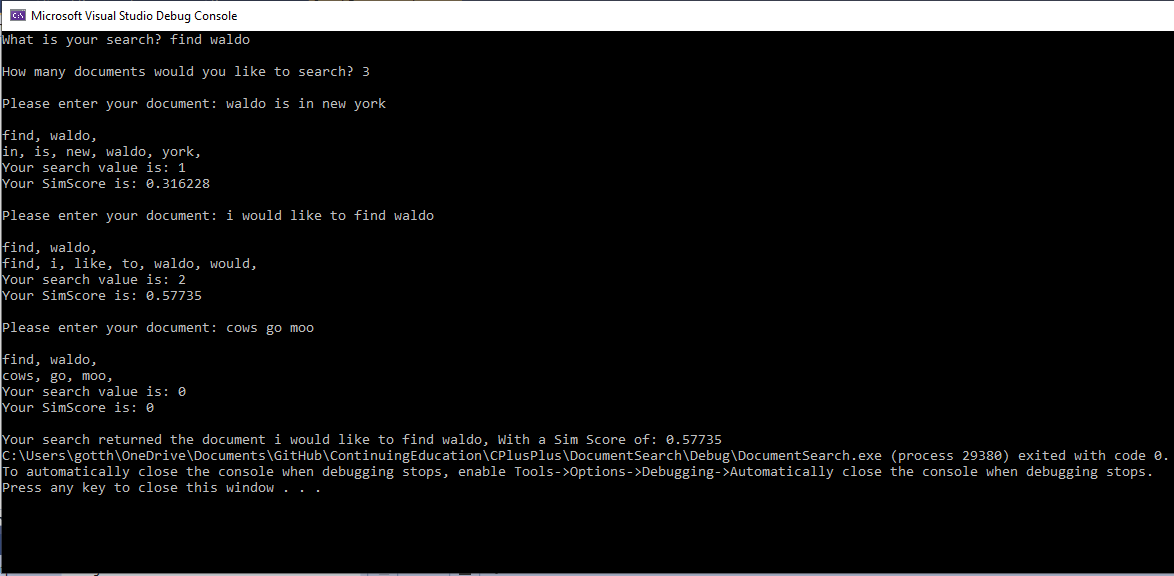
a. Your name – Joseph Rossitto

b. Id # - I don’t know how to find my Id #

c. Assignment # - Assignment 3

d. Due date – 10-14-20

e. Submitted date – 10-14-20



//a.File Name - Driver.cpp

//b.Author - Joseph Rossitto

//c.Date - 10-14-2020

//d.Compiler Used - Visual Studio

//e.Brief Description of the file - drives the program forward with simple selection menu

#include "SearchDocuments.h"

#include "Driver.h"

#include <iostream>

#include <string>

#include <algorithm>

#include <cctype>

#include <vector>

#include <numeric>

#include <iterator>

#include <list>

#include <set>

using namespace std;

int main()

{

int documentAmount;

string searchString;

string documentString;

double maxScore = 0;

double currentScore = 0;

string bestMatch;

cout << "What is your search? ";

getline(cin, searchString);

cout << endl;

cout << "How many documents would you like to search? ";

cin >> documentAmount;

cout << endl;

vector<string> documents;

cin.ignore();

//getline(cin, documentString);

for (int i = 0; i < documentAmount; i++)

{

cout << "Please enter your document: ";

getline(cin, documentString);

//while (documentString == "")

//{

// getline(cin, documentString);

//}

//cout << documentString << "\*\*";

cout << endl;

SearchDocuments firstDocument(searchString, documentString);

if (firstDocument.GetSimScore() > maxScore)

{

maxScore = firstDocument.GetSimScore();

bestMatch = documentString;

}

cout << endl;

}

cout << "Your search returned the document " << bestMatch << ", With a Sim Score of: " << maxScore;

//return notes();

//firstDocument.printWords(tempString);

//for (string word : document)

//{

// cout << word << endl;

//}

return 0;

}

int notes()

{

vector<int> v;

vector<string> s = { "hi", "my", "name", "is" };

//populate

for (int i = 0; i < 5; i++)

{

v.push\_back(i);

}

//total

int sum = 0;

for (int elem : v)

{

sum += elem;

}

//int total = std::accumulate(begin(s), end(s), 0);

int count3 = count(begin(v), end(v), 3);

cout << count3 << " Sum: " << sum << endl;

auto copyV = remove\_if(begin(v), end(v), [](int elem) {return (elem == 3); });

for (int i = 0; i < 5; i++)

{

v.push\_back(rand());

}

//Sort

sort(begin(v), end(v));

return 0;

}

//a.File Name - SearchDocuments.cpp

//b.Author - Joseph Rossitto

//c.Date - 10-14-2020

//d.Compiler Used - Visual Studio

//e.Brief Description of the file - main file for the SearchDocuments Class, handels reading in data, breaking it down into search phrases and calculating sim score

#include "SearchDocuments.h"

#include <iostream>

#include <string>

#include <algorithm>

#include <cctype>

#include <vector>

#include <numeric>

#include <iterator>

#include <list>

#include <set>

#include <sstream>

#include <cmath>

using namespace std;

SearchDocuments::SearchDocuments(std::string search, std::string document)

{

//string tempDocument;

//cin >> tempDocument;

if (document == "")

{

document = "Cows are big. Cows go moo. I love cows.";

}

addWords(search, this->search);

addWords(document, this->documentContents);

this->searchLength = this->search.size();

this->documentLength = this->documentContents.size();

ComputeSim();

//set<string> tempSet = tempString;

//vector<string> document = { "Cows are big. Cows go moo. I love cows." };

}

void SearchDocuments::printWords(string str)

{

// word variable to store word

string word;

// making a string stream

stringstream iss(str);

// Read and print each word.

while (iss >> word)

cout << word << endl;

}

void SearchDocuments::addWords(std::string newString, std::set<std::string>& whatToAddTo)

{

// word variable to store word

string word;

newString.erase(std::remove(newString.begin(), newString.end(), '.'), newString.end());

transform(newString.begin(), newString.end(), newString.begin(), ::tolower);

// making a string stream

stringstream iss(newString);

// Read and print each word.

while (iss >> word)

whatToAddTo.insert(word);

//cout << word << endl;

int count = 0;

for (string word : whatToAddTo)

{

count = count + 1;

cout << word << ", ";

if (count % 10 == 0)

{

cout << endl;

}

}

cout << endl;

}

double SearchDocuments::ComputeSim()

{

int searchMatch = 0;

int count = 0;

for (string word : this->search)

{

count = std::count\_if(begin(this->documentContents), end(this->documentContents), [&](string c) {return (c == word); });

searchMatch = searchMatch + count;

}

double sqrtOfSearch = sqrt(this->searchLength);

double sqrtOfDocument = sqrt(this->documentLength);

this->simScore = (searchMatch / (sqrtOfSearch \* sqrtOfDocument));

cout << "Your search value is: " << searchMatch << endl;

cout << "Your SimScore is: " << this->simScore << endl;

return this->simScore;

}

//a.File Name - SearchDocuments.h

//b.Author - Joseph Rossitto

//c.Date - 10-14-2020

//d.Compiler Used - Visual Studio

//e.Brief Description of the file - headerfile for the SearchDocuments Class

#pragma once

#include <string>

#include <set>

class SearchDocuments

{

private:

std::set<std::string> documentContents;

std::set<std::string> search;

int searchLength;

int documentLength;

double simScore;

public:

SearchDocuments(std::string search, std::string document);

void printWords(std::string newString);

void addWords(std::string newString, std::set<std::string>& whatToAddTo);

double ComputeSim();

auto GetDocument() { return documentContents; }

auto GetSimScore() { return simScore; }

};

What I learned:

I learned a lot about the C++ standard library and the algorithms available inside to make your life a bit easier. Containers are nice because they handle memory management for you. Using set was a good way to check for duplicates within your document. Count\_If made it easy to check if your search was in the document. Transform was a good way of dealing with mismatches of case, and erase gave me the ability to remove unwanted characters in the search.