/// Andrew Souza

/// Comp 210 -- Spring 2024

/// Set, Maps, and Hash Tables Assignment

/// Problem 1

#include <iostream>

#include <unordered\_set>

#include <fstream>

using namespace std;

// 15.11.1

// Write a program that counts how often each word occurs in a text file.

// Use an unordered\_multiset<string>.

int main() {

string fileName;

ifstream inFS;

unordered\_multiset<string> wordCount;

cout << "Input your file name: ";

cin >> fileName;

inFS.open(fileName);

if (!inFS.fail()) {

string data;

while (!inFS.eof()) {

inFS >> data;

wordCount.insert(data);

}

}

inFS.close();

cout << "Word count: \n";

string last;

for (auto i = wordCount.begin(); i != wordCount.end(); i++) {

if (last != \*i) {

cout << \*i << " - " << wordCount.count(\*i) << "\n";

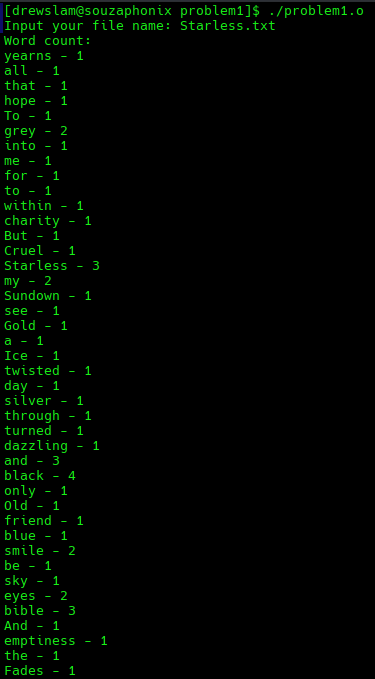
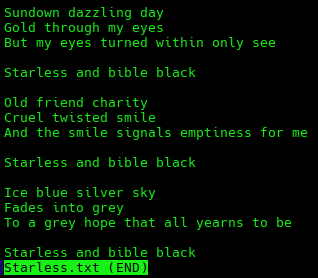
}

last = \*i;

}

return 0;

}



/// Problem 2

#include <iostream>

#include "HashTable.h"

using namespace std;

// 15.11.7

// Add a previous member function to the hash table iterator in Section 15.5.

void Iterator::previous() {

const Node \*ref = this->current;

while (this->getBI() > container->size()) {

// Iterating backwards from .end()

bucket\_index--; // Decrement index until a populated array index is reached

}

if (container->buckets[bucket\_index] != ref) {

// Iterating inside a bucket of multiple nodes

current = container->buckets[bucket\_index];

while (current->next != ref) {

current = current->next;

}

} else {

do {

bucket\_index--;

} while (container->buckets[bucket\_index] == nullptr);

current = container->buckets[bucket\_index];

while (current->next != nullptr) {

current = current->next;

}

}

}

int main() {

HashTable library(10);

string data;

cout << "Input some names, input -1 to exit: ";

getline(cin, data);

while (data != "-1") {

library.insert(data);

getline(cin, data);

}

auto iter = library.begin();

while (!iter.equals(library.end())) {

cout << iter.getBI() << " : " << iter.get() << "\n";

iter.next();

}

while (!iter.equals(library.begin())) {

iter.previous();

cout << iter.getBI() << " : " << iter.get() << "\n";

}

return 0;

}

