```cpp
// Name: James Small
// Program: 3A
// Class: CSE455
// Description: Counter class Implementation File

#include "Counter.h"
#include <fstream>
#include <iostream>

using namespace std;

// This is the default constructor

Counter::Counter(string fileName)
{
    this->currentFileName = classNameWithoutExtension(fileName);
}

// This method calcualtes the LOC in a file

void Counter::calculateLOC()
{
    ifstream infile;

    int count = 0;
    string currentString;

    currentFileName.append(".h");

    for (int i = 0; i < 2; i++) {
        infile.open(currentFileName.c_str());

        while (getline(infile,currentString))
            if ((currentString.find_first_not_of(' ') != string::npos))
                if ((currentString.find_first_not_of('\n') != string::npos))
                    if ((currentString.find_first_not_of('\r') != string::npos))
                                if (!(currentString[0] == '/'))
                                        count++;

        infile.close();

        currentFileName = classNameWithoutExtension(currentFileName);
        currentFileName.append(".cpp");
    }

    int methodC = methodCount();

    ClassInfo currentClass(classNameWithoutExtension(currentFileName), methodC, count);

    classVector.push_back(currentClass);

    vector<string> currentClasses = classListVector();

    for (int i = 0; i < currentClasses.size(); i++) {
        currentFileName = classNameWithoutExtension(currentClasses[i]);
        calculateLOC();
    }
}

// This method returns the number of methods in a given class

int Counter::methodCount()
{
    string currentString;

    string stringToCheck = classNameWithoutExtension(currentFileName);
    stringToCheck.append("::");
```

```
    currentFileName = classNameWithoutExtension(currentFileName);
    currentFileName.append(".cpp");

    int methodCount = 0;

    ifstream infile;

    infile.open(currentFileName.c_str());

    while (getline(infile,currentString))
        if (currentString.find(stringToCheck) != string::npos)
            methodCount++;

    infile.close();

    return methodCount;
}

// This method returns a vector listing the names of all classes found in current class

vector<string> Counter::classListVector()
{
    string currentString;

    string stringToCheck = "#include \"";

    vector<string> classList;

    currentFileName = classNameWithoutExtension(currentFileName);

    currentFileName.append(".h");

    ifstream infile;

    for (int i = 0; i < 2; i++) {
        infile.open(currentFileName.c_str());

        while (getline(infile,currentString)) {

            size_t found = currentString.find(stringToCheck);

            if (found != string::npos) {

                string temp;

                for (int i = currentString.find('\"') + 1; i < currentString.length(); i++) {

                    if (currentString[i] != '\"')
                        temp.push_back(currentString[i]);
                    else
                        break;
                }

                bool notFound = true;

                for (int i = 0; i < classList.size(); i++) {
                    if (classList[i] == temp) {
                        notFound = false;
                        break;
                    }
                }

                for (int i = 0; i < classVector.size(); i++) {
                    if (classVector[i].getClassName() == classNameWithoutExtension(temp)) {
                        notFound = false;
                        break;
```

```
                    }
                }

                if (notFound)
                    classList.push_back(temp);
            }
        }

        infile.close();

        currentFileName = classNameWithoutExtension(currentFileName);
        currentFileName.append(".cpp");
    }

    return classList;
}

// This method returns the class name without an extension on it

string Counter::classNameWithoutExtension(string className)
{
    string temp;

    for (int i = 0; i < className.length(); i++) {
        if (className[i] != '.')
            temp.push_back(className[i]);
        else
            break;
    }

    return temp;
}

// This method displays the results of the LOC for all files in the program

void Counter::displayReport()
{
    int masterCount = 0;

    cout << "\nProgram Name: " << classVector[0].getClassName() << "\n\n";
    masterCount += classVector[0].getLineCount();

    for (int i = 1; i < classVector.size(); i++) {

        cout << "Class Name: " << classVector[i].getClassName() << endl;
        cout << "Method Count: " << classVector[i].getMethodCount() << endl;
        cout << "Class Line Count: " << classVector[i].getLineCount() << "\n\n";
        masterCount += classVector[i].getLineCount();
    }

    cout << classVector[0].getClassName() << " Master Count: " << masterCount << "\n\n";
}
```