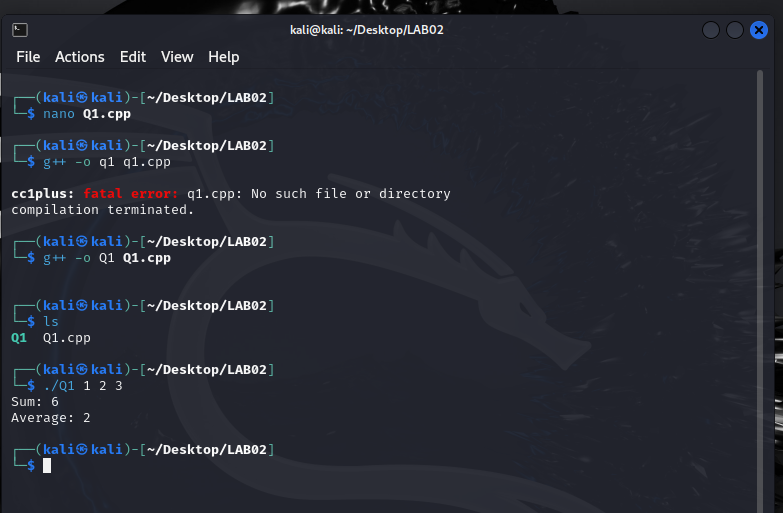
**Post-Lab Questions:**

**TASK # 1**



**Code:**

#include <iostream>

#include <cstdlib>

using namespace std;

int main(int argc, char \*argv[]) {

if (argc <= 1) {

cout << "Error: No integers provided.\n";

return 1;

}

int sum = 0;

int count = 0;

int \*arr = new int[argc - 1];

// Loop through command line arguments starting from index 1

for (int i = 1; i < argc; ++i) {

arr[i - 1] = atoi(argv[i]);

sum += arr[i - 1];

count++;

}

double average = static\_cast<double>(sum) / count;

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

cout << "Average: " << average << endl;

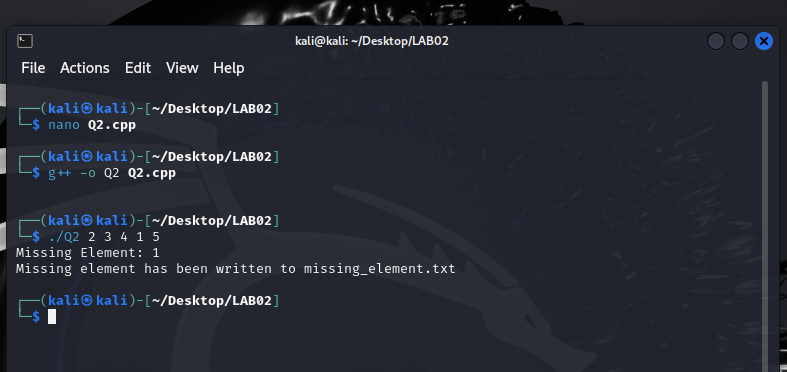
// Free dynamically allocated memory

delete[] arr;

return 0;

}

**TASK # 2**



**Code:**

#include <iostream>

#include <fstream>

#include <vector>

#include <cstdlib>

using namespace std;

int main(int argc, char \*argv[]) {

if (argc <= 2) {

cout << "Error: Insufficient integers provided.\n";

return 1;

}

vector<int> series;

// Loop through command line arguments starting from index 2

for (int i = 2; i < argc; ++i) {

series.push\_back(atoi(argv[i]);

}

int missingElement = atoi(argv[1]);

for (int num : series) {

missingElement ^= num;

}

ofstream outFile("missing\_element.txt");

if (!outFile) {

cerr << "Error opening file for writing." << endl;

return 1;

}

outFile << "Missing Element: " << missingElement << endl;

cout << "Missing Element: " << missingElement << endl;

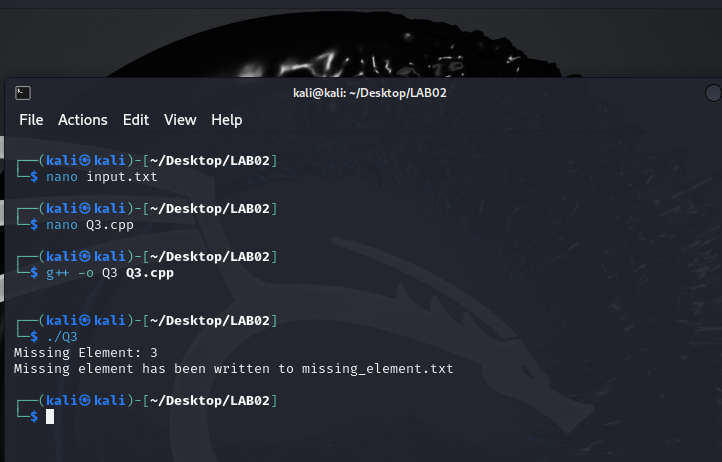
cout << "Missing element has been written to missing\_element.txt" << endl;

outFile.close();

return 0;

}

**TASK # 3**

****

**Code:**

#include <iostream>

#include <fstream>

#include <vector>

#include <algorithm>

using namespace std;

int findMissing(const vector<int>& series) {

int n = series.size() + 1;

int total = (n \* (n + 1)) / 2;

int sum = 0;

for (int num : series) {

sum += num;

}

return total - sum;

}

int main() {

string filename = "input.txt";

ifstream inFile(filename);

if (!inFile.is\_open()) {

cerr << "Error: Unable to open file " << filename << endl;

return 1;

}

vector<int> series;

int num;

while (inFile >> num) {

series.push\_back(num);

}

inFile.close();

int missingElement = findMissing(series);

string outputFilename = "missing\_element.txt";

ofstream outFile(outputFilename);

if (!outFile.is\_open()) {

cerr << "Error: Unable to open file " << outputFilename << " for writing." << endl;

return 1;

}

outFile << "Missing Element: " << missingElement << endl;

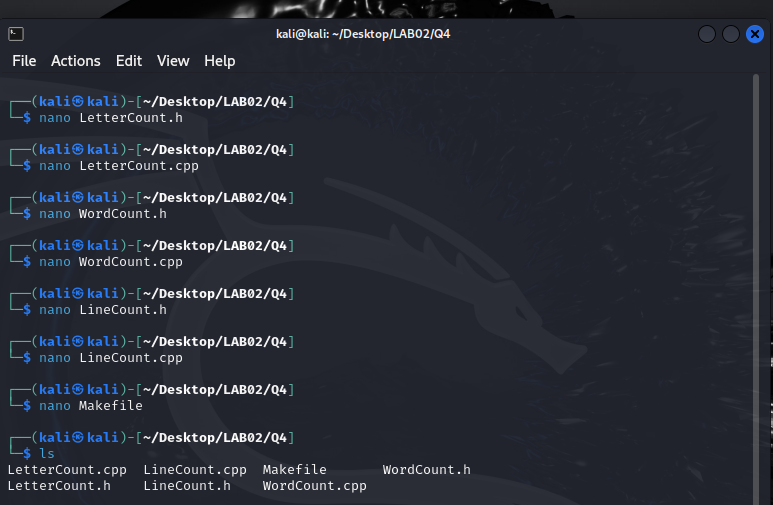
cout << "Missing Element: " << missingElement << endl;

cout << "Missing element has been written to " << outputFilename << endl

outFile.close();

return 0;}

**TASK # 4**



**Codes:**

LetterCount.h:

#ifndef LETTERCOUNT\_H

#define LETTERCOUNT\_H

#include <string>

#include <fstream>

class LetterCount {

private:

int count;

public:

LetterCount();

void countLetters(const std::string& filename);

int getCount() const;

};

#endif // LETTERCOUNT\_H

LetterCount.cpp:

#include "LetterCount.h"

LetterCount::LetterCount() : count(0) {}

void LetterCount::countLetters(const std::string& filename) {

std::ifstream inFile(filename);

if (!inFile.is\_open()) {

std::cerr << "Error: Unable to open file " << filename << std::endl;

return;

}

char ch;

while (inFile.get(ch)) {

if (std::isalpha(ch)) {

count++;

}

}

inFile.close();

}

int LetterCount::getCount() const {

return count;

}

WordCount.h:

#ifndef WORDCOUNT\_H

#define WORDCOUNT\_H

#include <string>

#include <fstream>

class WordCount {

private:

int count;

public:

WordCount();

void countWords(const std::string& filename);

int getCount() const;

};

#endif // WORDCOUNT\_H

WordCount.cpp:

#include "WordCount.h"

WordCount::WordCount() : count(0) {}

void WordCount::countWords(const std::string& filename) {

std::ifstream inFile(filename);

if (!inFile.is\_open()) {

std::cerr << "Error: Unable to open file " << filename << std::endl;

return;

}

std::string word;

while (inFile >> word) {

count++;

}

inFile.close();

}

int WordCount::getCount() const {

return count;

}

LineCount.h:

#ifndef LINECOUNT\_H

#define LINECOUNT\_H

#include <string>

#include <fstream>

class LineCount {

private:

int count;

public:

LineCount();

void countLines(const std::string& filename);

int getCount() const;

};

#endif // LINECOUNT\_H

LineCount.cpp:

#include "LineCount.h"

LineCount::LineCount() : count(0) {}

void LineCount::countLines(const std::string& filename) {

std::ifstream inFile(filename);

if (!inFile.is\_open()) {

std::cerr << "Error: Unable to open file " << filename << std::endl;

return;

}

std::string line;

while (std::getline(inFile, line)) {

count++;

}

inFile.close();

}

int LineCount::getCount() const {

return count;

}

Now, let's create a Makefile to compile these classes:

Makefile:

CC = g++

CFLAGS = -std=c++11 -Wall

all: main

main: main.o LetterCount.o WordCount.o LineCount.o

$(CC) $(CFLAGS) $^ -o $@

main.o: main.cpp

$(CC) $(CFLAGS) -c $<

LetterCount.o: LetterCount.cpp LetterCount.h

$(CC) $(CFLAGS) -c $<

WordCount.o: WordCount.cpp WordCount.h

$(CC) $(CFLAGS) -c $<

LineCount.o: LineCount.cpp LineCount.h

$(CC) $(CFLAGS) -c $<

clean:

rm -rf \*.o main