table.

You must use the provided linked list node data structure to implement chaining.

Grading Note: In addition to testing the output of your functions, we will also be inspecting how you're storing data in the underlying array/linked-list. Therefore, be sure to implement the algorithms to spec. For your chaining hash table, you must make sure to insert elements at the *END* of any given linked-list chain.

GENERAL NOTE: Please do not import any additional libraries into your code for this assignment. This applies to ALL problems in the assignment.

Logistics

You may use Python 2.7 or Python 3 to do this assignment. ***But please note***: if you want to use Python 3, please add a `#!PYTHON3` comment somewhere in one of your submitted files. If you aren't sure which you are using, typing `python` on the command line will print out the version you have.

Please modify and submit the following files:

- `bloom_filter.py`
- `hashtable_linear_probing.py`
- `hashtable_chaining.py`

To ensure compatibility with our grading software, please ensure that the provided test files run. You should be able to run the following without errors:

```
python bloom_filter_test.py

python hashtable_test.py
```