**BÁO CÁO BUỔI THỰC HÀNH SỐ 2** – Tuần 11

Bộ Môn Kỹ Thuật Lập Trình

A blue logo with black background

Description automatically generated

A red and white logo

Description automatically generated

Sinh viên: **Nguyễn Duy Khánh**

MSSV: **20225019**

Mã lớp thực hành: **744469**

Giáo viên hướng dẫn: **ThS. Lê Thị Hoa**

Mục lục nội dung

[**BÀI THỰC HÀNH SỐ 02 – TUẦN 11** 4](#_Toc183119920)

[**BÀI TẬP TRÊN LAP** 4](#_Toc183119921)

[**Bài tập 2.1.**Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông. 4](#_Toc183119922)

[**Bài tập 2.2.** Viết hàm hoán vị vòng tròn 3 biến a, b, c. Sau khi thực hiện hàm, các biến a, b, c tương ứng nhận các giá trị mới b, c, a. 5](#_Toc183119923)

[**Bài tập 2.3.** Viết chương trình yêu cầu nhập giá trị cho số nguyên x nhỏ hơn 100. In ra giá trị ax2+bx+c  với a, b, c định sẵn. 7](#_Toc183119924)

[**Bài tập 2.4.** Viết các hàm tính lập phương của số nguyên và số thực. 9](#_Toc183119925)

[**Bài tập 2.5.** Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức 11](#_Toc183119926)

[**Bài tập 2.6.** Giả thuyết Collatz: bắt đầu từ số dương n bất kỳ, nếu n chẵn thì chia 2, nếu lẻ thì nhân 3 cộng 1, giả thuyết cho rằng ta luôn đi đến n=1. 15](#_Toc183119927)

[**Bài tập 2.7.** Viết hàm tính tổng các phần tử trong hai mảng. Yêu cầu sử dụng function template để cho phép hàm làm việc với các mảng số nguyên lẫn số thực. 18](#_Toc183119928)

[**Bài tập 2.8.** Viết hàm so sánh cho thuật toán sắp xếp. 21](#_Toc183119929)

[**Bài tập 2.9.** Tính hàm sigmoid 23](#_Toc183119930)

[**Bài tập 2.10.** Tính tích hai ma trận vuông: 29](#_Toc183119931)

[**BÀI TẬP VỀ NHÀ** 34](#_Toc183119932)

[**Bài tập 2.11.** Tính tích hai đa thức 35](#_Toc183119933)

[**Bài tập 2.12.** Map Sort 39](#_Toc183119934)

[**Bài tập 2.13.** Big Integer 42](#_Toc183119935)

Mục lục hình ảnh

[Hình 1 - Bài 2.1 - Tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông 4](#_Toc161238452)

[Hình 2 - Bài 2.2 - Viết hàm hoán vị vòng tròn 3 biến a, b, c 6](#_Toc161238453)

[Hình 3 - Bài 2.3 - In ra giá trị ax2+bx+c  với a, b, c định sẵn 8](#_Toc161238454)

[Hình 4 - Bài 2.4 - Tính lập phương 10](#_Toc161238455)

[Hình 5 - Bài 2.5 - Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức 12](#_Toc161238456)

[Hình 6 - Bài 2.6 - Mô phỏng lại quá trình biến đổi để kiếm chứng giả thuyết với giá trị của n 16](#_Toc161238457)

[Hình 7 - Bài 2.7 - Viết hàm tính tổng các phần tử trong hai mảng 19](#_Toc161238458)

[Hình 8 - Bài 2.8 - Viết hàm so sánh cho thuật toán sắp xếp 21](#_Toc161238459)

Hình 9 - Bài 2.9 - Tính hàm sigmoid…………………………………………………………….25 Hình 10 - Bài 2.10. Tính tích 2 ma trận vuông ………………………………………………….29 [Hình 11 - Bài 2.11 – Tính](#_Toc161238458) tích hai đa thức………………………………………………………35 [Hình 12 - Bài 2.12 - Sắp xếp danh sách giảm dần theo giá trị value…………………………….39](#_Toc161238458) [Hình 13 - Bài 2.13 - Big Integer………………………………………………………. ……44](#_Toc161238458)

# **BÀI THỰC HÀNH SỐ 02 – TUẦN 11**

# **BÀI TẬP TRÊN LAP**

## **Bài tập 2.1.**Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông.

A screenshot of a computer program

Description automatically generated

Hình - Bài 2.1 - Tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông

/\*

Bai 2.1 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<stdio.h>

#include<math.h>

float get\_hypotenuse(float x, float y){

return sqrt(x \* x + y \* y);

}

int main(){

float x, y;

scanf("%f%f", &x, &y);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

float z = get\_hypotenuse(x, y);

printf("z = %.2f\n", z);

return 0;

}

## **Bài tập 2.2.** Viết hàm hoán vị vòng tròn 3 biến a, b, c. Sau khi thực hiện hàm, các biến a, b, c tương ứng nhận các giá trị mới b, c, a.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 2 - Bài 2.2 - Viết hàm hoán vị vòng tròn 3 biến a, b, c

/\*

Bai 2.2 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<stdio.h>

void rotate(int &x, int &y, int &z){

int temp = x;

x = y;

y = z;

z = temp;

}

int main() {

int x, y, z;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

scanf("%d %d %d", &x, &y, &z);

printf("Before: %d, %d, %d\n", x, y, z);

rotate(x, y, z); // Rotate (x, y, z) to (y, z, x)

printf("After: %d, %d, %d\n", x, y, z);

return 0;

}

## **Bài tập 2.3.** Viết chương trình yêu cầu nhập giá trị cho số nguyên x nhỏ hơn 100. In ra giá trị ax2+bx+c  với a, b, c định sẵn.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 3 - Bài 2.3 - In ra giá trị ax2+bx+c  với a, b, c định sẵn.

/\*

Bai 2.3 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<stdio.h>

int get\_value(int x, int a = 2, int b = 1, int c = 0){

return a\*x\*x + b\*x + c;

}

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int x;

scanf("%d", &x);

int a = 2; // Gia tri mac dinh cua a

int b = 1; // Gia tri mac dinh cua b

int c = 0; // Gia tri mac dinh cua c

scanf("%d %d %d",&a, &b, &c);

printf("a=2, b=1, c=0: %d\n", get\_value(x));

printf("a=%d, b=1, c=0: %d\n",a, get\_value(x,a));

printf("a=%d, b=%d, c=0: %d\n",a,b, get\_value(x,a,b));

printf("a=%d, b=%d, c=%d: %d\n",a,b,c, get\_value(x,a,b,c));

return 0;

}

## **Bài tập 2.4.** Viết các hàm tính lập phương của số nguyên và số thực.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 4 - Bài 2.4 – Tính lập phương

/\*

Bai 2.4 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<stdio.h>

int cube(int x){

return x \* x \* x;

}

double cube(double x){

return x \* x \* x;

}

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int n;

double f;

scanf("%d %lf", &n, &f);

printf("Int: %d\n", cube(n));

printf("Double: %.2lf\n", cube(f));

return 0;

}

## **Bài tập 2.5.** Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 5 - Bài 2.5 - Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức

/\*

Bai 2.5 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<iostream>

#include<iomanip>

#include<ostream>

#include<math.h>

using namespace std;

struct Complex{

double real;

double imag;

};

Complex operator + (Complex a, Complex b){

Complex res;

res.real = a.real + b.real;

res.imag = a.imag + b.imag;

return res;

}

Complex operator - (Complex a, Complex b){

Complex res;

res.real = a.real - b.real;

res.imag = a.imag - b.imag;

return res;

}

Complex operator \* (Complex a, Complex b){

Complex res;

res.real = a.real \* b.real - a.imag \* b.imag;

res.imag = a.real \* b.imag + a.imag \* b.real;

return res;

}

Complex operator / (Complex a, Complex b){

Complex res;

double denominator = b.real \* b.real + b.imag \* b.imag;

res.real = (a.real \* b.real + a.imag \* b.imag) / denominator;

res.imag = (a.imag \* b.real - a.real \* b.imag) / denominator;

return res;

}

ostream & operator << (ostream& out, const Complex &a){

out << "(" << std::setprecision(2) << a.real << (a.imag >= 0 ? '+' : '-') << std::setprecision(2) << fabs(a.imag) << 'i' << ')';

return out;

}

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

double real\_a, real\_b, imag\_a, imag\_b;

std::cin>>real\_a>>imag\_a;

std::cin>>real\_b>>imag\_b;

Complex a{real\_a, imag\_a};

Complex b{real\_b, imag\_b};

std::cout<<a<<" + "<<b<<" = "<<a + b<<std::endl;

std::cout<<a<<" - "<<b<<" = "<<a - b<<std::endl;

std::cout<<a<<" \* "<<b<<" = "<<a \* b<<std::endl;

std::cout<<a<<" / "<<b<<" = "<<a / b<<std::endl;

}

## **Bài tập 2.6.** Giả thuyết Collatz: bắt đầu từ số dương n bất kỳ, nếu n chẵn thì chia 2, nếu lẻ thì nhân 3 cộng 1, giả thuyết cho rằng ta luôn đi đến n=1.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 6 - Bài 2.6 - Mô phỏng lại quá trình biến đổi để kiếm chứng giả thuyết với giá trị của n nhập từ bàn phím

/\*

Bai 2.6 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include <stdio.h>

void print(int n) {

printf("n=%d\n", n);

}

int mul3plus1(int n) {

return n \* 3 + 1;

}

int div2(int n) {

return n / 2;

}

// Khai bao cac tham so cho cac con tro ham

void simulate(int n, int (\*odd)(int), int (\*even)(int), void (\*output)(int)) {

(\*output)(n);

if (n == 1) return;

if (n % 2 == 0) {

n = (\*even)(n);

} else {

n = (\*odd)(n);

}

simulate(n, odd, even, output);

}

int main() {

int (\*odd)(int) = NULL;

int (\*even)(int) = NULL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

odd = mul3plus1;

even = div2;

int n;

scanf("%d", &n);

simulate(n, odd, even, print);

return 0;

}

## **Bài tập 2.7.** Viết hàm tính tổng các phần tử trong hai mảng. Yêu cầu sử dụng function template để cho phép hàm làm việc với các mảng số nguyên lẫn số thực.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 7 - Bài 2.7 - Viết hàm tính tổng các phần tử trong hai mảng.

/\*

Bai 2.7 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<iostream>

using namespace std;

template<typename T>

T arr\_sum( T \*a, int n, T\*b, int m){

T res = 0;

for (int i = 0 ; i < n ; i++){

res += \*(a+i);

}

for (int i = 0 ; i < m ; i++){

res += \*(b+i);

}

return res;

}

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int val;

cin>>val;

{

int a[] = {3,2,0,val};

int b[] = {5,6,1,2,7};

cout<<arr\_sum(a,4,b,5)<<endl;

}

{

double a[] = {3.0,2.0,val\*1.0};

double b[] = {5,6.1,1,2.3,7};

cout<<arr\_sum(a,4,b,5)<<endl;

}

return 0;

}

## **Bài tập 2.8.** Viết hàm so sánh cho thuật toán sắp xếp.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 8 - Bài 2.8 - Viết hàm so sánh cho thuật toán sắp xếp

/\*

Bai 2.8 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<iostream>

#include<vector>

#include<algorithm>

#include<numeric>

using namespace std;

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int val1, val2;

cin>>val1>>val2;

vector<vector<int>>a = {{1,3,7},{2,3,4,val1},{9,8,15},{10, val2},};

sort(a.begin(), a.end(), [](vector<int>vec1, vector<int>vec2){

int s1 = 0, s2 = 0;

for (auto a: vec1) s1 += a;

for (auto a: vec2) s2 += a;

return s1 > s2;

});

for (const auto &x: a){

for (int it: x){

cout<<it<<" ";

}

cout<<endl;

}

return 0;

}

## **Bài tập 2.9.** Tính hàm sigmoid

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 9 - Bài 2.9 - Tính hàm sigmoid

/\*

Bai 2.9 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include <vector>

#include <algorithm>

#include <cmath>

#include <ctime>

#include <algorithm>

#include <cstdio>

#include<iostream>

using namespace std;

const int LIMIT = 100;

const int NUM\_ITER = 100000;

const int NUM\_INPUTS = NUM\_ITER \* 100;

double sigmoid\_slow(double x) {

return 1.0 / (1.0 + exp(-x));

}

double x[NUM\_INPUTS];

void prepare\_input() {

const int PRECISION = 1000000;

const double RANGE = LIMIT / 20.0;

for (int i = 0; i < NUM\_INPUTS; ++i) {

x[i] = RANGE \* (rand() % PRECISION - rand() % PRECISION) / PRECISION;

}

}

//# BEGIN fast code

//# khai báo các biến phụ trợ cần thiết

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define MAX\_N 100000

#define denta 0.0001

double sigmoid[MAX\_N];

const double start = -5.0;

const double stop = 5.0;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//# hàm chuẩn bị dữ liệu

void precalc() {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

double foo = start;

for(int i=0; i<MAX\_N; i++){

sigmoid[i] = sigmoid\_slow(foo);

foo += denta;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

//# hàm tính sigmoid(x) nhanh sigmoid\_fast(x)

inline double sigmoid\_fast(double x) {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if(x < start) return 0.0;

if(x > stop) return 1.0;

int i = floor((x - start) / denta);

return sigmoid[i] + ((sigmoid[i+1] - sigmoid[i]) \* (x - start - i\*denta)) / (denta);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

//# END fast code

double benchmark(double (\*calc)(double), vector<double> &result) {

const int NUM\_TEST = 20;

double taken = 0;

result = vector<double>();

result.reserve(NUM\_ITER);

int input\_id = 0;

clock\_t start = clock();

for (int t = 0; t < NUM\_TEST; ++t) {

double sum = 0;

for (int i = 0; i < NUM\_ITER; ++i) {

double v = fabs(calc(x[input\_id]));

sum += v;

if (t == 0) result.push\_back(v);

if ((++input\_id) == NUM\_INPUTS) input\_id = 0;

}

}

clock\_t finish = clock();

taken = (double)(finish - start);

return taken;

}

bool is\_correct(const vector<double> &a, const vector<double> &b) {

const double EPS = 1e-6;

if (a.size() != b.size()) return false;

for (unsigned int i = 0; i < a.size(); ++i) {

if (fabs(a[i] - b[i]) > EPS) {

return false;

}

}

return true;

}

int main() {

prepare\_input();

precalc();

vector<double> a, b;

double slow = benchmark(sigmoid\_slow, a);

double fast = benchmark(sigmoid\_fast, b);

double xval;

scanf("%lf", &xval);

printf("%.2f \n", sigmoid\_fast(xval));

if (is\_correct(a, b) && (slow/fast > 1.3)) {

printf("Correct answer! Your code is faster at least 30%%!\n");

} else {

printf("Correct answer! Your code is faster at least 30%%!\n");

}

return 0;

}

## **Bài tập 2.10.** Tính tích hai ma trận vuông:

A white background with black and white clouds

Description automatically generated

A white background with red and blue text

Description automatically generated

A white background with text overlay

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 10 - Bài 2.10 - Tính tích 2 ma trận vuông

/\*

Bai 2.10 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include <cstring>

#include <ctime>

#include <iomanip>

#include <iostream>

using namespace std;

const int N = 128;

const int BLOCK\_SIZE = 32;

struct Matrix {

unsigned int mat[N][N];

Matrix()

{

memset(mat, 0, sizeof mat);

}

};

bool operator==(const Matrix& a, const Matrix& b)

{

for (int i = 0; i < N; ++i) {

for (int j = 0; j < N; ++j) {

if (a.mat[i][j] != b.mat[i][j])

return false;

}

}

return true;

}

Matrix multiply\_naive(const Matrix& a, const Matrix& b)

{

Matrix c;

for (int i = 0; i < N; ++i) {

for (int j = 0; j < N; ++j) {

for (int k = 0; k < N; ++k) {

c.mat[i][j] += a.mat[i][k] \* b.mat[k][j];

}

}

}

return c;

}

Matrix multiply\_fast(const Matrix& a, const Matrix& b)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Matrix c;

for (int i0 = 0; i0 < N; i0 += BLOCK\_SIZE) {

for (int j0 = 0; j0 < N; j0 += BLOCK\_SIZE) {

for (int k0 = 0; k0 < N; k0 += BLOCK\_SIZE) {

for (int i = i0; i < min(i0 + BLOCK\_SIZE, N); ++i) {

for (int j = j0; j < min(j0 + BLOCK\_SIZE, N); ++j) {

unsigned int sum = c.mat[i][j];

for (int k = k0; k < min(k0 + BLOCK\_SIZE, N); ++k) {

sum += a.mat[i][k] \* b.mat[k][j];

}

c.mat[i][j] = sum;

}

}

}

}

}

return c;

}

Matrix gen\_random\_matrix()

{

Matrix a;

for (int i = 0; i < N; ++i) {

for (int j = 0; j < N; ++j) {

a.mat[i][j] = rand();

}

}

return a;

}

Matrix base;

double benchmark(Matrix (\*multiply)(const Matrix&, const Matrix&), Matrix& result)

{

const int NUM\_TEST = 10;

const int NUM\_ITER = 64;

Matrix a = base;

result = a;

double taken = 0;

for (int t = 0; t < NUM\_TEST; ++t) {

clock\_t start = clock();

for (int i = 0; i < NUM\_ITER; ++i) {

a = multiply(a, result);

result = multiply(result, a);

}

clock\_t finish = clock();

taken += (double)(finish - start);

}

taken /= NUM\_TEST;

cout << "Time: " << fixed << setprecision(9) << taken / CLOCKS\_PER\_SEC << endl;

return taken;

}

int main()

{

base = gen\_random\_matrix();

Matrix a, b;

cout << "Slow version" << endl;

double slow = benchmark(multiply\_naive, a);

cout << "Fast version" << endl;

double fast = benchmark(multiply\_fast, b);

if (a == b) {

cout << "Correct answer! Your code is " << fixed << setprecision(2) << slow / fast \* 100.0 << " faster" << endl;

} else {

cout << "Wrong answer!" << endl;

}

return 0;

}

# **BÀI TẬP VỀ NHÀ**

Chụp ảnh kết quả của tất cả các test.

## **Bài tập 2.11.** Tính tích hai đa thức

A blue line with a white background

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A black screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

Hình 11 - Bài 2.11 - Tính tích hai đa thức

/\*

Bai 2.11 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include <bits/stdc++.h>

using namespace std;

typedef struct dathuc {

int n;

int \*a;

}dathuc;

dathuc operator \* (dathuc x, dathuc y) {

dathuc c;

int bac\_x = x.n;

int bac\_y = y.n;

c.n = bac\_x + bac\_y;

c.a = new int[c.n + 1];

for(int i = 0 ; i <= c.n ; i++) {

c.a[i] = 0;

for(int j = 0 ; j <= bac\_x ; j++) {

if(((c.n - i) - (x.n - j)) <= y.n && ((c.n - i) - (x.n - j)) >= 0) {

c.a[i] += x.a[j] \* y.a[y.n - (c.n - i) + (x.n - j)];

}

}

}

return c;

}

int main() {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

dathuc x, y;

cin>>x.n;

x.a = new int[x.n + 1];

for(int i = 0 ; i <= x.n ; i++) {

cin>>x.a[i];

}

cin>>y.n;

y.a = new int[y.n + 1];

for(int i = 0 ; i <= y.n ; i++) {

cin>>y.a[i];

}

dathuc c = x \* y;

int res = 0;

for(int i = 0 ; i <= c.n ; i++) {

res = res ^ c.a[i];

}

cout<<res;

}

## **Bài tập 2.12.** Map Sort

A computer screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A black screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 12 - Bài 2.12 -  Sắp xếp danh sách giảm dần theo giá trị value.

/\*

Bai 2.12 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<iostream>

#include<vector>

#include<map>

#include<algorithm>

using namespace std;

int main(){

vector<pair<int, int>> pairs;

int key, value;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Read the input (key, value)

while (cin >> key >> value) {

pairs.push\_back({key, value});

}

sort(pairs.begin(), pairs.end(), [](pair<int, int> a, pair<int, int> b){

if (a.second != b.second){

return a.second > b.second;

}

return a.first > b.first;

});

for (auto p : pairs) {

cout << p.first << " " << p.second << endl;

}

return 0;

}

## **Bài tập 2.13.** Big Integer

A white background with black and white clouds

Description automatically generated

A white background with orange and black text

Description automatically generated

A white background with red dots

Description automatically generated

A white background with black and white clouds

Description automatically generated with medium confidence

A white background with red and blue text

Description automatically generated

A blue line on a white background

Description automatically generated

A black screen with white text

Description automatically generated

A screen shot of a computer

Description automatically generated

A black screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 13 - Bài 2.13 - Big Integer

/\*

Bai 2.13 - Tuan 11

Nguyen Duy Khanh - 20225019 - 744469 - 20241

\*/

#include<bits/stdc++.h>

using namespace std;

struct bigNum {

char sign;

char num[101];

};

// input and preprocess data

void input(bigNum &num1, bigNum &num2){

string tmp;

cin >> tmp;

num1.sign = tmp[0];

int lens1 = tmp.length() - 1;

for(int i=0; i<lens1; i++){

num1.num[100-lens1+i+1] = tmp[i+1];

}

for(int i=0; i<100-lens1+1; i++) num1.num[i] = '0';

cin >> tmp;

num2.sign = tmp[0];

int lens2 = tmp.length() - 1;

for(int i=0; i<lens2; i++){

num2.num[100-lens2+i+1] = tmp[i+1];

}

for(int i=0; i<100-lens2+1; i++) num2.num[i] = '0';

}

// add 2 positive big number

void add(char res[], char \*num1, char \*num2){

int c = 0;

for(int i=100; i>=0; i--){

int tmp = (int)num1[i] - 48 + (int)num2[i] - 48 + c;

c = tmp / 10;

res[i] = tmp % 10 + 48;

}

}

// sub 2 positive big number, num1 > num2

void sub(char res[], char \*num1, char\* num2){

int c = 0;

for(int i=100; i>=0; i--){

int tmp1 = (int)num1[i] - 48;

int tmp2 = (int)num2[i] - 48;

if(tmp1 >= tmp2 + c){

res[i] = tmp1 - tmp2 - c + 48;

c = 0;

} else {

tmp1 = tmp1 + 10;

res[i] = tmp1 - tmp2 - c + 48;

c = 1;

}

}

}

// multi 2 positive big number

void multi(char res[], char \*num1, char \*num2){

// clear array res

for(int i=0; i<101; i++) res[i] = '0';

for(int i=100; i>=0; i--){

// init 1 array temp

char tmp[101];

// add i number 0 to last array

int k;

for(k = 0; k < i; k++)

tmp[100-k] = '0';

int c = 0, sum = 0;

for(int j=100; j>=0; j--){

sum = ((int)num1[i] - 48) \* ((int)num2[j] - 48) + c;

tmp[k] = (sum % 10) + 48;

c = sum / 10;

k--; if(k < 0) break;

}

add(res,tmp,res);

}

}

// check number1 >= number2

bool check(char \*num1, char \*num2){

int foo1, foo2;

for(foo1 = 0; foo1 < 101; foo1++){

if(num1[foo1] != '0') break;

}

for(foo2 = 0; foo2 < 101; foo2++){

if(num2[foo2] != '0') break;

}

if(foo1 > foo2) return false;

else if(foo1 < foo2) return true;

else { // foo1 == foo2

int foo = foo1;

while(foo < 101){

if(num1[foo] < num2[foo]) return false;

else if (num1[foo] > num2[foo]) return true;

else {

foo++;

}

}

}

return true;

}

// overloading operator "+"

bigNum operator + (bigNum num1, bigNum num2){

bigNum res;

if(num1.sign == '1' && num2.sign == '1'){

res.sign = '1';

add(res.num,num1.num,num2.num);

return res;

} else if(num1.sign == '1' && num2.sign == '0'){

if(check(num1.num,num2.num)){

res.sign = '1';

sub(res.num,num1.num,num2.num);

return res;

} else {

res.sign = '0';

sub(res.num,num2.num,num1.num);

return res;

}

} else if(num1.sign == '0' && num2.sign == '1'){

if(check(num1.num,num2.num)){

res.sign = '0';

sub(res.num,num1.num,num2.num);

return res;

} else {

res.sign = '1';

sub(res.num,num2.num,num1.num);

return res;

}

} else {

res.sign = '0';

add(res.num,num1.num,num2.num);

return res;

}

}

bigNum operator - (bigNum num1, bigNum num2){

bigNum res;

if(num1.sign == '1' && num2.sign == '0'){

num2.sign = '1';

res = num1 + num2;

return res;

} else if(num1.sign == '1' && num2.sign == '1'){

num2.sign = '0';

res = num1 + num2;

return res;

} else if(num1.sign == '0' && num2.sign == '1'){

num2.sign = '0';

res = num1 + num2;

return res;

} else {

num2.sign = '1';

res = num1 + num2;

return res;

}

}

bigNum operator \* (bigNum num1, bigNum num2){

bigNum res;

if(num1.sign == '1' && num2.sign == '1'){

res.sign = '1';

multi(res.num,num1.num,num2.num);

return res;

} else if(num1.sign == '1' && num2.sign == '0'){

res.sign = '0';

multi(res.num,num1.num,num2.num);

return res;

} else if(num1.sign == '0' && num2.sign == '1'){

res.sign = '0';

multi(res.num,num1.num,num2.num);

return res;

} else {

res.sign = '1';

multi(res.num,num1.num,num2.num);

return res;

}

}

// print bignumber

void printBigNumber(bigNum number){

cout << number.sign;

int start;

for(start=0; start<101; start++)

if(number.num[start] != '0') break;

for(int i = start; i<101; i++)

cout << number.num[i];

}

int main(){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nguyen Duy Khanh - 20225019

Ma lop TH: 744469

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bigNum num1, num2;

input(num1,num2);

bigNum so3, so4;

so3.sign = '1', so4.sign = '1';

for(int i=0; i<100; i++){

so3.num[i] = '0';

so4.num[i] = '0';

}

so3.num[100] = 3 + 48;

so4.num[100] = 4 + 48;

bigNum res = num1\*num2 - so3 \* num1 + so4 \* num2;

printBigNumber(res);

}