Contents

[Bài 2.1.  Viết hàm tính độ dài cạnh huyền của tam giác theo độ hai cạnh góc vuông. 1](#_Toc69157899)

[Bài 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. 3](#_Toc69157900)

[Bài 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. 5](#_Toc69157901)

[Bài 2.4. Viết các hàm tính lập phương của số nguyên và số thực. 6](#_Toc69157902)

[Bài 2.5. Viết các toán tử tính tổng, hiệu, tích và thương của hai số phức 8](#_Toc69157903)

[Bài 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. 11](#_Toc69157904)

[Hãy viết chương trình mô phỏng lại quá trình biến đổi để kiếm chứng giả thuyết với giá trị của n nhập từ bàn phím. 11](#_Toc69157905)

[Bài 2.7. Viết hàm tính tổng các phần tử trong hai mảng. 13](#_Toc69157906)

[Bài 2.8. Viết hàm so sánh cho thuật toán sắp xếp. 15](#_Toc69157907)

[Bài 2.9. Tính hàm sigmoid 17](#_Toc69157908)

[Bài tập 10 (bonus): Tính tích hai ma trận vuông 21](#_Toc69157909)

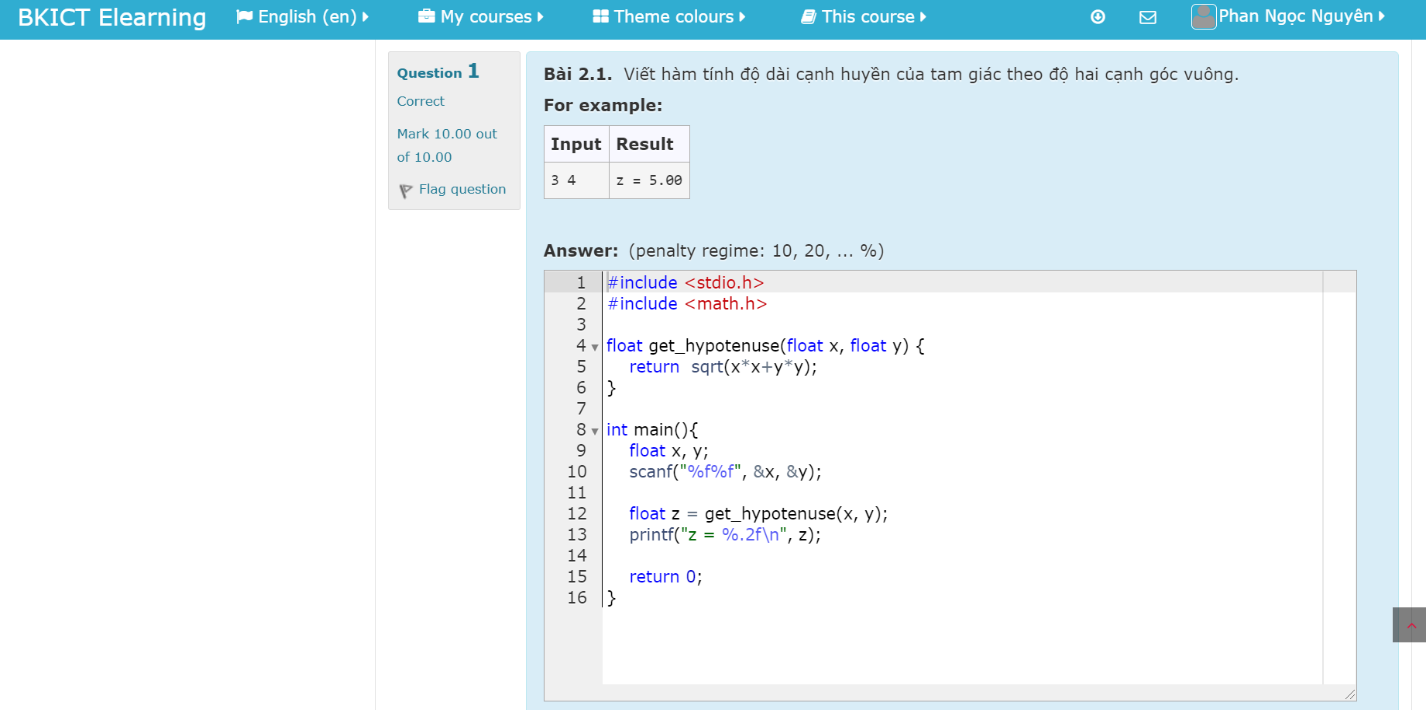
[26](#_Toc69157910)

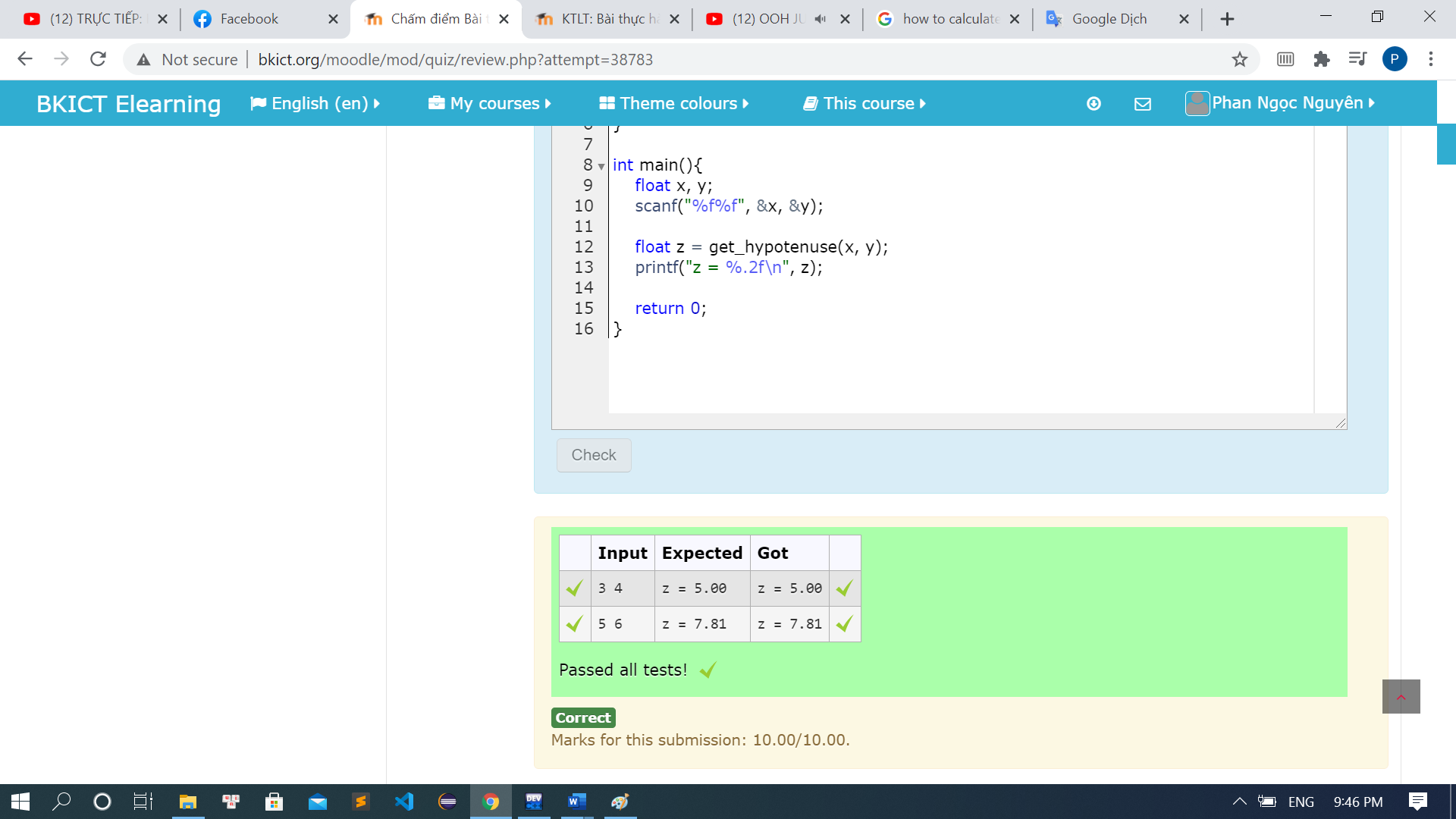
[Bài tập 11: Tính tích hai đa thức 26](#_Toc69157911)

[Bài tập 12: Map Sort 28](#_Toc69157912)

[Bài 13: Big Integer 30](#_Toc69157913)

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





Code:

#include <stdio.h>

#include <math.h>

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

return sqrt(pow(x,2)+pow(y,2));

}

int main(){

float x, y;

printf("Phan Ngoc Nguyen-20194134\n");

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

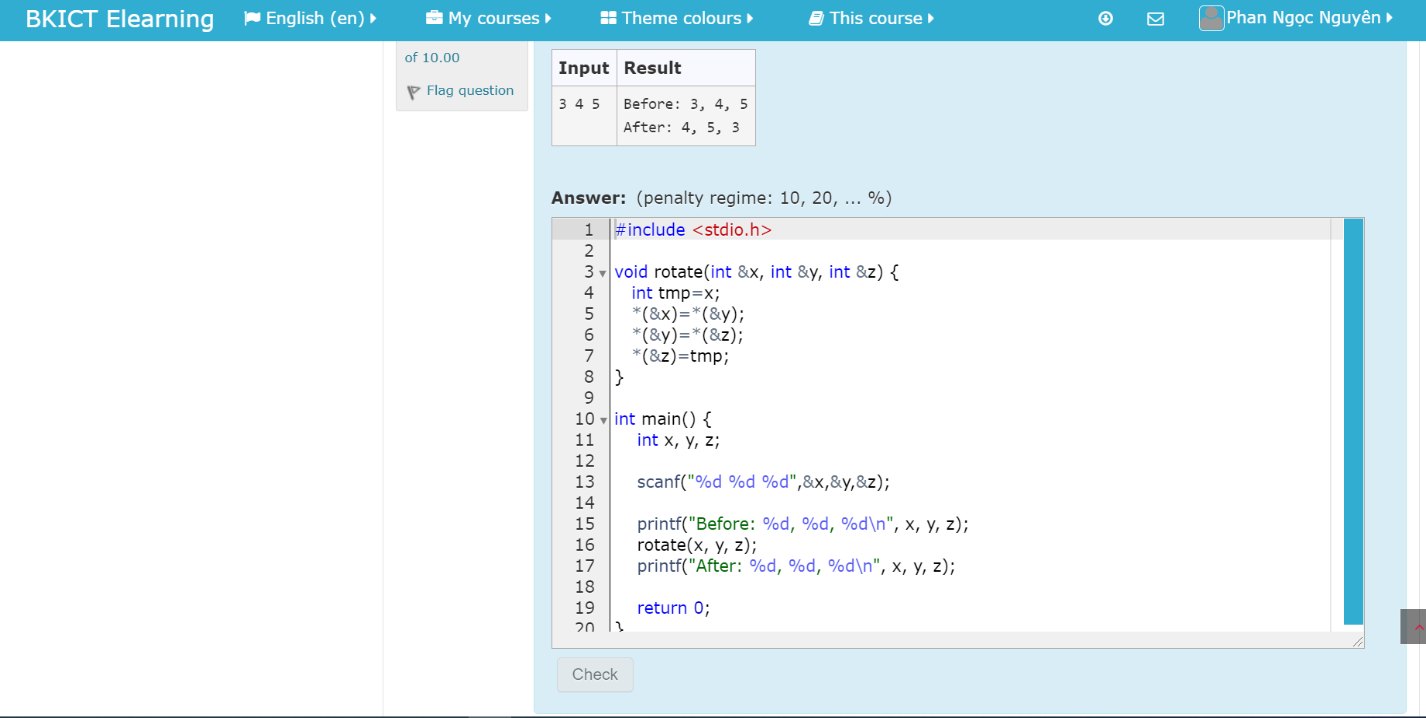
float z = get\_hypotenuse(x, y);

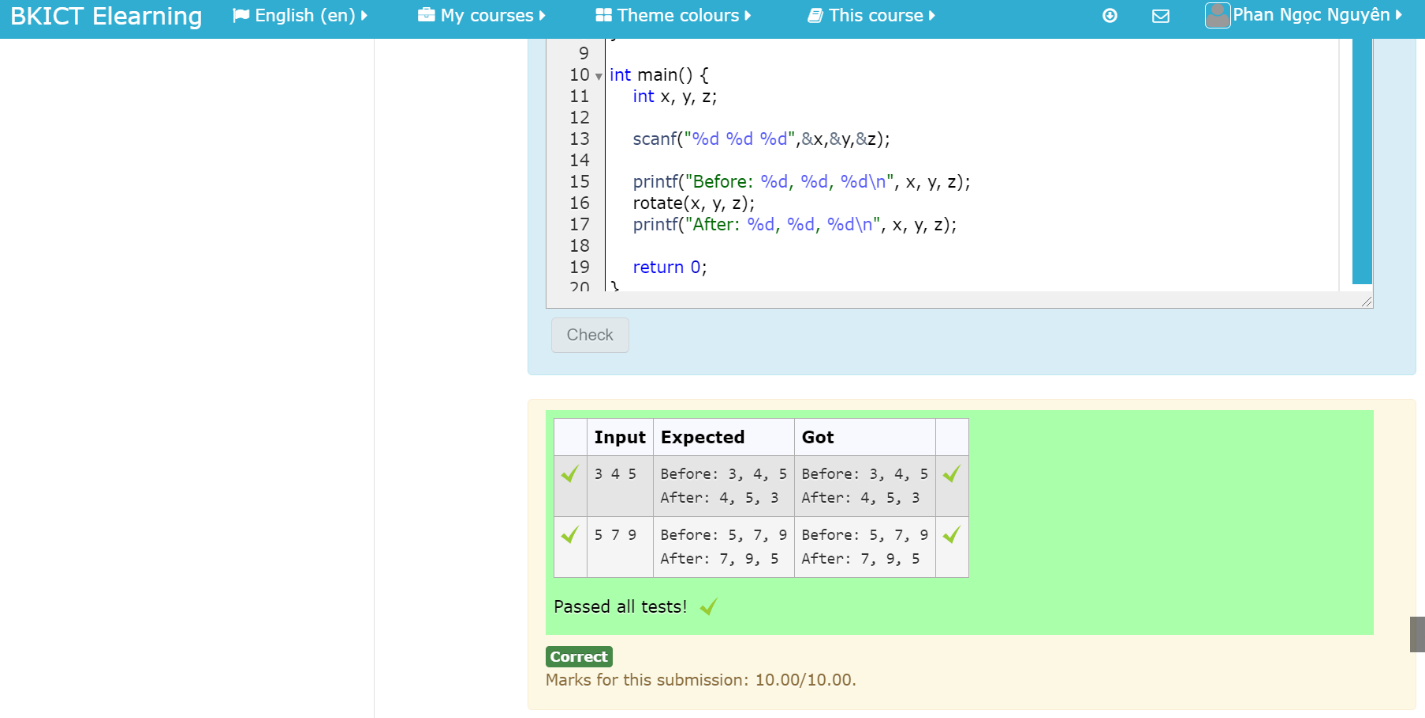
printf("z = %.2f\n", z);

return 0;

}

## Bài 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.





Code:

#include <stdio.h>

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

int a=x;

int b=y;

int c=z;

\*(&x)=b;

\*(&y)=c;

\*(&z)=a;

}

int main() {

int x, y, z;

printf("Phan Ngoc Nguyen-20194134\n");

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

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

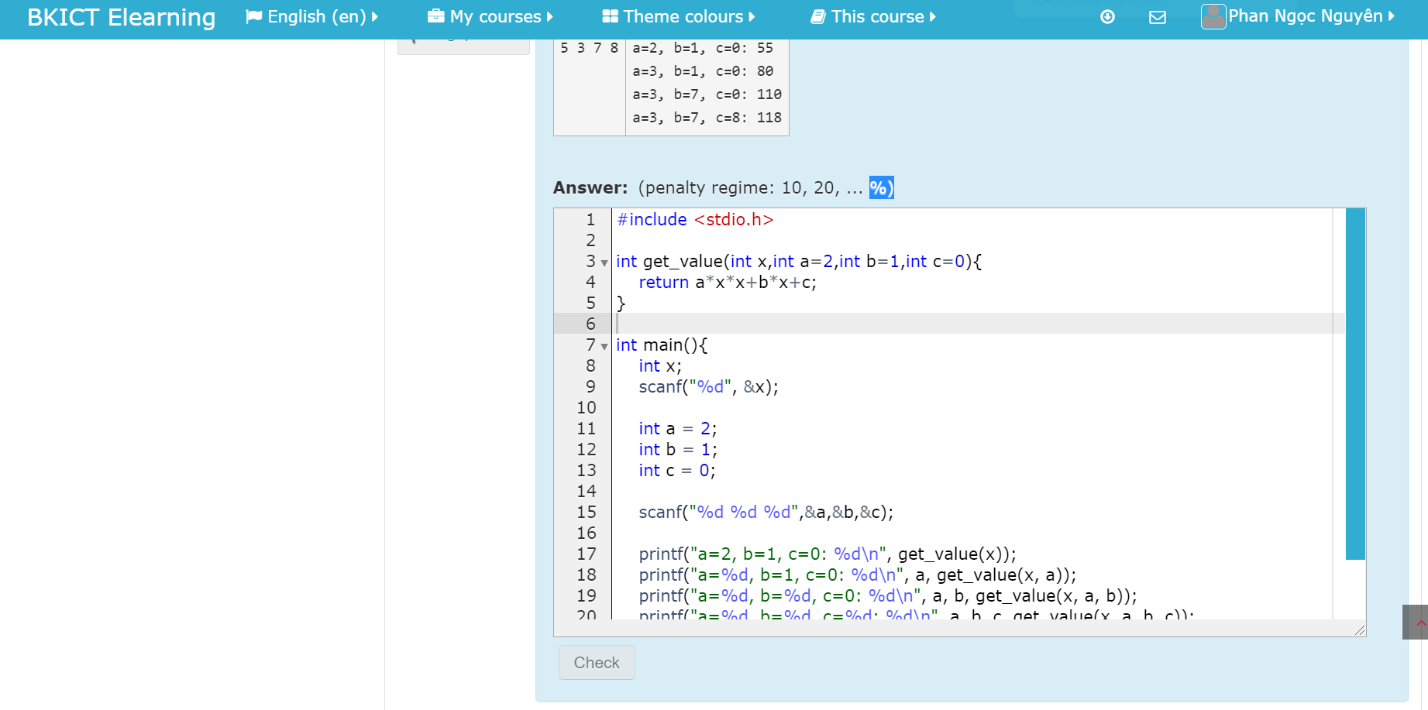
rotate(x, y, z);

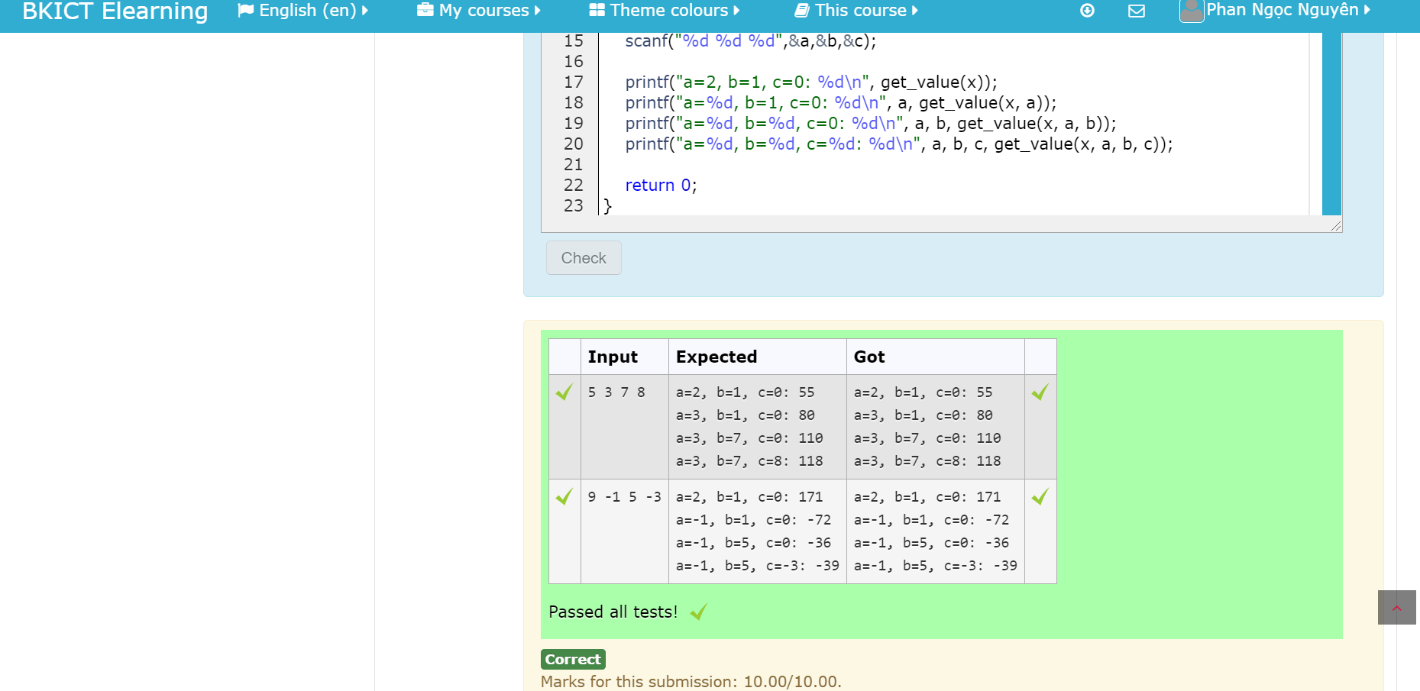
printf("After: %d, %d, %d\n", x, y, z);

return 0;

}

## Bài 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.





Code:

#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(){

printf("Phan Ngoc Nguyen-20194134\n");

int x;

scanf("%d", &x);

int a = 2;

int b = 1;

int c = 0;

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));

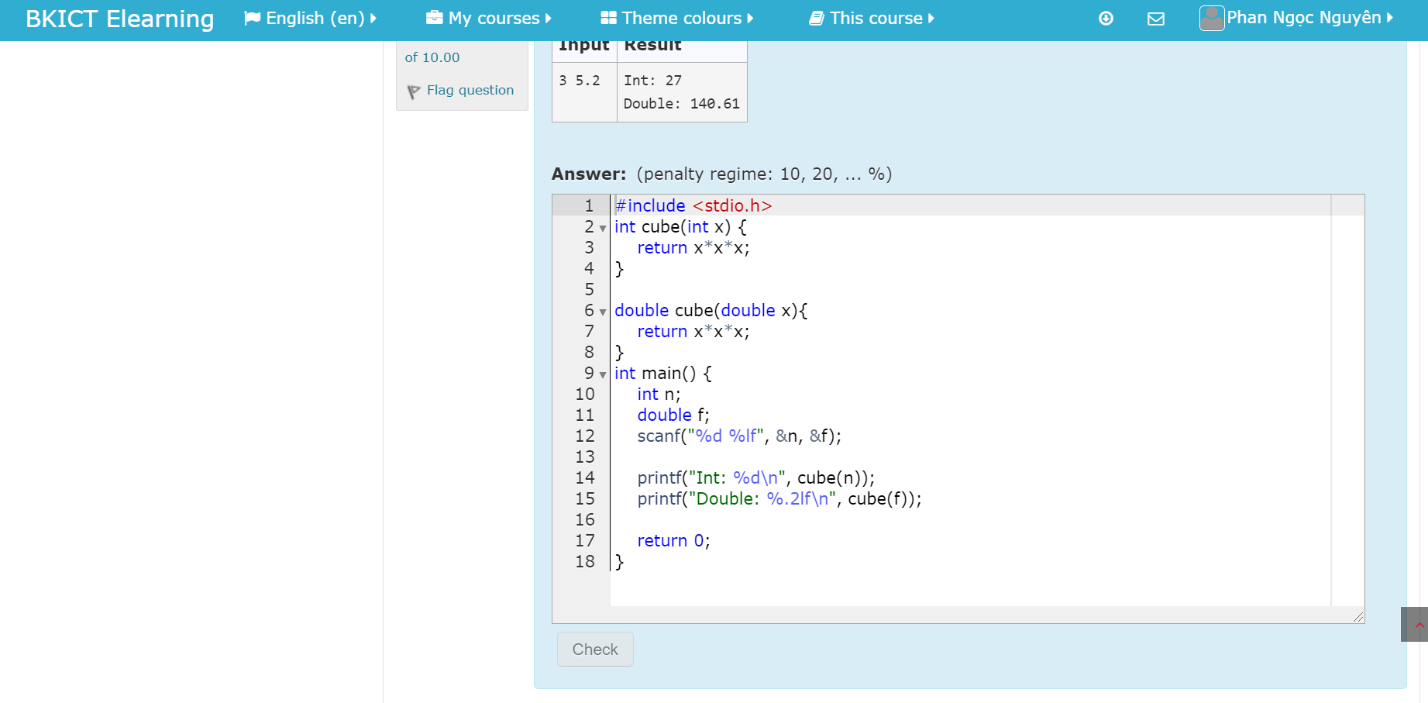
printf("a=%d, b=%d, c=0: %d\n", a, b, get\_value(x, a, b));

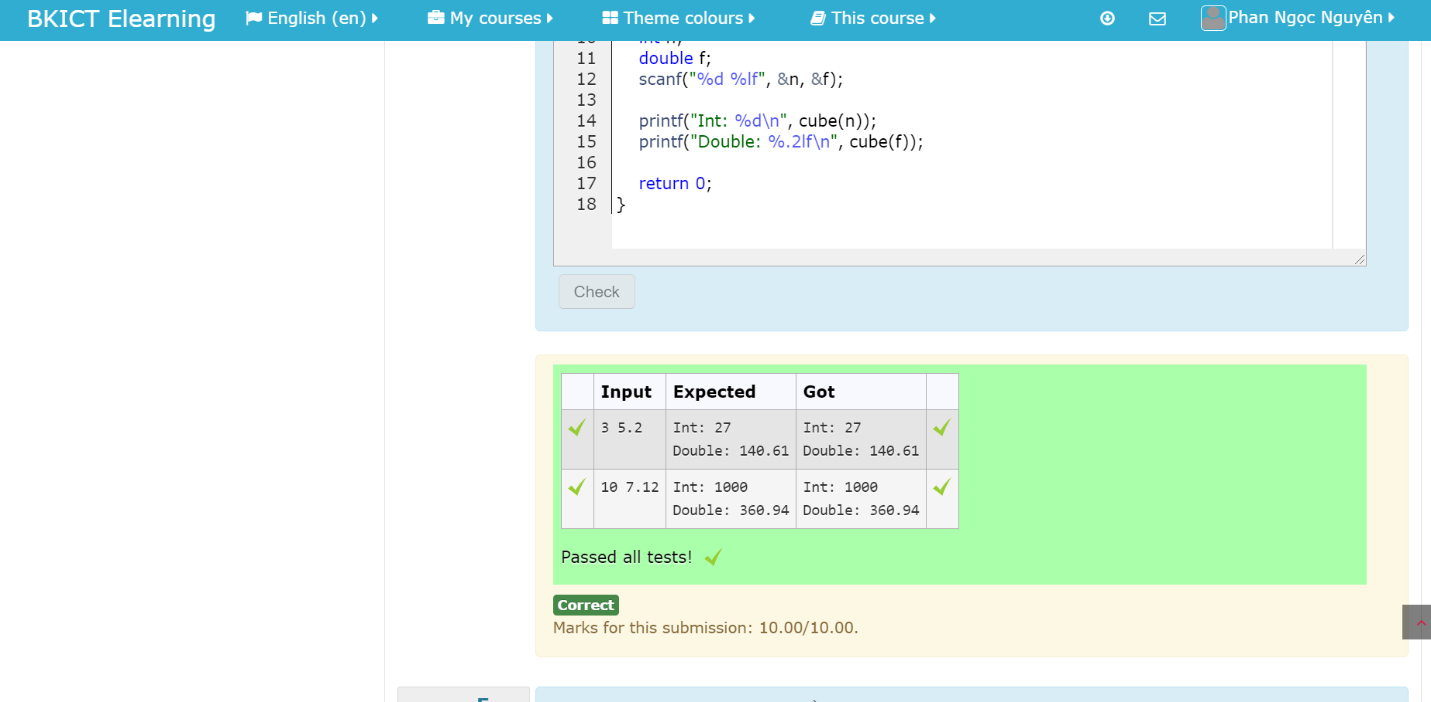
printf("a=%d, b=%d, c=%d: %d\n", a, b, c, get\_value(x, a, b, c));

return 0;

}

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





Code:

#include <stdio.h>

int cube(int x) {

return x\*x\*x;

}

double cube(double x){

return x\*x\*x;

}

int main() {

printf("Phan Ngoc Nguyen-20194134\n");

int n;

double f;

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

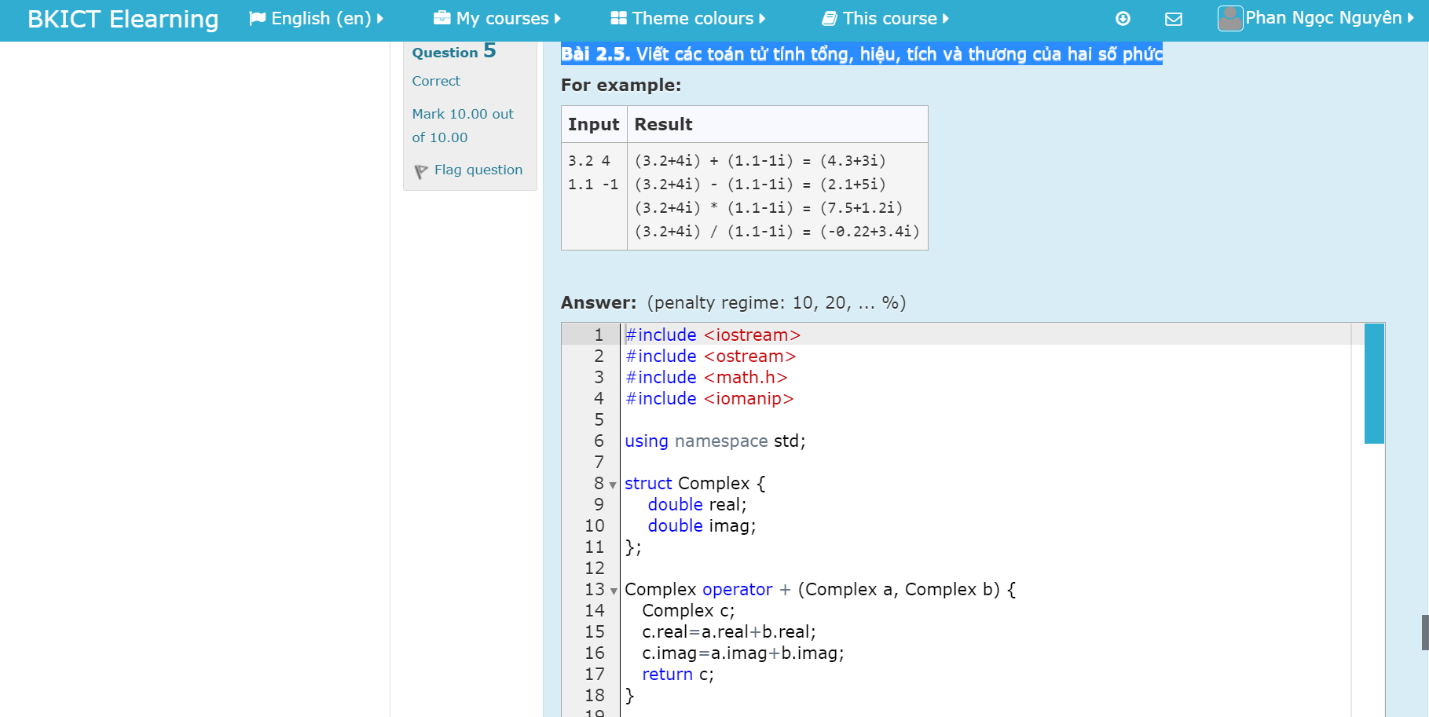
printf("Int: %d\n", cube(n));

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

return 0;

}

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





Code:

#include <iostream>

#include <ostream>

#include <math.h>

#include <iomanip>

using namespace std;

struct Complex {

double real;

double imag;

};

Complex operator + (Complex a, Complex b) {

Complex c;

c.real=a.real+b.real;

c.imag=a.imag+b.imag;

return c;

}

Complex operator - (Complex a, Complex b) {

Complex c;

c.real=a.real-b.real;

c.imag=a.imag-b.imag;

return c;

}

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

Complex c;

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

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

return c;

}

Complex operator / (Complex a, Complex b) {

Complex c;

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

c.imag=(b.real\*a.imag-a.real\*b.imag)/(b.real\*b.real+b.imag\*b.imag);

return c;

}

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() {

printf("Phan Ngoc Nguyen-20194134\n");

double real\_a, real\_b, img\_a, img\_b;

cin >> real\_a >> img\_a;

cin >> real\_b >> img\_b;

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

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

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

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

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

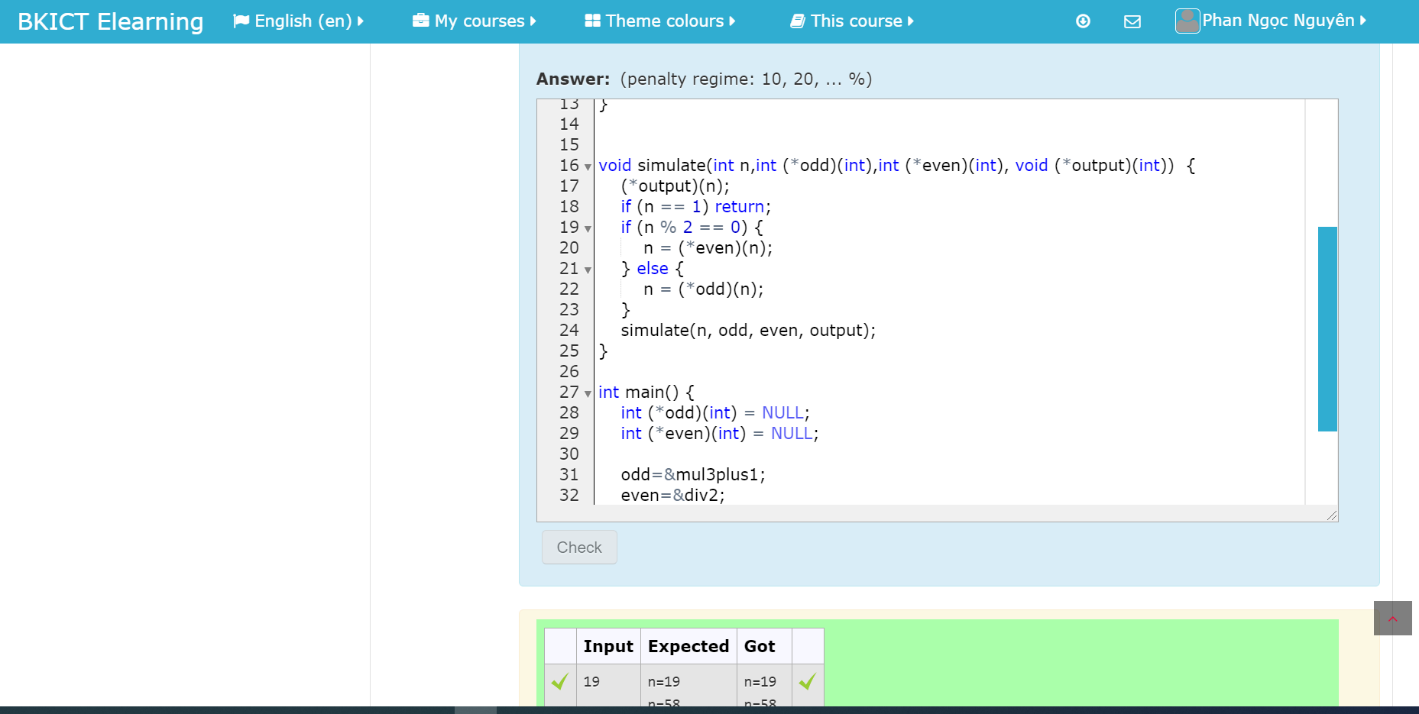
cout << a << " / " << b << " = " << a / b << endl;

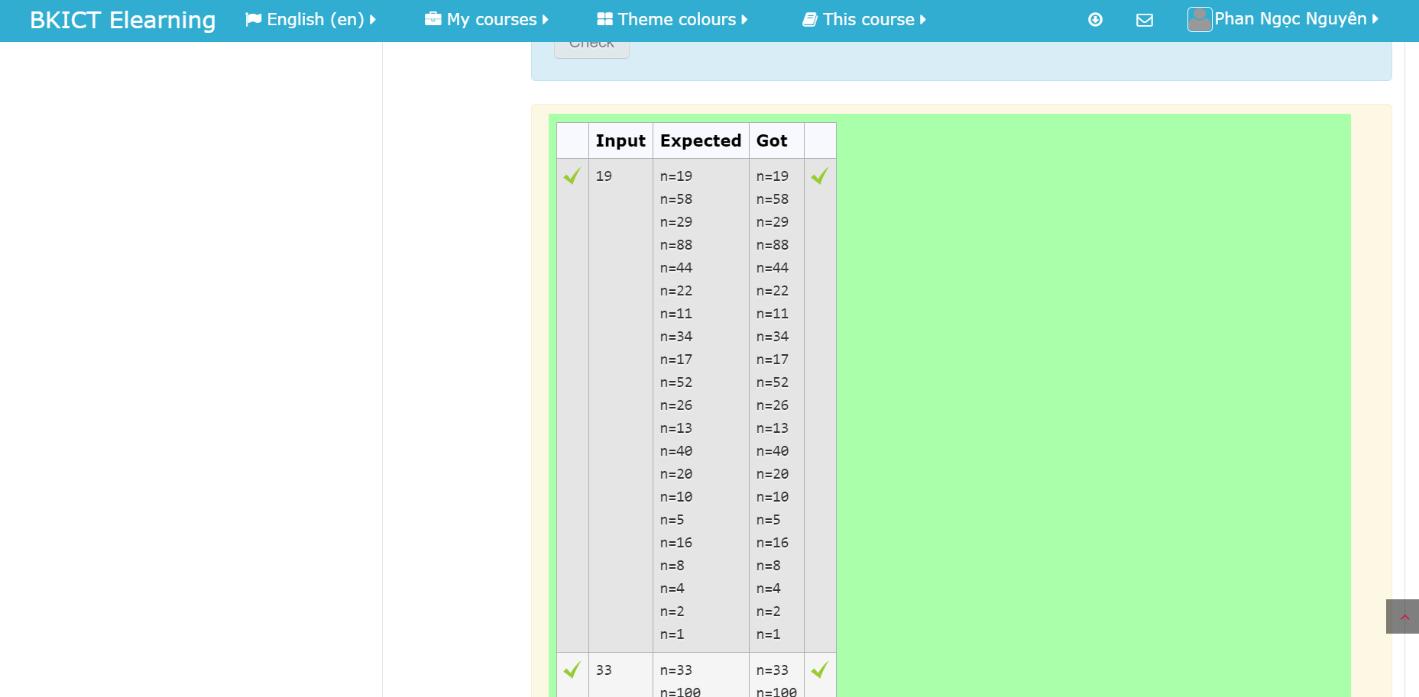
return 0;

}

## Bài 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.

## Hãy viết chương trình mô phỏng lại quá trình biến đổi để kiếm chứng giả thuyết với giá trị của n nhập từ bàn phím.

****



Code:

#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;

}

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() {

printf("Phan Ngoc Nguyen-20194134\n");

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

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

odd=&mul3plus1;

even=&div2;

int n;

scanf("%d", &n);

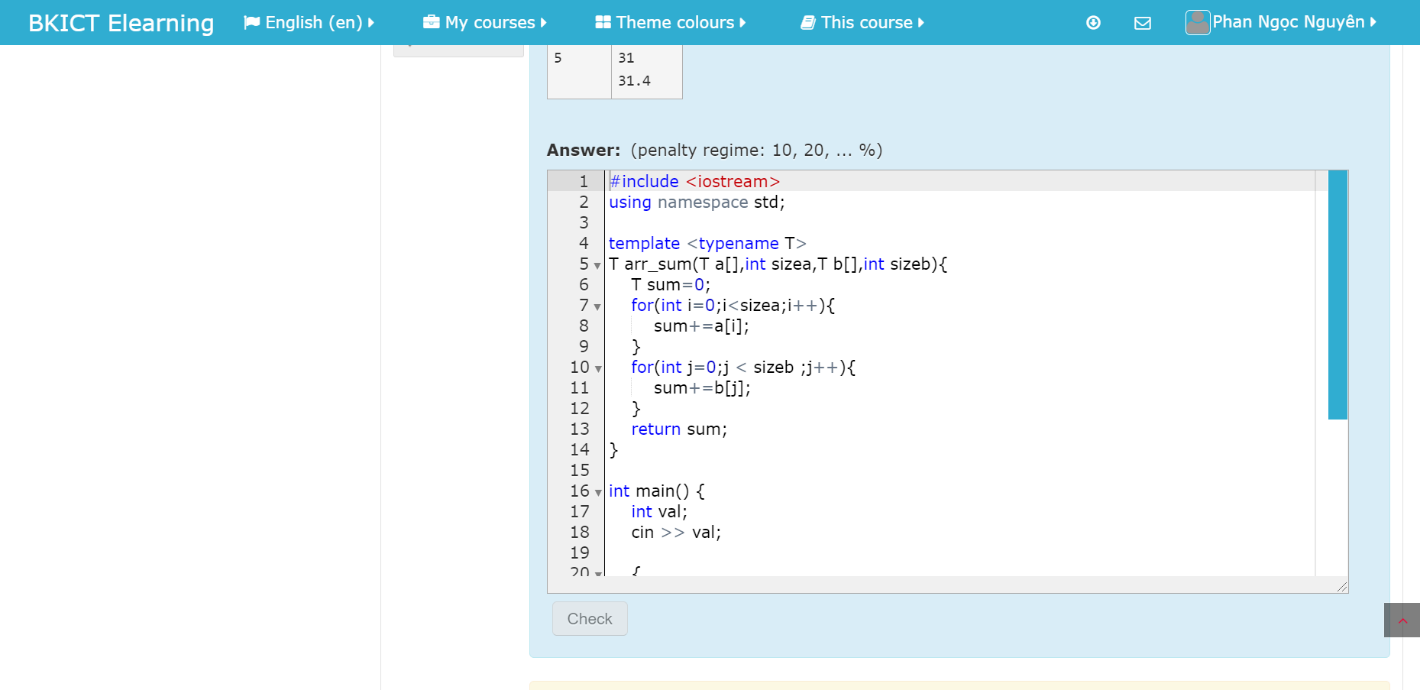
simulate(n, odd, even, print);

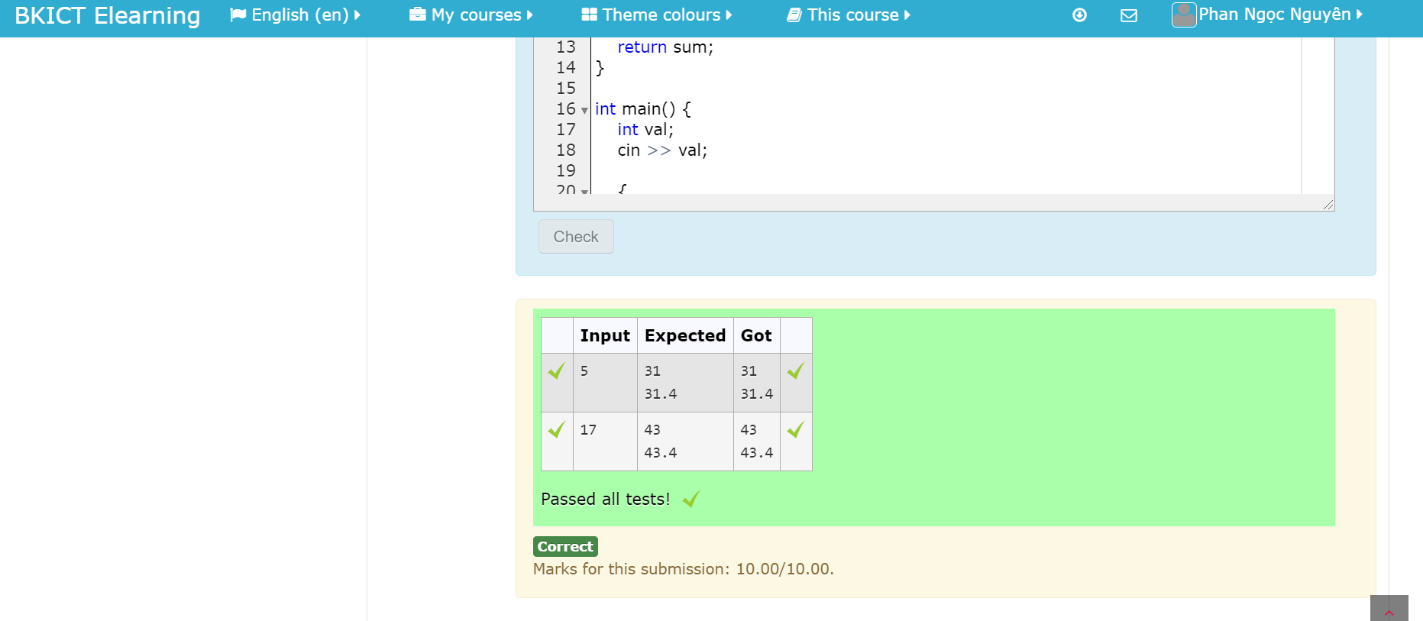
return 0;

}

## Bài 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.





Code:

#include <iostream>

using namespace std;

template <typename T>

T arr\_sum(T a[],int size\_a,T b[],int size\_b){

T sum=0;

for(int i=0;i<size\_a;i++){

sum+=a[i];

}

for(int j=0;j<size\_b;j++){

sum+=b[j];

}

return sum;

}

int main() {

printf("Phan Ngoc Nguyen-20194134\n");

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};

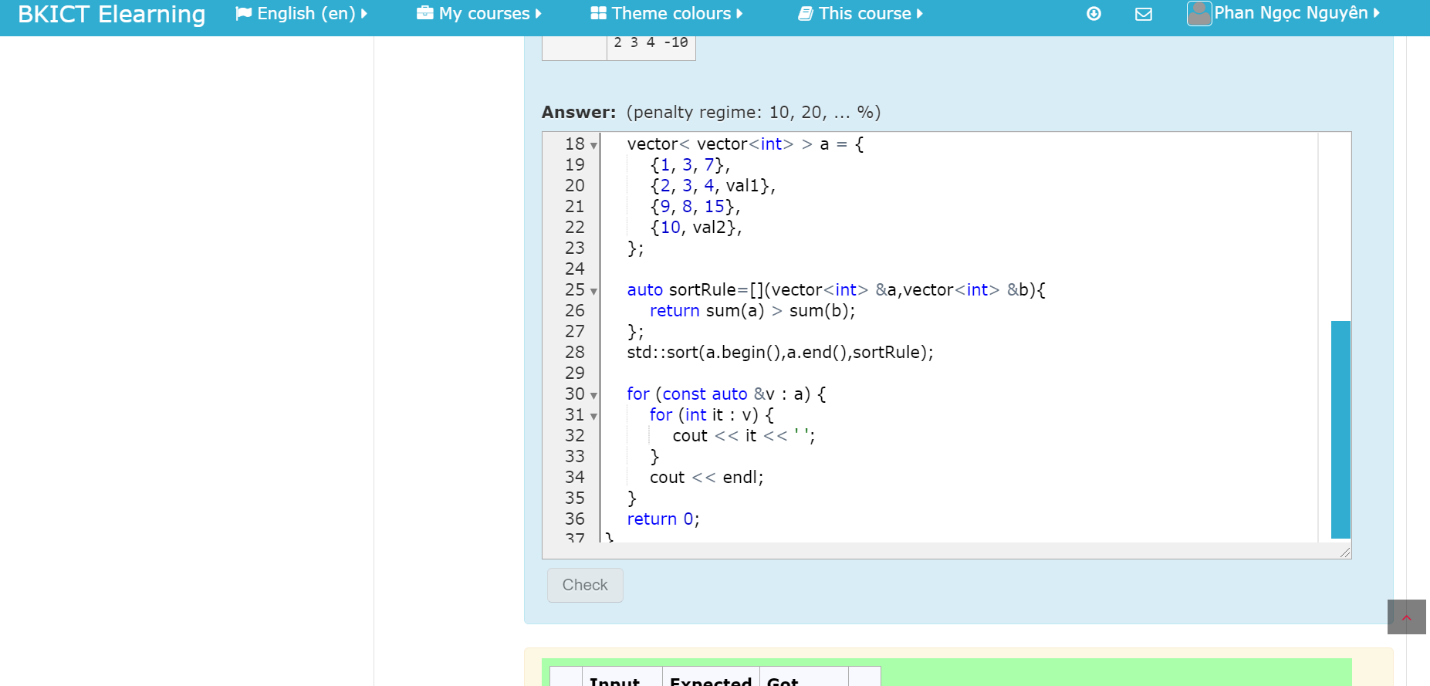
cout << arr\_sum(a, 4, b, 5) << endl;

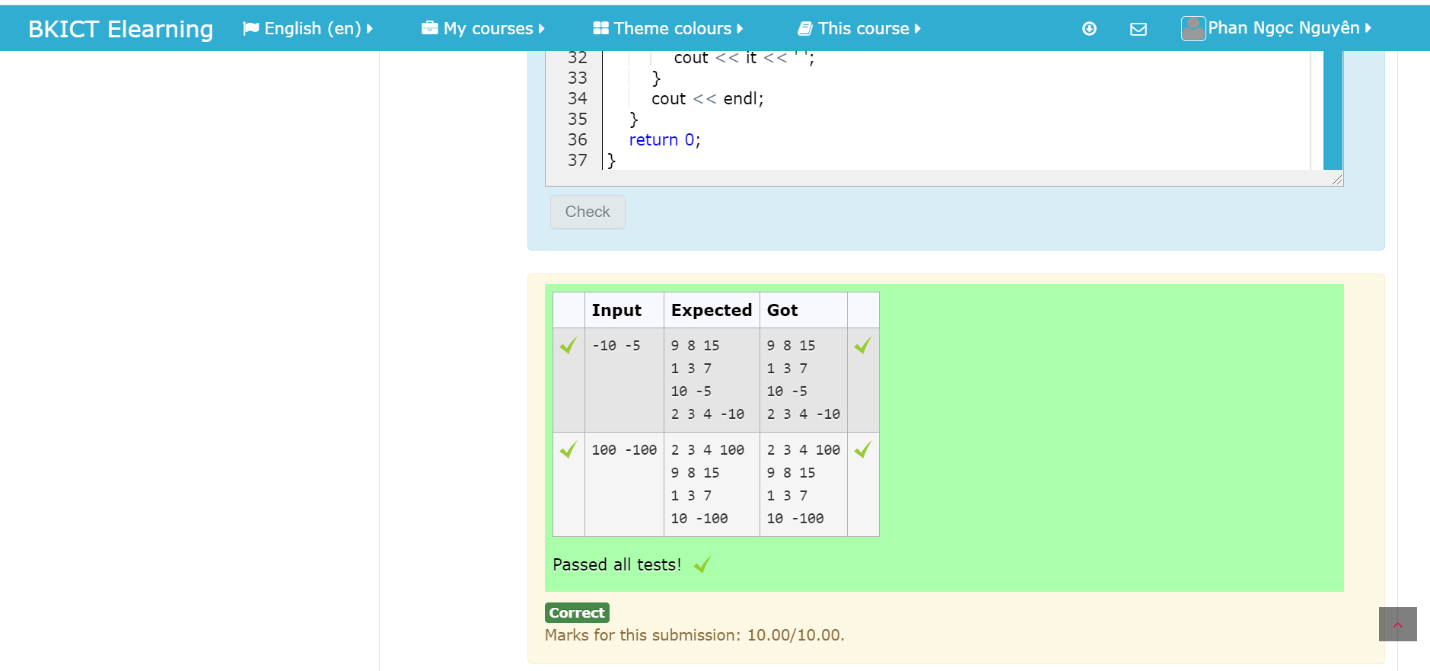
}

return 0;

}

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





#include <iostream>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

int sum(vector<int> x){

int sum=0;

for(int it:x){

sum+=it;

}

return sum;

}

int main() {

printf("Phan Ngoc Nguyen-20194134\n");

int val1, val2;

cin >> val1 >> val2;

vector< vector<int> > a = {

{1, 3, 7},

{2, 3, 4, val1},

{9, 8, 15},

{10, val2},

};

auto sortRule=[] (vector<int> const& x1,vector<int> const& x2){

return sum(x1) > sum(x2);

};

std::sort(a.begin(),a.end(),sortRule);

for (const auto &v : a) {

for (int it : v) {

cout << it << ' ';

}

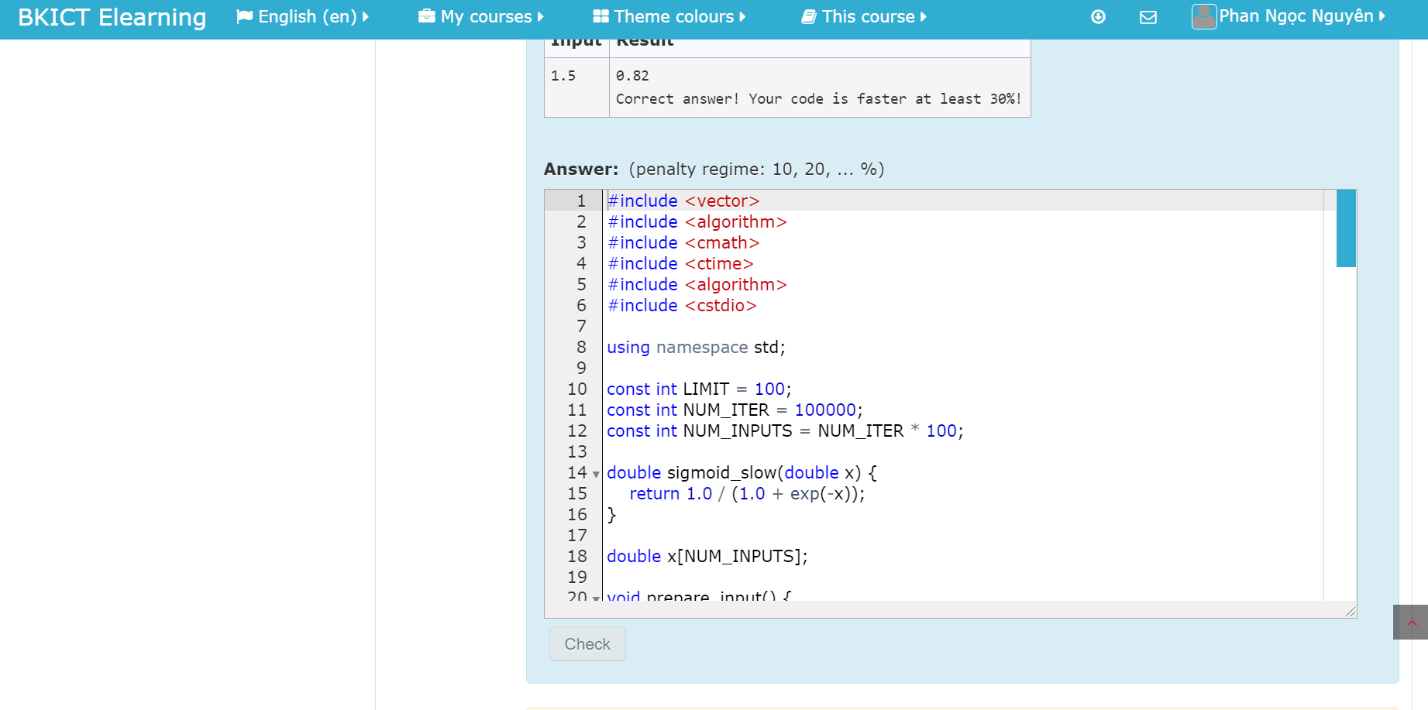
cout << endl;

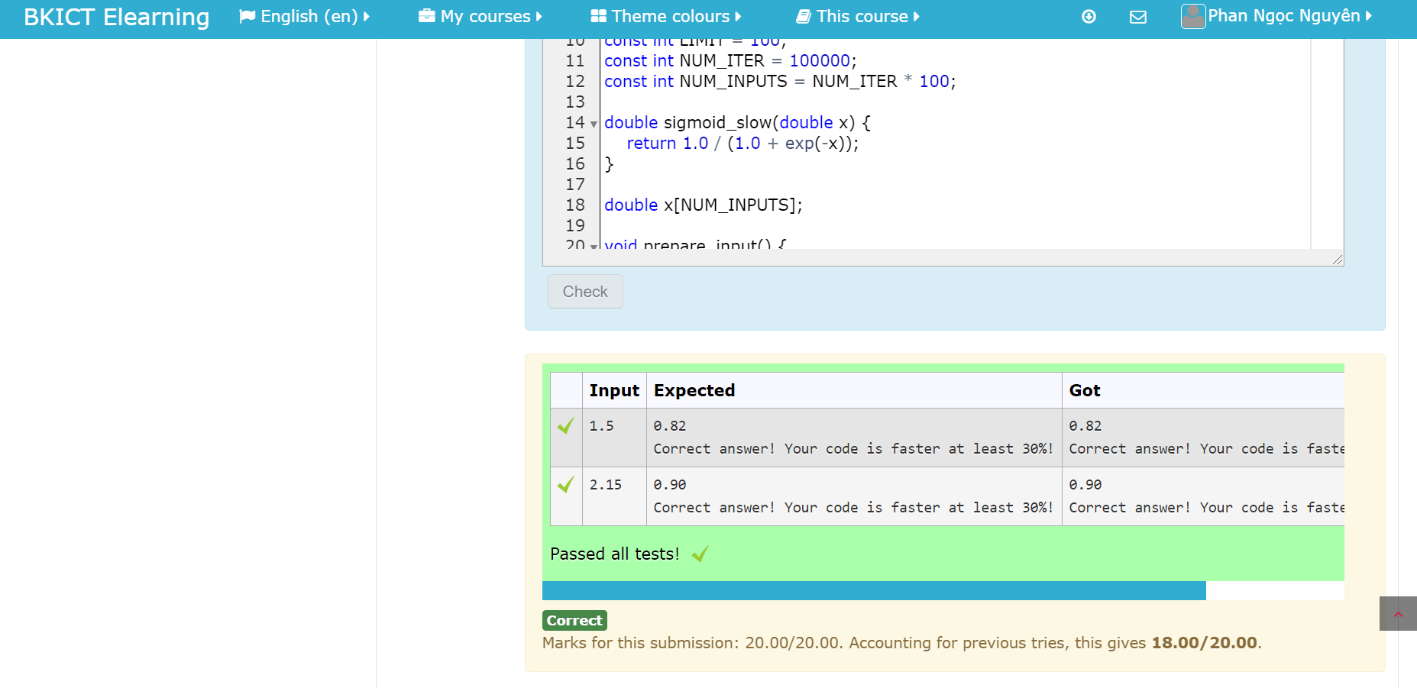
}

return 0;

}

## Bài 2.9. Tính hàm sigmoid





Code:

#include <vector>

#include <algorithm>

#include <cmath>

#include <ctime>

#include <algorithm>

#include <cstdio>

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;

}

}

const int N = 10000000;

double y[N];

void precalc() {

double delta = (LIMIT \* 2.0) / N;

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

y[i] = sigmoid\_slow(-LIMIT + delta \* i);

}

}

inline double sigmoid\_fast(double x) {

if(x <= -LIMIT) return 0;

if(x >= +LIMIT) return 1;

static double delta = (LIMIT \* 2.0)/N;

int i = (x + LIMIT) / delta;

return (x - (-LIMIT + (delta \* i))) / delta \* (y[i+1] - y[i]) + y[i];

}

//# 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);

//# printf("Time: %.9f\n", taken / CLOCKS\_PER\_SEC);

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() {

printf("Phan Ngoc Nguyen-20194134\n");

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("Wrong answer or your code is not fast enough!\n");

} else {

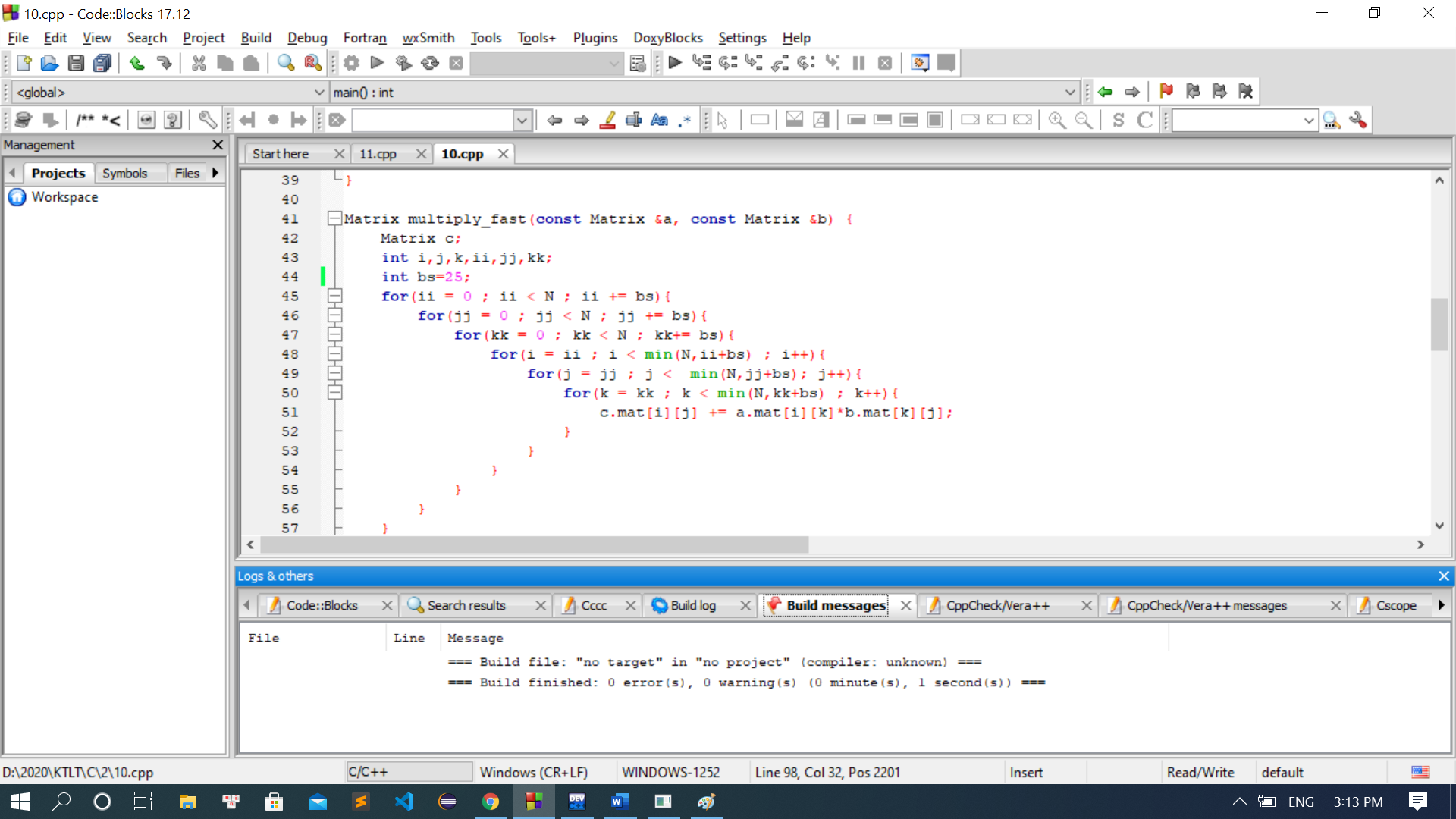
printf("Correct answer! Your code is faster at least 30%%!\n");

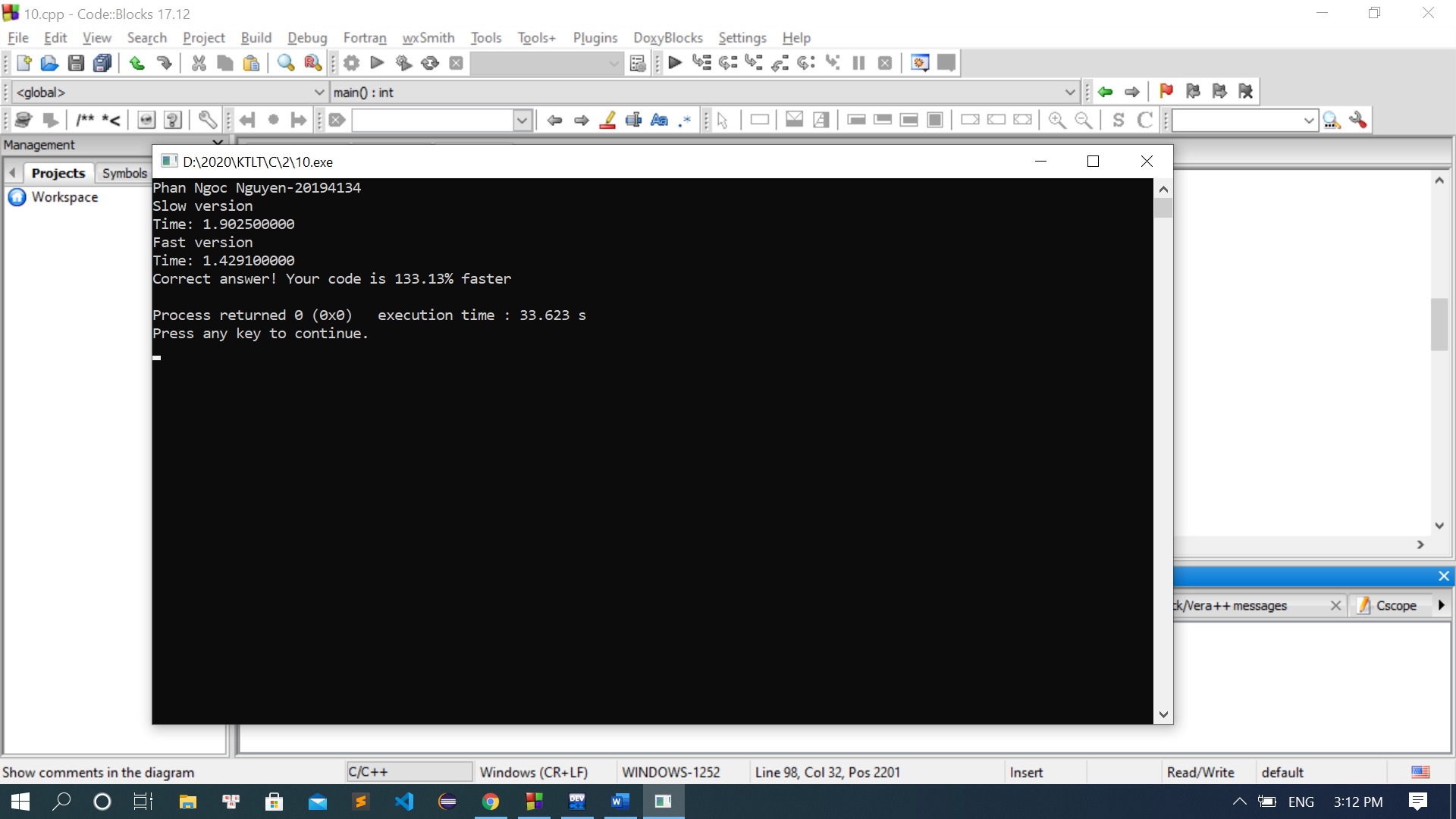
}

return 0;

}

## Bài tập 10 (bonus): Tính tích hai ma trận vuông





Code:

#include <iostream>

#include <cstring>

#include <time.h>

#include <math.h>

#define min(a,b) (((a)<(b))?(a):(b))

using namespace std;

const int N = 128;

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) {

Matrix c;

int i,j,k,ii,jj,kk;

int bs=25;

for(ii = 0 ; ii < N ; ii += bs){

for(jj = 0 ; jj < N ; jj += bs){

for(kk = 0 ; kk < N ; kk+= bs){

for(i = ii ; i < min(N,ii+bs) ; i++){

for(j = jj ; j < min(N,jj+bs); j++){

for(k = kk ; k < min(N,kk+bs) ; k++){

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

}

}

}

}

}

}

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;

printf("Time: %.9f\n", taken / CLOCKS\_PER\_SEC);

return taken;

}

int main() {

printf("Phan Ngoc Nguyen-20194134\n");

base = gen\_random\_matrix();

Matrix a, b;

printf("Slow version\n");

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

printf("Fast version\n");

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

if (a == b) {

printf("Correct answer! Your code is %.2f%% faster\n", slow / fast \* 100.0);

} else {

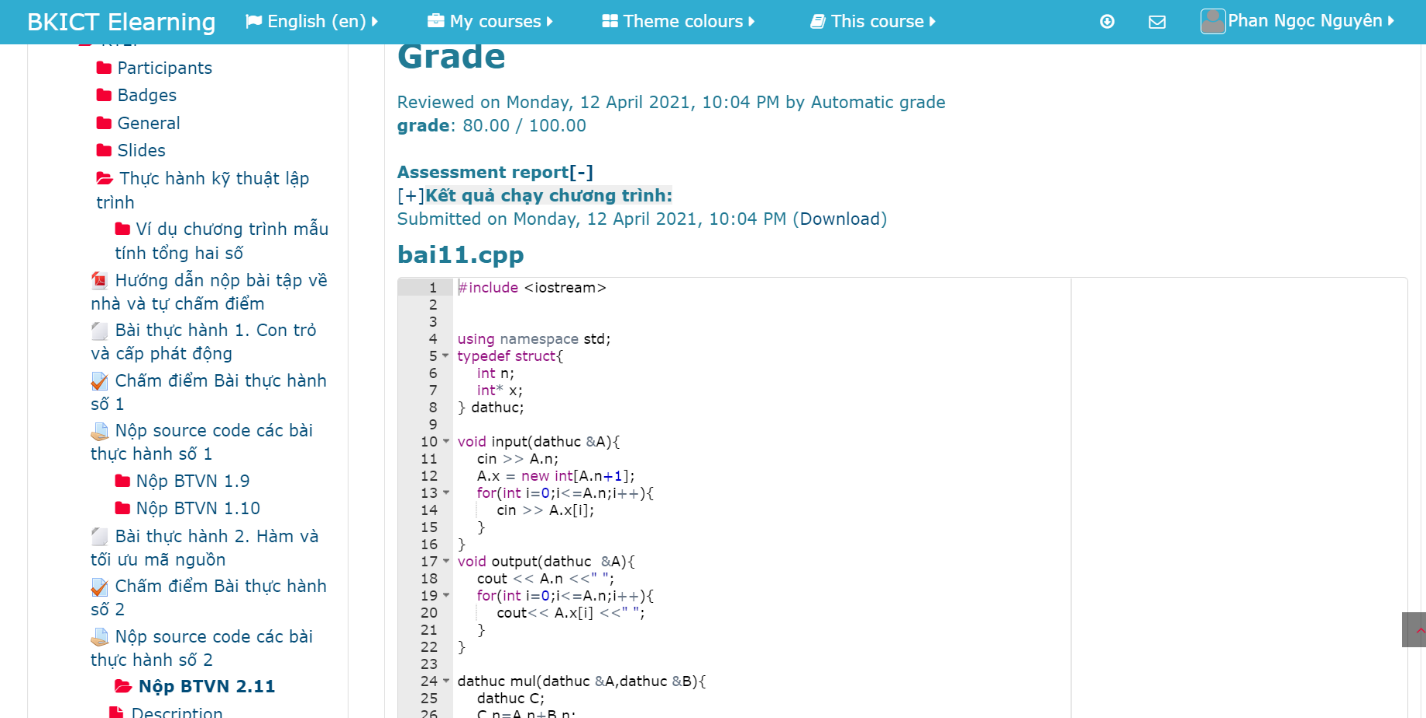
printf("Wrong answer!\n");

}

return 0;

## }

## Bài tập 11: Tính tích hai đa thức



Code:

#include <iostream>

using namespace std;

typedef struct{

int n;

int\* x;

} dathuc;

void input(dathuc &A){

cin >> A.n;

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

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

cin >> A.x[i];

}

}

void output(dathuc &A){

cout << A.n <<" ";

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

cout<< A.x[i] <<" ";

}

}

dathuc mul(dathuc &A,dathuc &B){

dathuc C;

C.n=A.n+B.n;

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

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

C.x[i]=0;

for(int j=0;j<=A.n;j++)

if(i>=j && i-j<=B.n) C.x[i]+=A.x[j]\*B.x[i-j];

}

return C;

}

int xorDt(dathuc &A){

int result=A.x[0];

for(int i=1;i<=A.n;i++){

result=result^A.x[i];

}

return result;

}

void deleteDt(dathuc &A){

delete[] A.x;

}

int main(){

cout<<"Phan Ngoc Nguyen-20194134"<<endl;

dathuc A,B,C;

input(A);

input(B);

C=mul(A,B);

cout<<xorDt(C)<<endl;

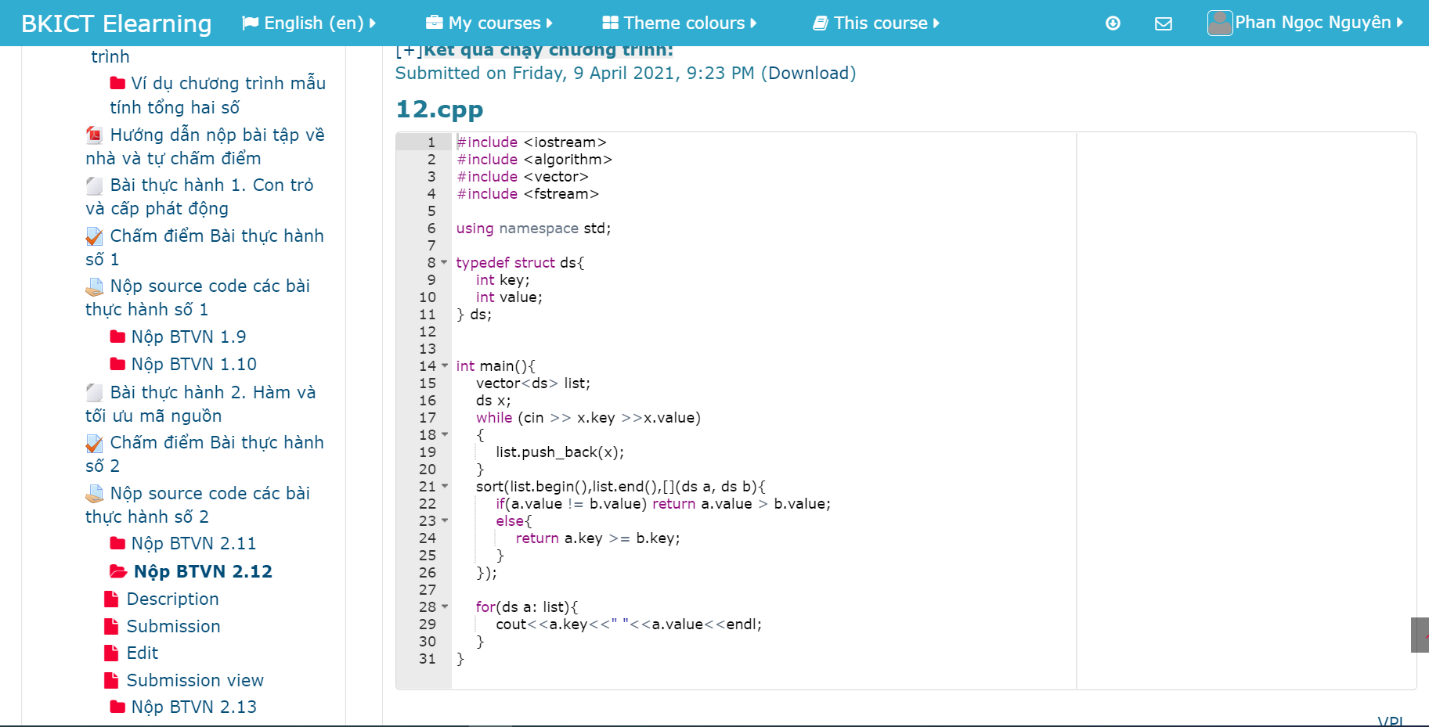
deleteDt(A);

deleteDt(B);

deleteDt(C);

}

## Bài tập 12: Map Sort



Code:

#include <iostream>

#include <algorithm>

#include <vector>

#include <fstream>

using namespace std;

typedef struct ds{

int key;

int value;

} ds;

int main(){

printf("Phan Ngoc Nguyen-20194134\n");

vector<ds> list;

ds x;

while (cin >> x.key >>x.value)

{

list.push\_back(x);

}

sort(list.begin(),list.end(),[](ds a, ds b){

if(a.value != b.value) return a.value > b.value;

else{

return a.key >= b.key;

}

});

for(ds a: list){

cout<<a.key<<" "<<a.value<<endl;

}

}

## Bài 13: Big Integer



Code:

#include<stdio.h>

#include<string.h>

#include <algorithm>

using namespace std;

typedef struct bigNum{

char sign;

char num[101];

}bigNum;

void swap1(bigNum &a,int size)

{

a.sign=a.num[0];

int r=size-1;

int l=0;

while(l<r)

{

swap(a.num[l],a.num[r]);

l++;r--;

}

a.num[size-1]='\0';

}

void swap2(bigNum &a,int size)

{

int r=size-1;

int l=0;

while(l<r)

{

swap(a.num[l],a.num[r]);

l++;r--;

}

}

bigNum operator + (bigNum a, bigNum b)

{

int test=0;

bigNum c;

int x=strlen(a.num);

int y=strlen(b.num);

int n=(x>y)?x:y;

a.sign=a.num[0];

b.sign=b.num[0];

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

if(x<y)

{

char temp[101];

strcpy(temp,a.num);

strcpy(a.num,b.num);

strcpy(b.num,temp);

test=1;

}

else{

char temp[101];

strcpy(temp,a.num);

temp[0]=b.num[0];

if(x==y&&(strcmp(b.num,temp)>0))

{

char temp2[101];

strcpy(temp2,a.num);

strcpy(a.num,b.num);

strcpy(b.num,temp2);

test=1;

}

}

}

if(test==1){

swap1(a,y);swap1(b,x);

}else{

swap1(a,x);swap1(b,y);

}

x=strlen(a.num)+1;

y=strlen(b.num)+1;

if(x!=y){

if(x<y)

{

for(int i=x-1;i<y-1;i++)

{

a.num[i]='0';

}

}else{

for(int i=y-1;i<x-1;i++)

{

b.num[i]='0';

}

}}

a.num[n-1]='0';

b.num[n-1]='0';

int i=0;

int k=0;

// printf("\n%s",a.num);

// printf("\n%s",b.num);

if(a.sign==b.sign)

{

while(i<n)

{

int soa=((int)a.num[i])-48;

int sob=((int)b.num[i])-48;

int sok=soa+sob+k;

c.num[i]=(char)(48+sok%10);

k=sok/10;

i++;

}

if(c.num[n-1]=='0')

{

swap2(c,n);

c.num[0]=a.sign;

c.num[n]='\0';

}

else{

if(c.num[n-1]=='1')

{

c.num[n]=a.sign;

c.num[n+1]='\0';

swap2(c,n+1);

}}

}else{

while(i<n)

{

int soa=((int)a.num[i])-48;

int sob=((int)b.num[i])-48;

int sok=soa-sob-k;

if(sok>=0)

{

c.num[i]=char(48+sok);k=0;

}else{

c.num[i]=char(58+sok);

k=1;

}

i++;

}

// printf("\n%s",c.num);

if(test==1)

{

if(a.sign=='1'){

c.num[n]='1';}

else{

if(a.sign=='0'){

c.num[n]='0';

}

}

}

else{

if(a.sign=='1')

c.num[n]='1';

else {

if(a.sign=='0')

{

c.num[n]='0';

}}

}

c.num[n+1]='\0';

swap2(c,n+1);

}

while(c.num[1]=='0'){

for(int i=1;i<strlen(c.num);i++)

{

c.num[i]=c.num[i+1];

}

c.num[strlen(c.num)]='\0';

}

return c;

}

void mabu1(bigNum a,int size)

{

for(int i=1;i<size;i++)

{

a.num[i]=(char)(105-(int)a.num[i]);

}

a.num[0]='1';

}

bigNum operator - (bigNum a, bigNum b)

{

a.sign=a.num[0];

b.sign=b.num[0];

if(b.sign=='1')

{

b.sign='0';

b.num[0]='0';

return a+b;

}else{

b.sign='1';

b.num[0]='1';

return a+b;

}

}

void dichbit(bigNum &a,int n){

int x=strlen(a.num);

int i=0;

while(i<n)

{

a.num[x+i]='0';

i++;

}

a.num[x+n]='\0';

}

bigNum nhan(bigNum a, int n)

{

int i=0;int k=0;

a.sign=a.num[0];

int x=strlen(a.num);

swap1(a,x);

bigNum c;

while(i<x-1)

{

int soa=((int)a.num[i]-48)\*n;

int k1=soa+k;

int k2=k1%10;

c.num[i]=char(48+k2);

k=k1/10;

i++;

}

c.num[x-1]=char(48+k);

c.num[x]=a.sign;

swap2(c,x+1);

c.num[x+1]='\0';

return c;

}

bigNum operator \* (bigNum a, bigNum b)

{

bigNum c;

c.num[0]='1';c.num[1]='0';c.num[2]='\0';

a.sign=a.num[0];

b.sign=b.num[0];

char x=a.sign;

char y=b.sign;

a.num[0]='1';b.num[0]='1';

a.sign='1';b.sign='1';

int m=strlen(b.num);

int i=0;

while(i<m-1)

{

int n=(int)b.num[m-1-i]-48;

bigNum d = nhan(a,n);

dichbit(d,i);

c=c+d;

i++;

}

if(x==y)

{

c.num[0]='1';

}

else{

c.num[0]='0';

}

return c;

}

int main()

{

printf("Phan Ngoc Nguyen- 20194134\n");

bigNum x,y,d,e,c;

gets(x.num);

fflush(stdin);

gets(y.num);

c=x\*y;

d=nhan(x,3);

e=nhan(y,4);

c=x\*y-d+e;

printf("%s",c.num);

return 0;

}