**CSIII: Programming Patterns**

**Lab 2 Assignment: Word Count**

The project is due in one week: by the beginning of the next lab.

Implement a program that reads in a text file, counts how many times each word occurs in the file and outputs the words (and counts) in the decreasing order of occurrence, i.e. the counts need to be output in sorted order. *Word* is any sequence of alphanumeric characters. Whitespace and punctuation marks are to be discarded. That is, the punctuation marks should not be counted either as a part of the word or as a separate word. You are free to make your program case sensitive (Hello and hello are counted as separate words) or case insensitive. File name is supplied on command line. You are to use the following classes.

class WordOccurrence

{

public:

WordOccurrence(const string& word="", int num=0);

bool matchWord(const string &); // returns true if word matches stored

void increment(); // increments number of occurrences

string getWord() const;

int getNum() const;

private:

string word\_;

int num\_;

};

class WordList

{

public:

// add copy constructor, destructor, overloaded assignment

void addWord(const string &);

void printList();

private:

// a dynamically allocated array of WordOccurrences

// may or may not be sorted

WordOccurrence \*wordArray\_;

int size\_;

};

Using vectors is not allowed. You may use standard sorting algorithms or implement insertion sort from scratch. For the second class (WordList), implement a copy constructor (implementing deep copy), destructor and an overloaded assignment (either classical or copy-and-swap). Make sure your class works correctly with the “WordListDriver.cpp” For your constructors, use member initialization lists where possible, use default values for function parameters where appropriate. Enhance class interface if necessary.

**Milestone**. Code that inputs a file and prints a list of words (possibly repeating).

**Hint:** you can use ctype functions to check if a character is alphanumeric. Using following example.

// Prints a table of the character manipulation functions output

// Author: Sean McCulloch

// Date 1/16/2018

#include

#include

using namespace std;

int main()

{

char c; // The character to manipulate

do

{

cout << "Enter the Character(n to stop): ";

cin >> c;

cout << "Function" << " Result" << endl << "-------------------" << endl

// the functions return non-zero value if true

<< "isalnum" << " " << isalnum(c) << endl << "isalpha" << " " << isalpha(c)

<< endl << "iscntrl" << " " << iscntrl(c) << endl << "isdigit" << " " << isdigit(c) << endl << "isgraph" << " " << isgraph(c) << endl << "islower" << " "

<< islower(c) << endl << "isprint" << " " << isprint(c) << endl << "ispunct"

<< " " << ispunct(c) << endl << "isspace" << " " << isspace(c) << endl

<< "isupper" << " " << isupper(c) << endl

// the functions return integers, we use type casting (char) to have it printed in //character representation

<< "tolower" << " " << char(tolower(c)) << endl << "toupper" << " "

<< char(toupper(c)) << endl;

}while(c!='n');

Return 0;

}