Samuel Larson  
Project 3

9/24/18

The objective of this project is to create a class that will allow the user to create a linked list that will be initialized empty, allow the user to empty an existing list, insert numbers into the list, remove numbers from the list, print the length of the list to the screen, search for a value in the list, find the kth value in the list, and write out all values of the list.

I will solve this problem by creating a class that consists of these functions

* A function that will create an empty list at the beginning of the program
* A function that will display a menu to give the user a list of commands they can use
* An insert function that will find the numbers larger and smaller than the number entered and place it in between
* A remove function that will delete a chosen number from the list and make the previous number point to the next
* A function that will print each number of the list out to the screen
* A function that will find the length of the list by walking a pointer down the list
* A function that will search for a number by walking a pointer down the list
* A number that will find what number is in a specific position of the list
* And a function that will create new nodes

When the program starts it should create the class and print the menu of commands for the user. The user will be able to enter a command and the program will use a switch statement to decide what function to use. Once the user is finished they will be able to enter a command to exit program.

//Samuel Larson

//program 3

//9/24/2018

#include <iostream>

#include <fstream>

using namespace std;

void printmenu();

class list

{

public:

typedef int item;

//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

void remove(item target); //removes a number from the list

//constant member functions

void print(); //prints the list

int length(); //prints the length

bool present(item target); //finds a number in the list

void kth(int k); //finds the kth number in the list

private:

//data members

struct node //structure of a node

{

item data;

node\*next;

};

node\*first;

//private function

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

{

node\*temp;

temp = new node;

temp->data = entry ;

temp->next = link;

return temp;

};

};

int main()

{

char funct;

int num;

list numbers;

printmenu();

while(funct != 'q')

{

cin >> funct;

//triggers a function based on what the user entered

switch(funct)

{

case 'e': numbers.make\_empty();

cout << "The list has been emptied" << endl;

break;

case 'i': cin >> num;

numbers.insert(num);

cout << num << " has been inserted into the list" << endl;

break;

case 'r': cin >> num;

numbers.remove(num);

break;

case 'l': cout << numbers.length() << " is the length of the array" << endl;

break;

case 'p':{cin >> num;

if(numbers.present(num) == false)

cout << num << " was not found in the list" << endl;

else

cout << num << " was found in the list" << endl;

break;

}

case 'k':{cin >> num;

cout << "the kth number in the array is ";

numbers.kth(num);

cout << endl;

break;

}

case 'w': cout << "List:" << endl;

numbers.print();

break;

case 'h': printmenu();

}

}

return 0;

}

//post: function prints the menu

void printmenu()

{

cout << "Commands" << endl

<< "e -- Empty the list" << endl

<< "i v -- Insert an integer (v) into the list" << endl

<< "r v -- Remove an integer (v) from the list" << endl

<< "l -- Print the length of the list" << endl

<< "p v -- Searches the list for an integer (v)" << endl

<< "k v -- Print the kth value of the list" << endl

<< "w -- Print the list" << endl

<< "h -- See this menu" << endl

<< "q -- Quit" << endl;

}

//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\*prev;

if(first == NULL || entry < first->data)

first = get\_node(entry, first);

else

{

prev = first;

while(prev->next != NULL && prev->next->data < entry)

prev = prev->next;

prev->next = get\_node(entry, prev->next);

}

}

//pre: user enters a number to remove post:function removes the value from the list

void list::remove(item target)

{

node\*prev;

node\*temp;

prev=first;

if(first == NULL){

cout << target << " was not in the list" << endl;

return;

}

if(first -> data == target)

{

first = first ->next;

delete prev;

}

else

{

while(prev->next != NULL && prev->next->data != target)

prev = prev->next;

if(prev->next == NULL){

cout << target << " was not in the list" << endl;

return;

}

temp = prev->next;

prev->next = temp->next;

delete temp;

}

cout << target << " was removed from the list" << endl;

}

//post:function prints the list

void list::print()

{

node\*p;

p = first;

while(p != NULL){

cout << p->data << 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;

}

//pre: user enters a number to search for post: function returns true or false if found

bool list::present(item target)

{

node\*p;

p = first;

while(p != NULL){

if(p->data == target)

return true;

p = p->next;

}

return false;

}

//pre: user enters a number post: function returns a number in the position of the list

void list::kth(int k)

{

int count = 0;

node\*p;

p = first;

if(k > length()){

cout << "non existing" << endl;

return;

}

while(count != k - 1){

p = p->next;

count++;

}

cout << p->data;

}

Samuel Larson

Project 3

9/24/2018

User Document

This program will allow you to create a dynamic list of numbers.

The program will display this menu when launched:

Commands

e -- Empty the list

i v -- Insert an integer (v) into the list

r v -- Remove an integer (v) from the list

l -- Print the length of the list

p v -- Searches the list for an integer (v)

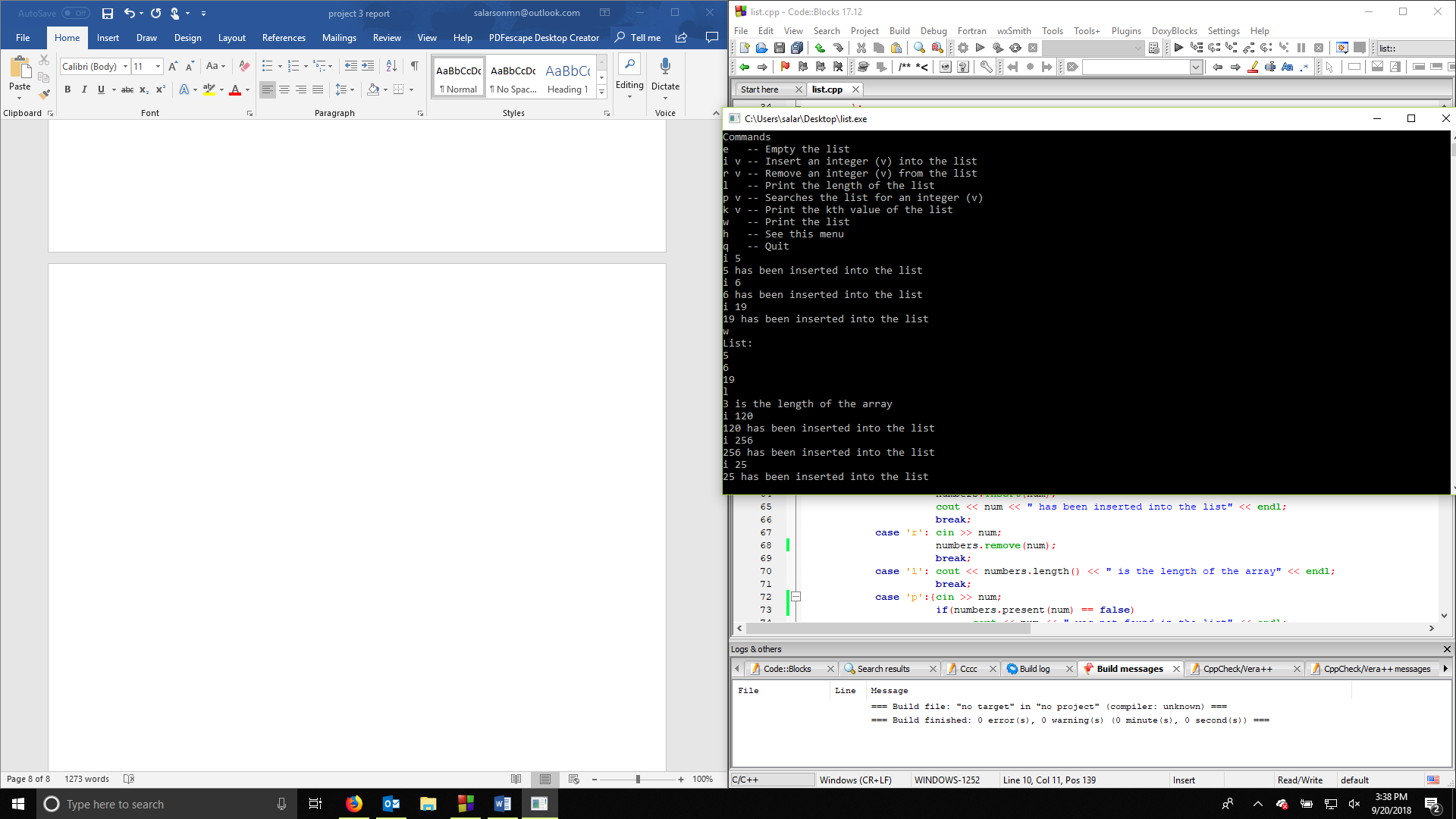
k v -- Print the kth value of the list

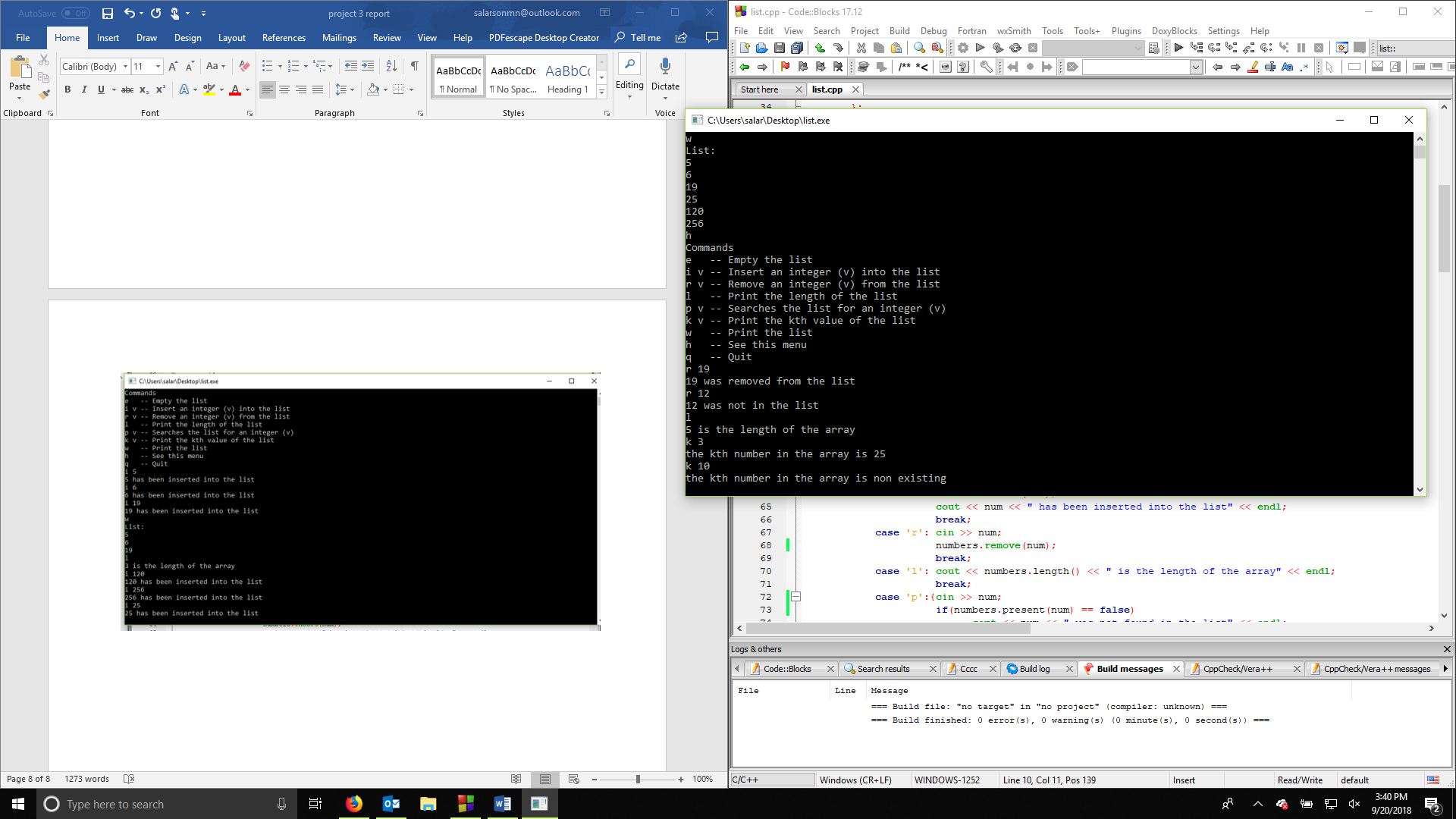
w -- Print the list

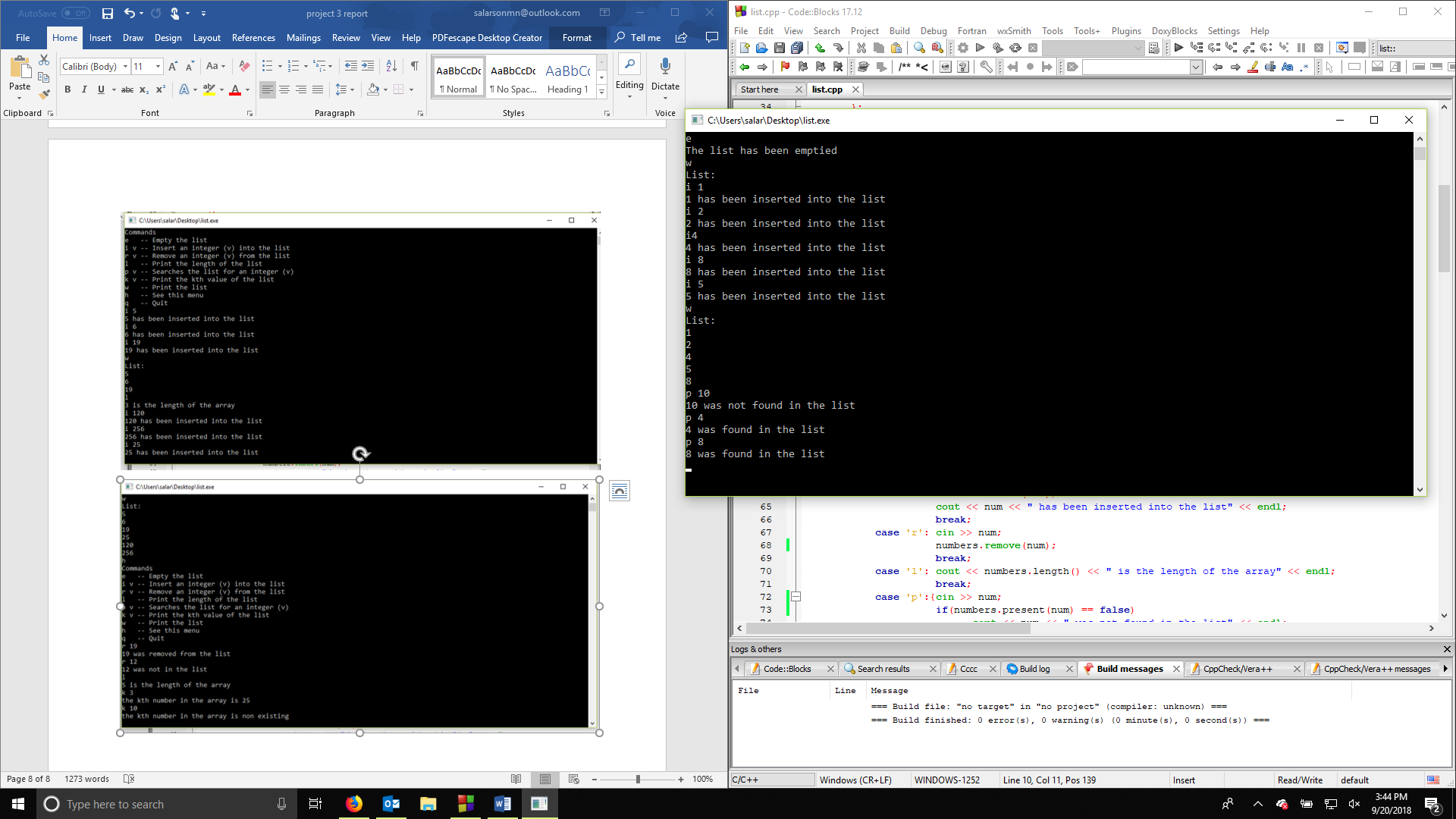
h -- See this menu

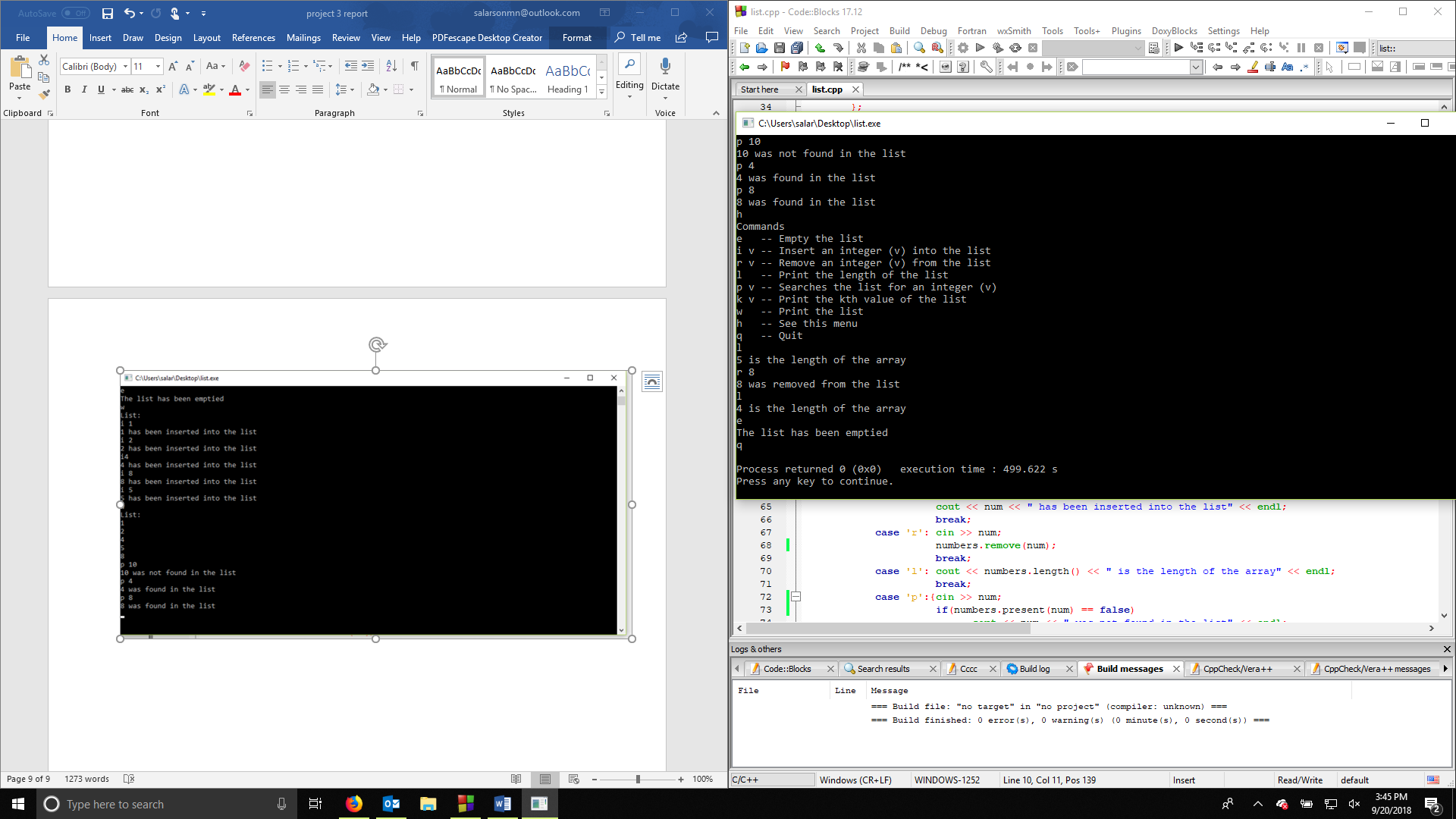
q -- Quit

to empty a list you just need to enter e into the terminal, to add a number to the list enter i and the number you would like to enter, to remove a number from the list enter r and the number you would like to remove, if you would like to find the length of the list enter l, if you would like to search the list for a number enter p and the number you would like to search for, if you want to print a number in a specific position of the array enter k and the number of the position, if you enter w the program will print the full list of numbers, if you ever need this menu in the program enter h, and enter q to exit the program.









Samuel Larson

Project 3

9/24/2018

With this project I demonstrated how to create a list and use it as dynamic storage. We had to make a list of functions that the user of the program could use to manipulate the list and retrieve information.

The program had to be able to empty the list, insert a number in order of least to greatest, remove a number, find and print the length of the list to the user, search the list for an integer, print the kth value of the list for the user, print the full list for the user, redisplay the list of commands for the user, or quit the program.

This program could have been improved by making it so when the user searches the list for a number it would display the position of the number. To do this I would have to add a counter to the loop that searches for the number that increments every time the pointer was moved down the list until the number is found. Another thing I could have done to improve this project is making it so the user can save the list they created to a document or read in a list from a document. I would have to make two new functions save() and load(). Save() would need to go down the list and output each value in the list to a file. The load function would need to read numbers in from a list and similar to the insert function create nodes and place them in numerical order.