Assignment 7: The Great Firewall of Santa Cruz

Leo DeAnda

November 28, 2021

1 Brief Synopsis

This lab will be using a bloom filter to take in a message and filter out and replace some specified words. There will be hash tables, bit vectors, and binary search trees used as well.

2 bf

This file will contain functions for the Bloom Filter.
- bf_create: This function acts as the constructor for the bloom filter. Also, implementation of a bit vector and the salts from salts.h will be used as well.
- bf_delete: The allocated memory from the constructor is freed and the pointer is set to null.
- bf_size: Will return the number of bits that the bloom filter has access too. AKA the size of the bloom filter(bit vector length).
- bf_insert: This wil insert the given oldspeak into the Bloom Filter. Will need to hash the oldspeak with each of the three salts from the indices.
- bf_probe: Will hash the oldspeak with each of the three salts for three indices. If all the bits are set true will be returned and false will be returned otherwise.
- bf_count: This will return the number of bits within the Bloom Filter.
- bf_print: This is debugging function that will just print out the Bloom Filter.

3 bv

This file will contain functions for the bit vector.
- bv_create: This is the constructor for the bit vector and it will be as long as the length that is given.
- bv_delete: The destructor that will free the allocated memory and set the pointer equal to NULL afterwards.
- bv_length: This will return the length of the bit vector.
- bv_set_bit: This will set the 'ith' bit in the bit vector. If the given 'i' is out of range, false will be returned. Otherwise true will be returned.
- bv_clr_bit: This will clear the 'ith' bit in the bit vector. If the given 'i' is out

of range, false will be returned. Otherwise true will be returned.
- bv_get_bit: This will return the 'ith' bit in the bit vector. If the given 'i' is out of range, false will be returned. Otherwise true will be returned.
- bv_print: This is a debug function that will print out the bits of the bit vector to determine if the bit vector was properly created and the functions work properly.

4 ht

This file will contain the functions for the hash table.
- ht_create: This will be the constructor for the hash table. The number of indices will be given by 'size' that will be passed in.
- ht_delete: This is the destructor for the hash table. The pointer will then be set to NULL afterwards.
- ht_size: This will return the size of the hash table.
- ht_lookup: This will go through and search for a node that has oldspeak. If the node can be found, it will be returned. Otherwise NULL will be returned.
- ht_insert: Inserts the oldspeak and its newspeak into the hash table. This will be done by hashing the oldspeak.
- ht_count: This will return the number of non-NULL binary search trees in the hash table.
- ht_avg_bst_size: This will return the average binary search tree size. bst_size function.
- ht_avg_bst_height: This will return the average binary tree search size. Using the bst_height function.
- ht_print: This will be the debug function that will print out the hash table to see if it was created properly.

5 node

This file will contain the functions for the nodes.
- node_create: Constructor for the node. Will require a copy of oldspeak and newspeak translations that are given.
- node_delete: This is the destructor for the node and will free the allocated memory. Only the node will be freed and the next nodes will not be deleted. Also setting the pointer to the node to NULL.
- node_print: This will be a debug function to produce the correct program output. The printing is taken from the assignment document which has examples on how to print.

6 bst

This file will contain the functions for binary search trees.
- bst_create: This will construct the binary search tree which will be set to

NULL (it is empty).
- bst_delete: This will use postorder traversal and delete each node of the tree. Essentially freeing the allocated memory.
- bst_height: This will return a uint32_t which will be the height of the binary search tree.
- bst_size: Also returns a uint32_t, but it will be the size of the binary search tree. The size will be the number of nodes that are in the tree.
- bst_find: This will search for the node that has the given oldspeak. If the node with the oldspeak can be found, the node will be returned. Otherwise, a NULL pointer will be returned if the oldspeak node cannot be found.
- bst_insert: This will insert a node that has the given oldspeak and newspeak into the binary tree. If the node already exists, do not enter the node.
- bst_print: Debug function that will print out the binary tree with inorder traversal.

7 banhammer

This file will contain the 'main()' and have other functions necessary to the file. This is were the files will be read and determining which message to be printed to the user afterwards.