Samuel Larson

Project 4

10/4/18

The objective of this project was to make a program that will read words in from a file and create a simple singly linked list that holds the word up to 8 letters and keeps track of how many times the word shows up in the document.

To accomplish this, I first need to make a statement that asks for the user to enter a file name. once the user has entered the files name the program will attempt to open the file and if the file doesn’t exist allow the user to enter a new file name.

Once the program has a working file it should call a function to read in the data from the file until it is empty. The function should scan the list for a match to the word that is read in from the file and increment a counter. If the word is not found the function should call another function to create a new node.

Once all the words and counts are added to the list the list should be printed to the screen by a function that scans the list and displays each word alongside with its number.

After the list is displayed to the screen another function should calculate the length of the list by bringing a pointer from the first node and counting until it gets to a node with a null pointer. The function will then display the length of the list as “unique words”.

The program will need the string.h header file to use the strcmpi() function which compares strings to each other without taking capitalization into account. The function also needs the fstream header file to read the information from the document.

//Samuel Larson

//program 3

//10/4/2018

#include <iostream>

#include <fstream>

#include <string.h>

using namespace std;

const int MAX = 8;

class list

{

public:

typedef char item[MAX+1];

typedef int num;

//constructors

list(){first = NULL;} //creates the start of the list

list(const list&);

//destructor - gives back a list's nodes

~list();

//modification member functions

void make\_empty(); //empties the list

void insert(item entry); //inserts a number to the list

//constant member functions

void print(); //prints the list

int length(); //prints the length

private:

//data members

struct node //structure of a node

{

item word;

num amount;

node\*next;

};

node\*first;

//private function

node\*get\_node(item entry, node\*link) //creates a node

{

node\*temp;

temp = new node;

strcpy(temp->word,entry);

temp->amount = 1;

temp->next = link;

return temp;

};

};

int main()

{

list wordlist;

string filename;

char word[MAX+1];

ifstream infile;

cout << "Please enter a file name" << endl;

cin >> filename;

infile.open(filename); //opens the file m1.dat to take in information

while(infile.good() == false)

{

cout << "file does not exist" << endl

<< "Please enter a file name" << endl;

cin >> filename;

infile.open(filename); //opens the file m1.dat to take in information

}

while(!infile.eof())//inserts more words while it isn't the end of the file

{

infile >> word;

wordlist.insert(word);

}

cout << "Word & Count" << endl;//start of the display of all words

wordlist.print();

cout << "There is " << wordlist.length() << " unique word(s)" << endl;

wordlist.~list();

return 0;

}

//post:list is deconstructed

list::~list()

{

node\*temp;

while(first != NULL)

{

temp = first;

first = first->next;

delete temp;

}

}

//post: list is emptied

void list::make\_empty()

{

node\*temp;

while(first != NULL)

{

temp = first;

first = first->next;

delete temp;

}

}

//pre:user enters a number to be added post: number is added to the list

void list::insert(item entry)

{

node\*p;

if(first == NULL)

first = get\_node(entry, first);

else

{

p = first;

//scans the list for a match or the end

while(p->next != NULL && strcmpi(entry,p->word) != 0)

{

p = p->next;

}

if(p->next == NULL)//checks if the next node exists

p->next = get\_node(entry, p->next);//creates a new node

if(strcmpi(entry,p->word) == 0)//checks if the words are the same

p->amount = p->amount + 1;//increments the counter of that word

}

}

//post:function prints the list

void list::print()

{

node\*p;

p = first;

while(p != NULL){

cout << p->word << " " << p->amount << endl;

p = p->next;

}

return;

}

//post:function returns the length of the list

int list::length()

{

node\*p;

int length = 0;

p = first;

while(p != NULL){

length++;

p = p->next;

}

return length;

}

Samuel Larson

Project 4

10/4/18

User Document

To use this program the program must be in the same directory as the file of text you would like to read. When you launch the program, it will prompt you to enter the file name and extension

Ex. Text.txt or wordlist.cpp

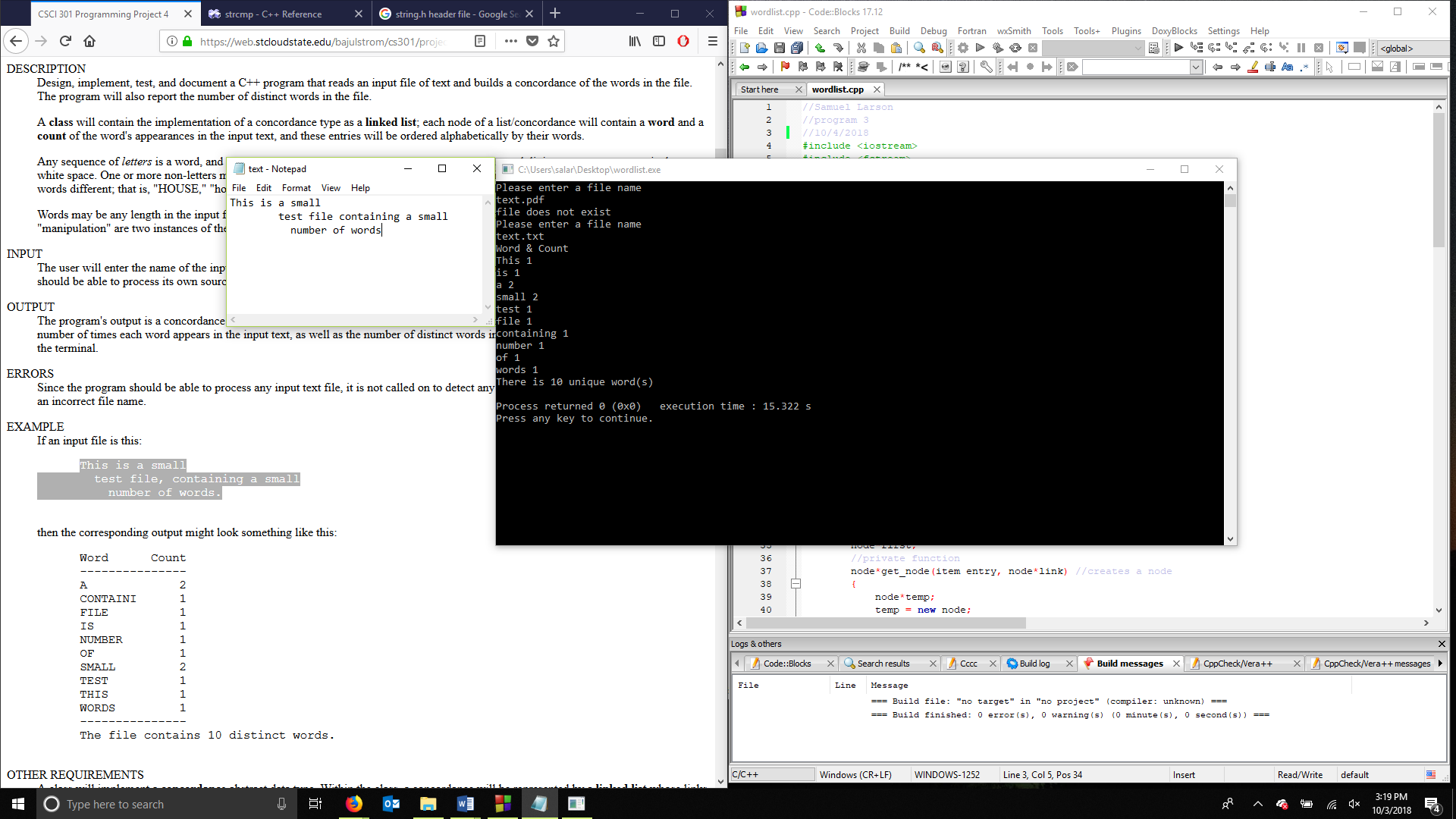
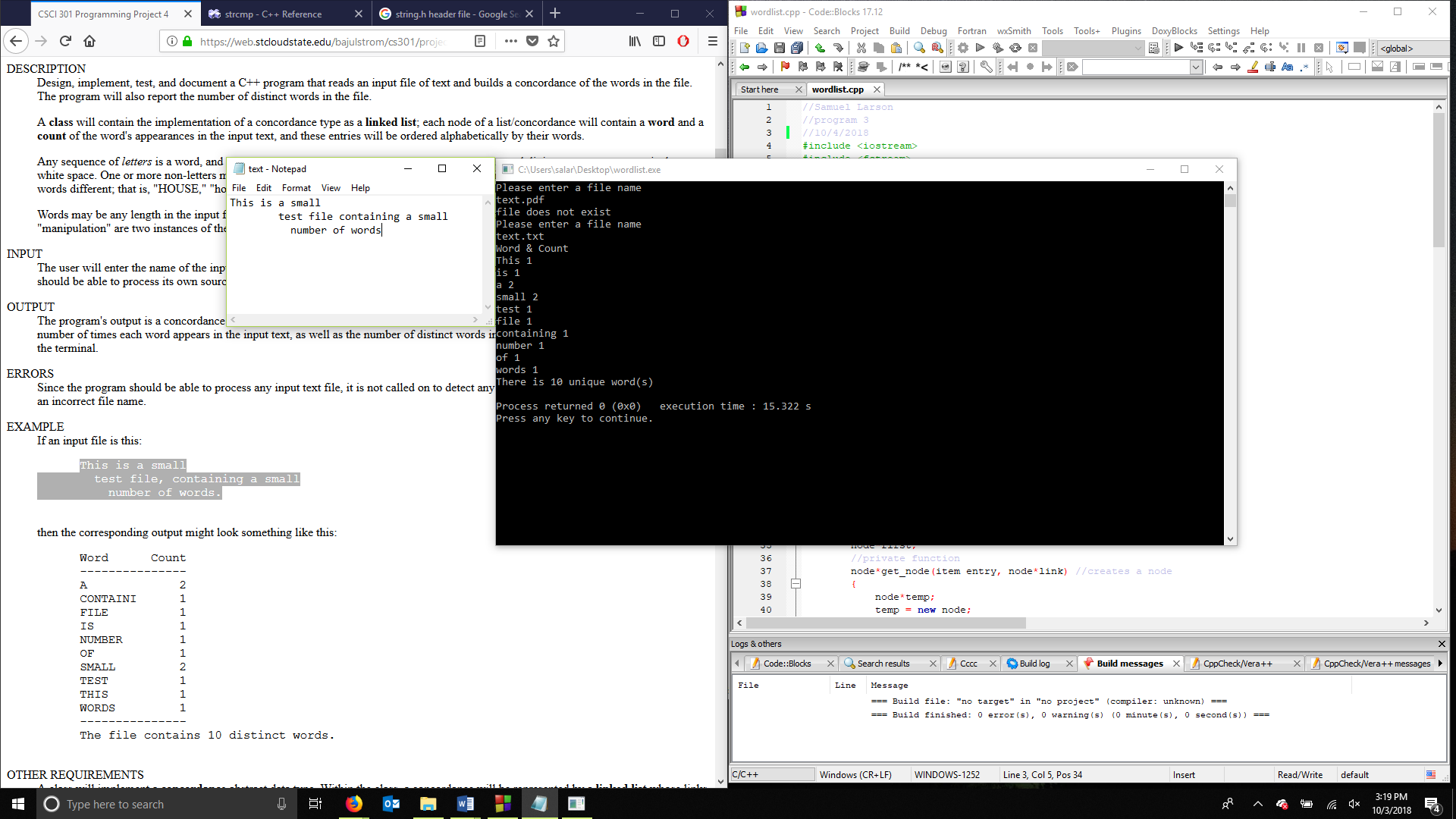
Hit enter and the program will proceed to track each unique word in the document and add them to a list with how many times they appear in the document.

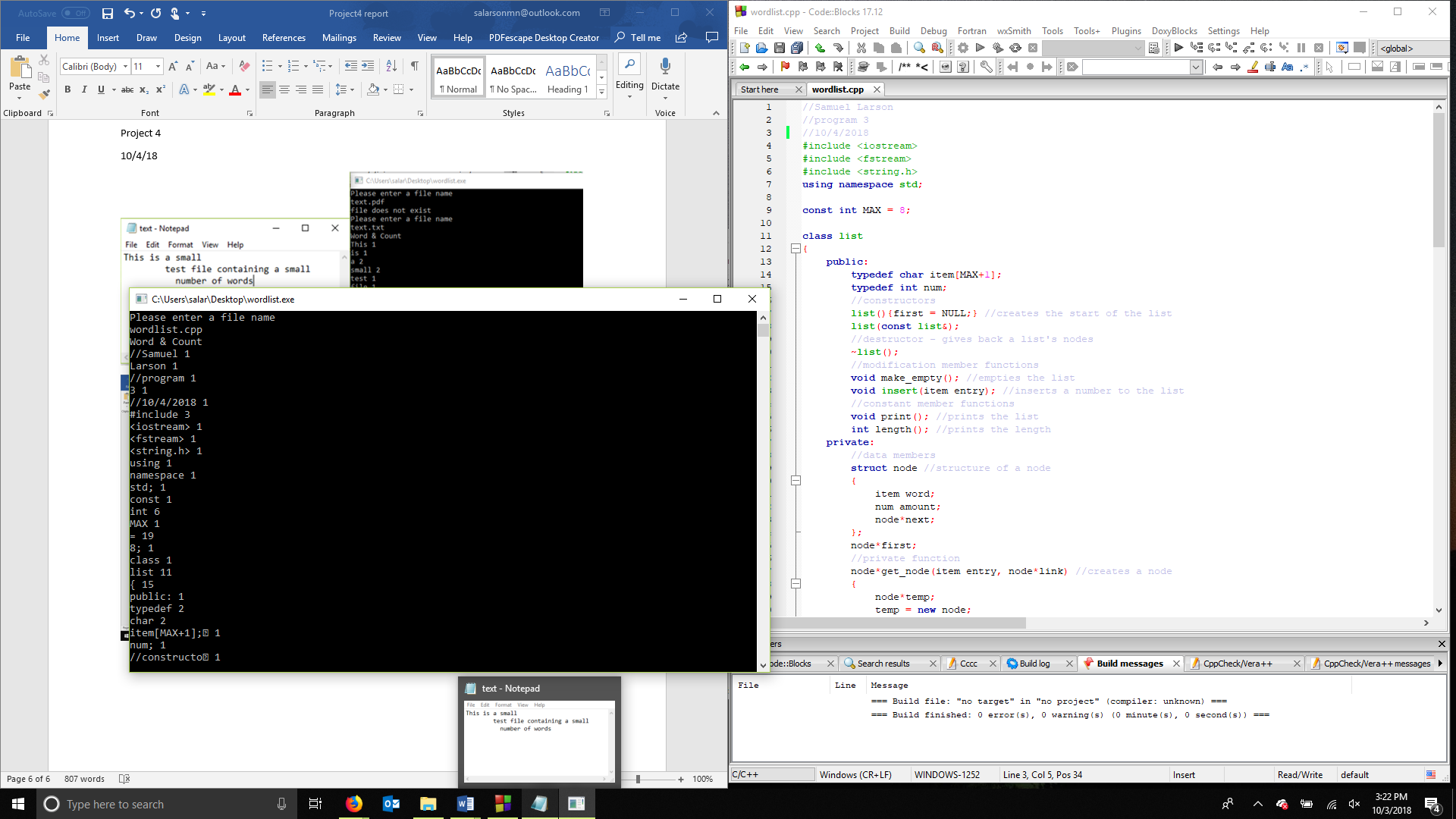
The program will print the resulting information to the screen.

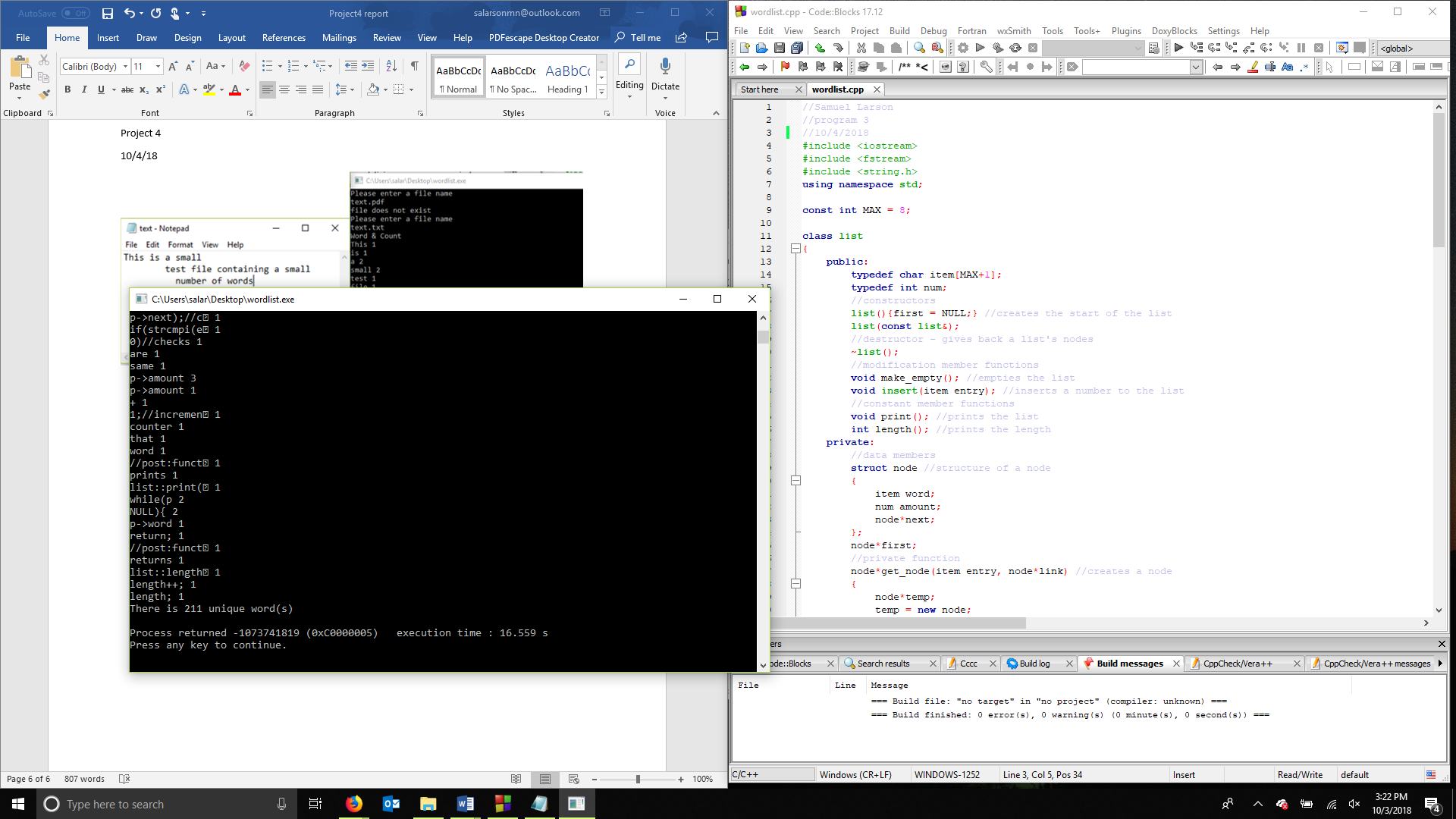
Samuel Larson

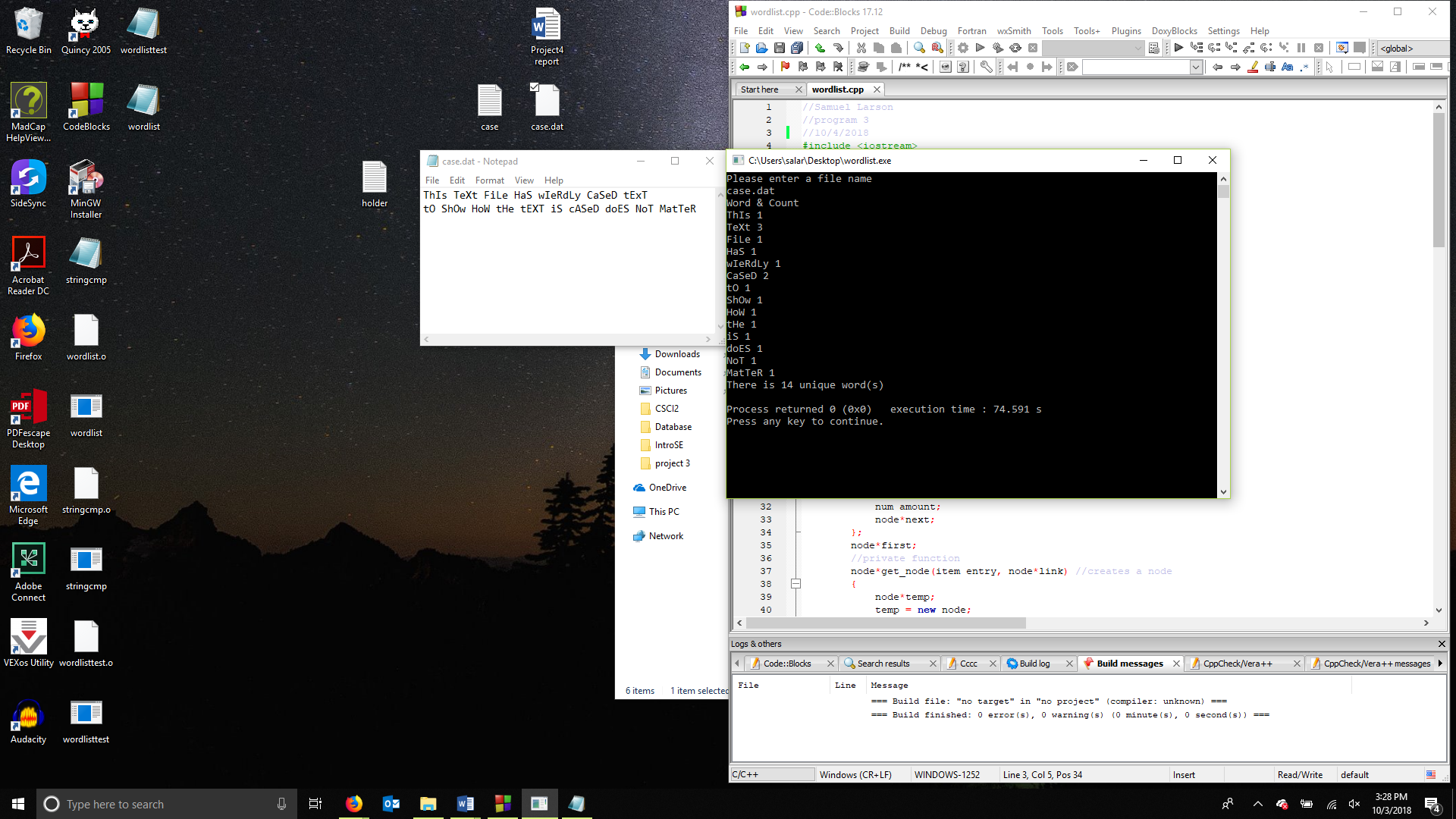
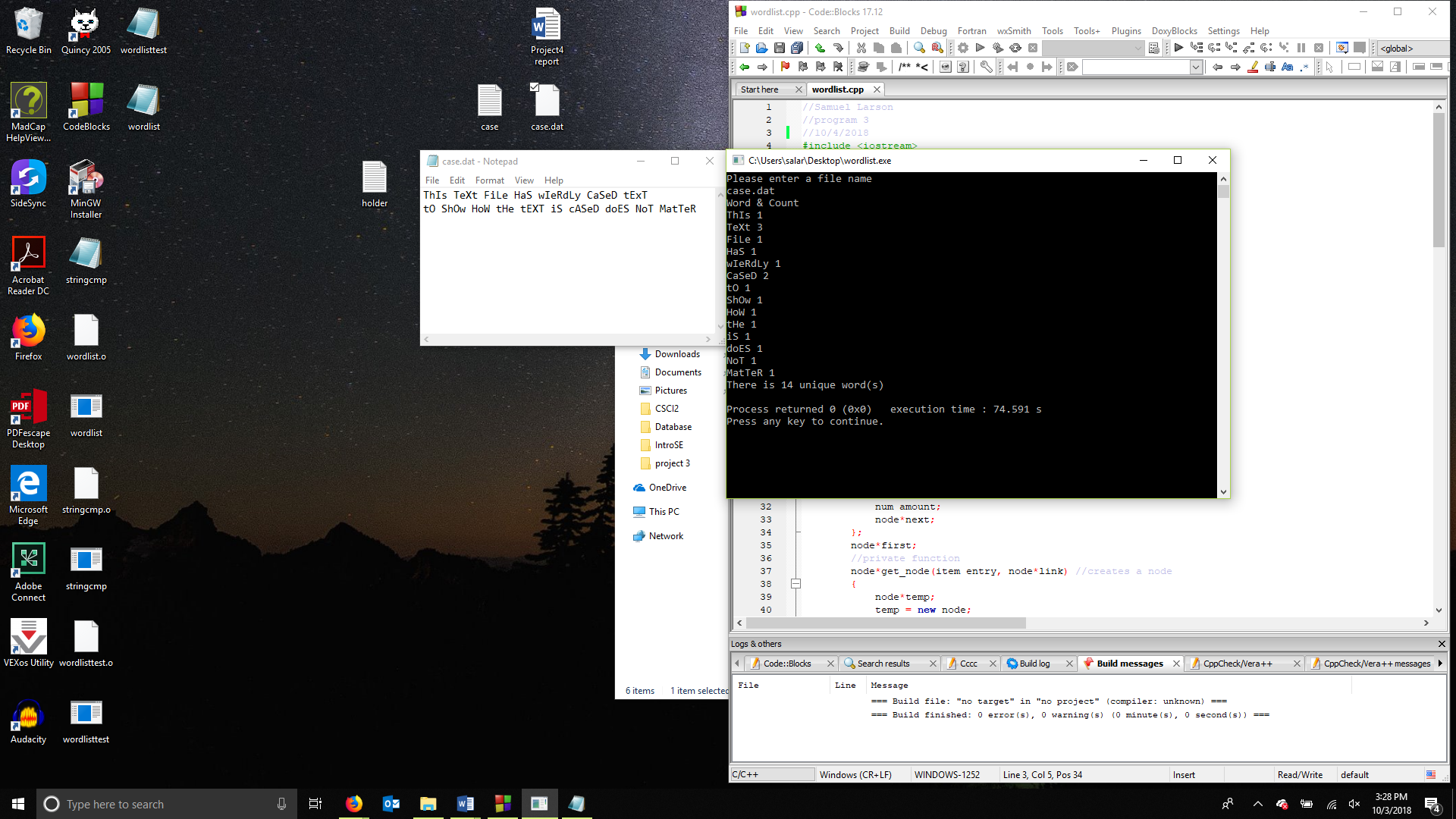
Project 4

10/4/18









Samuel Larson

Project 4

10/4/18

Summary

With this project I implemented linked lists to make a program that could read in words from a text file and add them to a list along with keeping track of how many times the word shows up in the file. The program would then print out the list and how many unique words there are.

My program could be made better by allowing the user to request the program to read in another file and give them the option to clear the list or add on to it with the next file. I used a destructor on the list at the end of the file, but I could have allowed the user to save what the program found to an output file. I could have better contained the function of reading in the file because how I did it the function was called once for each word and I could have passed the in file to the function and had it read each word in.