Mục lục nội dung

Bài thực hành 3: Đệ quy và khử đệ quy để giải quyết một số bài toái

Phần 1. Thực hành về đệ quy

1.1 Đệ quy - quay lui.

Bài tập 1: Tính dãy Lucaa

Dãy Lucas được định nghĩa bởi L n=1+Ln=2Ln=Ln=1#Ln~2 và bắt đầu bởi L0=2L0=2, L1=1L1=1.

Viết hàm tính số Lucas thứ nn.

Bài tập 2: Quân mã đi tuần

“Trên bàn cờ vua kích thước nxnnxn có một quân mã đang ở ô (1, 1). Hãy đưa ra một đãy các di chuyển của mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần (ô (1, 1) được xem là đã đi qua)

1.2 Kỹ thuật nhánh cận

Bài tập 3: Bài toán người du lịch

Một người xuất phát tại thành phố 1, muốn đi thăm tất cả các thành phố khác, mỗi thành phố đúng 1 lần và quay về 1. Chỉ phí để đi từ thành phố ii sang thành phố jj là ci,jci,j. Hãy tìm tổng chỉ phí nhỏ nhất có thể.

1.3 Đệ quy có nhớ

Bài tập 4: LIS

Cho dãy aa có nn phần tử. Một dãy con của aa là dấy thu được bằng cách xóa đi một số phần tử của aa và giữ nguyên thứ tự các phần tử còn lại (có thể không xóa phần tử nào). Hãy tìm dãy con tăng dài nhất của aa

Phần 2. Khử đệ quy.

Bài tập 5: Tính tổ hợp.

Tính CknCnk

Bài tập 6: Tìm ước chung lớn nhất.

Tính ước chung lớn nhất của hai số cho trước.

Bài tập 7: Liệt kê xâu nhị phân.

Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân độ dài nn không có kk bit 1 nào liên tiếp

Bài tập 8: Cân đĩa.

Bạn đang muốn kiểm tra xem một vật cho trước có đúng nặng MM như người ta nói hay không. Có một cân thăng bằng và nn quả cân. Quả thứ ïi năng mimi. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã ngưồn dưới

Phần 3. Bài tập về nhà

Mục lục hình ảnh

[Hình 1 Bài 3.1 Viết hàm tính số Lucas thứ n 3](#_Toc161238452)

[Hình 2 Bài 3.2 Viết hàm đưa ra một dãy các di chuyển của mã trên bàn cờ vua 5](#_Toc161238453)

[Hình 3 Bài 3.3 Viết chương trình tính tổng chi phí nhỏ nhất có thể 7](#_Toc161238454)

[Hình 4 Bài 3.4 Viết hàm tìm dãy con dài nhất của a 9](#_Toc161238455)

[Hình 5 Bài 3.5 Tính hệ số tổ hợp C (n,k) 11](#_Toc161238456)

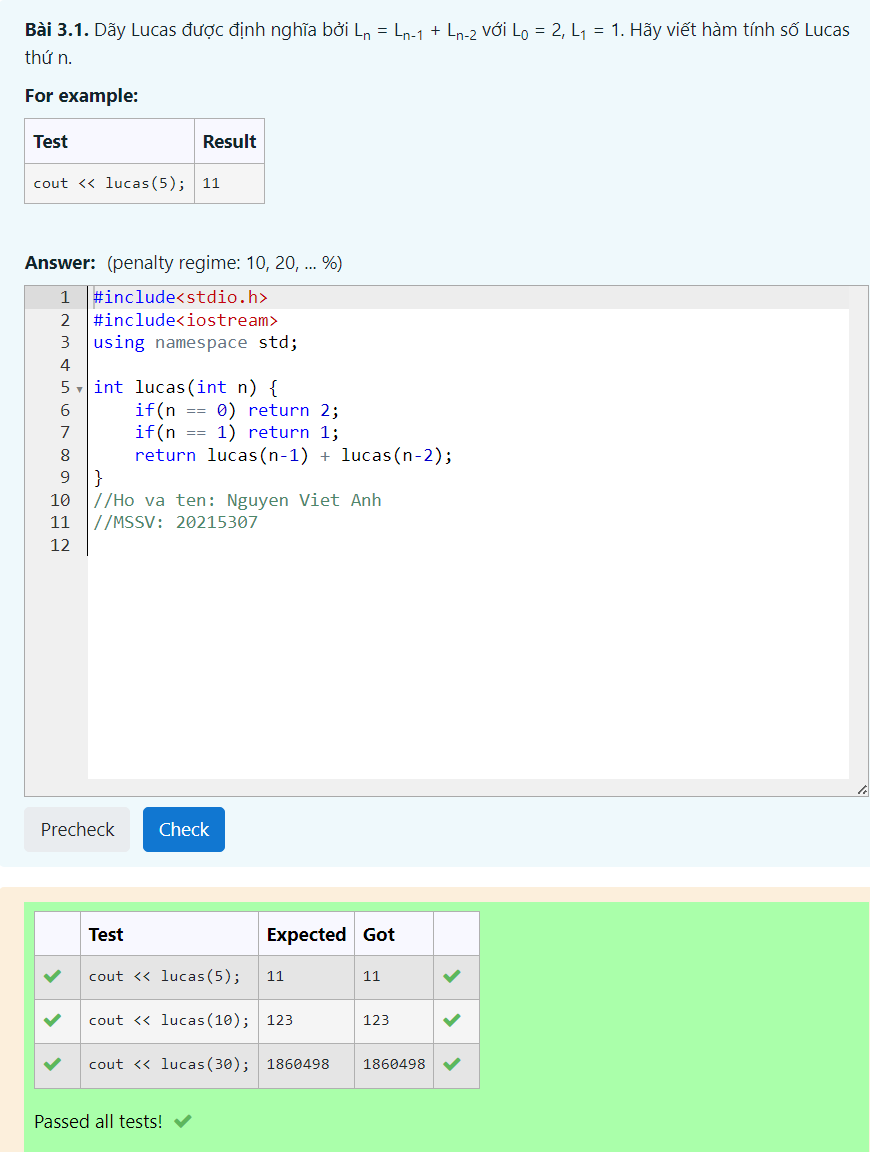
[Hình 6 Bài 3.6 Tìm ước chung lớn nhất của 2 số nguyên a,b cho trước 15](#_Toc161238457)

[Hình 7 Bài 3.7 Viết hàm liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp…. 18](#_Toc161238458)

[Hình 8 Bài 3.8 Viết hàm kiểm tra quả cân 21](#_Toc161238459)

# Bài thực hành số 2 – Tuần 11

# BÀI TẬP TRÊN LAP



Hình Bài 3.1: Viết hàm tính số Lucas thứ n

#include<stdio.h>

#include<iostream>

using namespace std;

int lucas(int n) {

if(n == 0) return 2;

if(n == 1) return 1;

return lucas(n-1) + lucas(n-2);

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

int main(){

int nn;

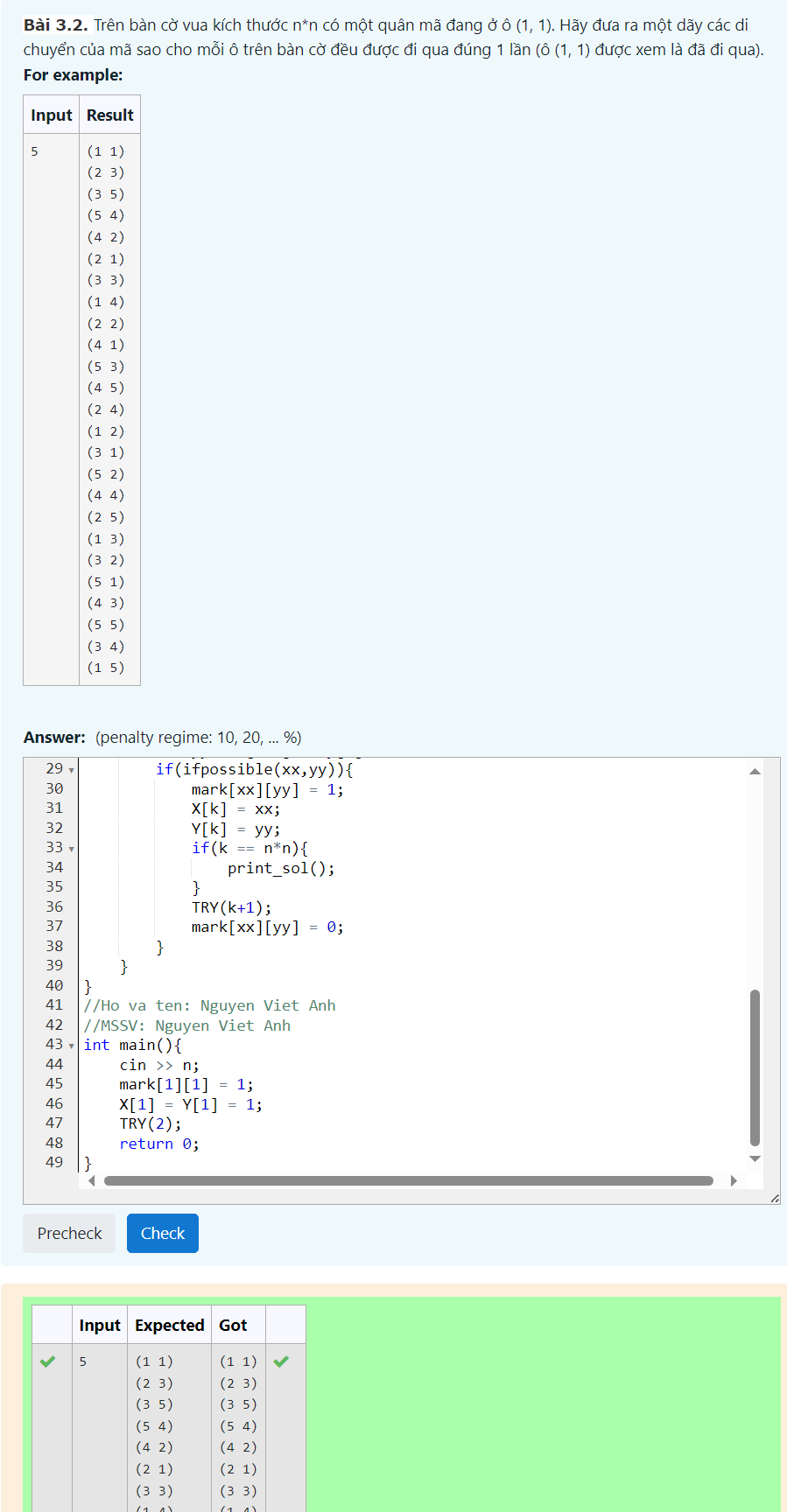
scanf ("%d", &nn);

printf ("Ho va ten: Nguyen Viet Anh\n");

printf ("MSSV:20215307\n");

printf ("%d", lucas(nn));

}



Hình Bài 3.2: Viết hàm đưa ra dãy di chuyển của mã trên bàn cờ vua

#include <iostream>

using namespace std;

int n;

int X[100], Y[100]; //# Luu vi tri các buoc di chuyen cua quân ma

int mark[100][100]; //# Ðánh dau vi tri các ô mà quân mã dã di chuyen qua

//# Mang hx, hy mô ta 8 vi tri quân mã có the di chuyen tu vi tri hien tai

const int hx[] = {1, 1, 2, 2, -1, -1, -2, -2};

const int hy[] = {2, -2, 1, -1, 2, -2, 1, -1};

//# In ra day các buoc di chuyen tim dc

void print\_sol(){

for (int j = 1; j <= n \* n; ++j)

printf("(%d %d)\n", X[j], Y[j]);

exit(0);

}

bool ifpossible(int x, int y){

if(x < 1 || x > n) return false;

if(y < 1 || y > n) return false;

if(mark[x][y] == 1) return false;

return true;

}

//# Thuat toán quay lui

void TRY(int k){

for(int i = 0; i < 8; i++){

int xx = X[k-1] + hx[i];

int yy = Y[k-1] + hy[i];

if(ifpossible(xx,yy)){

mark[xx][yy] = 1;

X[k] = xx;

Y[k] = yy;

if(k == n\*n){

print\_sol();

}

TRY(k+1);

mark[xx][yy] = 0;

}

}

}

//Ho va ten: Nguyen Viet Anh

//MSSV: Nguyen Viet Anh

int main(){

printf ("Ho va ten: Nguyen Viet Anh\n");

printf ("MSSV 20215307\n");

cin >> n;

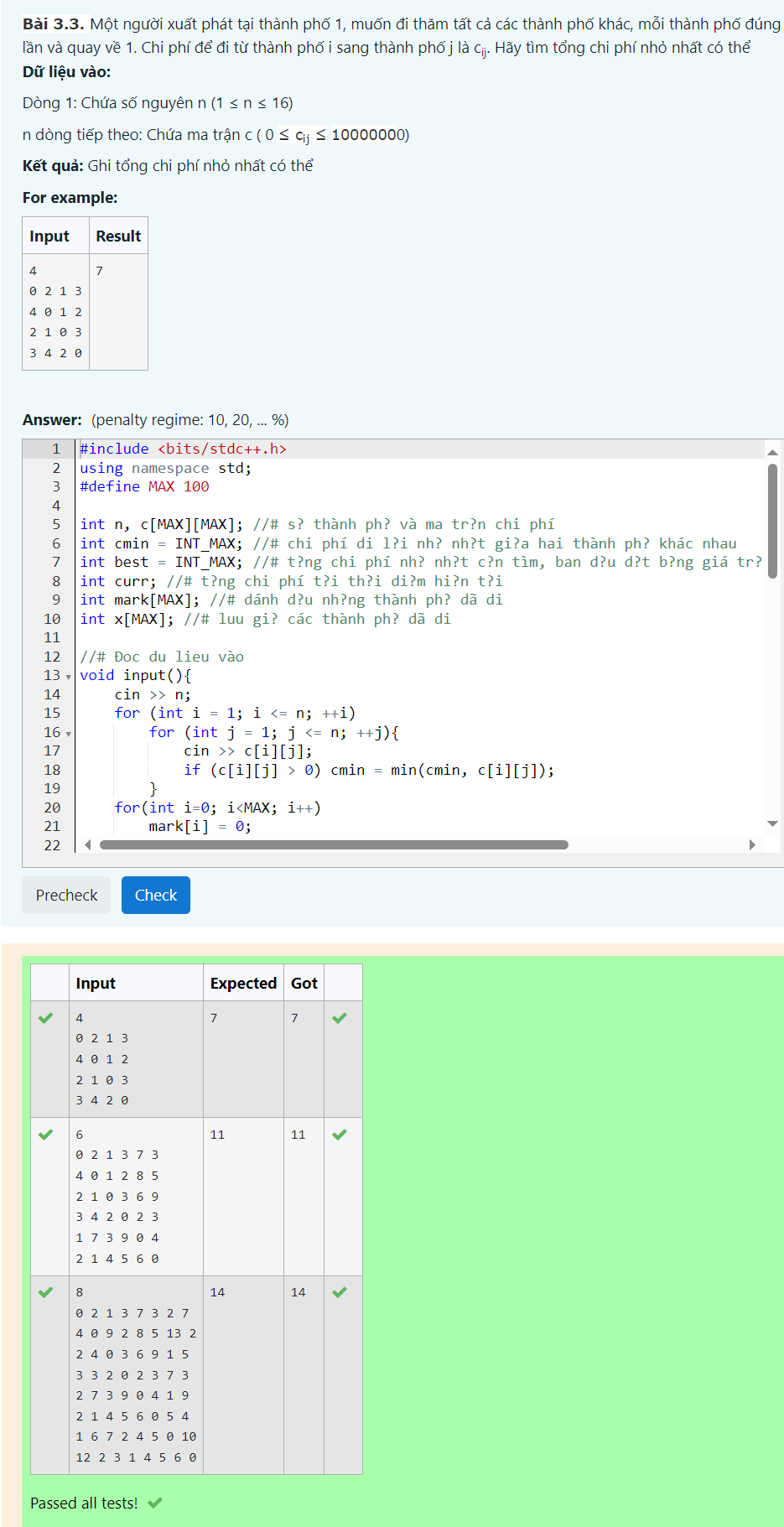
mark[1][1] = 1;

X[1] = Y[1] = 1;

TRY(2);

return 0;

}



Hình Bài 2.3 Viết chương trình tính tổng chi phí nhỏ nhất có thể

#include <bits/stdc++.h>

using namespace std;

#define MAX 100

int n, c[MAX][MAX]; //# s? thành ph? và ma tr?n chi phí

int cmin = INT\_MAX; //# chi phí di l?i nh? nh?t gi?a hai thành ph? khác nhau

int best = INT\_MAX; //# t?ng chi phí nh? nh?t c?n tìm, ban d?u d?t b?ng giá tr? vô cùng l?n INT\_MAX = 2^31-1

int curr; //# t?ng chi phí t?i th?i di?m hi?n t?i

int mark[MAX]; //# dánh d?u nh?ng thành ph? dã di

int x[MAX]; //# luu gi? các thành ph? dã di

//# Ðoc du lieu vào

void input(){

cin >> n;

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

for (int j = 1; j <= n; ++j){

cin >> c[i][j];

if (c[i][j] > 0) cmin = min(cmin, c[i][j]);

}

for(int i=0; i<MAX; i++)

mark[i] = 0;

curr = 0;

}

bool check(int k, int i){

if(mark[i] == 1) return false;

return true;

}

void solution(){

best = min(best,curr);

}

//# Thuat toán quay lui

void TRY(int k){

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

if(check(k,i)){

mark[i] = 1;

curr += c[x[k-1]][i];

x[k] = i;

if(k == n) {

curr += c[x[n]][1];

solution();

curr -= c[x[n]][1];

}

else TRY(k+1);

mark[i] = 0;

curr -= c[x[k-1]][i];

}

}

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

int main() {

printf ("Ho va ten: Nguyen Viet Anh\n");

printf ("MSSV:20215307\n");

input();

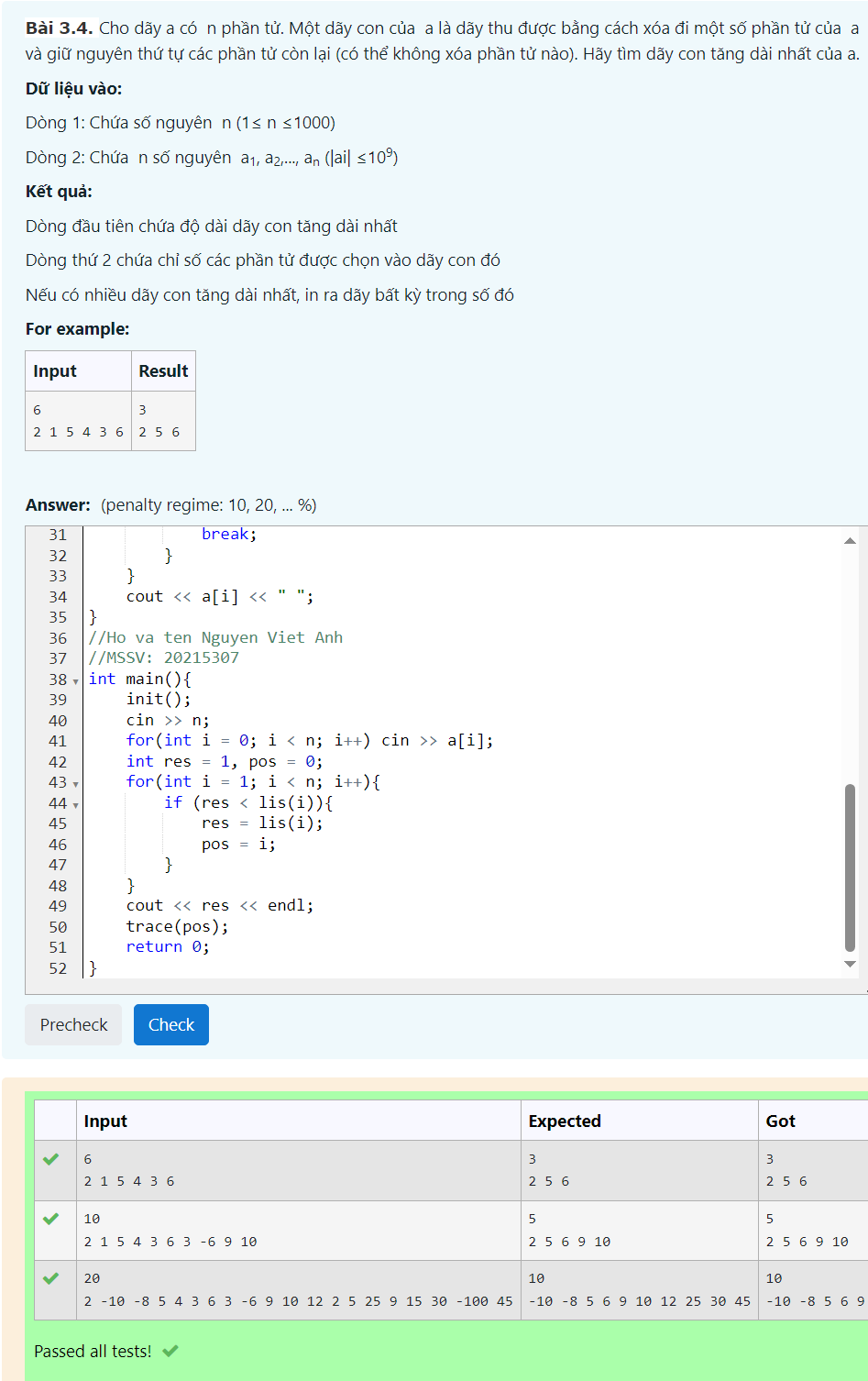
x[1] = 1;

TRY(2);

cout << best;

return 0;

}



Hình Bài 2.4 Viết hàm tìm dãy con dài nhất của a

#include <bits/stdc++.h>

using namespace std;

int a[1000], n;

int mem[1000]; //# mang ghi nho loi giai các bài toán con dã duoc giai

void init(){

memset(mem, -1, sizeof(mem));

}

//# Quy hoach dong,

//# Hàm lis(i) tra ve do dài dãy con tang dai nhat ket thuc boi a[i]

int lis(int i) {

if(i < 0)

return 0;

if(mem[i]!=-1){

return mem[i];

}

mem[i] = 1;

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

if(a[j] < a[i])

mem[i] = max(mem[i], lis(j) + 1);

}

return mem[i];

}

//# Truy vet loi giai

void trace(int i){

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

if (a[j] < a[i] && mem[i] == 1 + mem[j]){

trace(j);

break;

}

}

cout << a[i] << " ";

}

//Ho va ten Nguyen Viet Anh

//MSSV: 20215307

int main(){

printf("Ho va ten: Nguyen Viet Anh\n");

printf("MSSV: 20215307\n");

init();

cin >> n;

for(int i = 0; i < n; i++) cin >> a[i];

int res = 1, pos = 0;

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

if (res < lis(i)){

res = lis(i);

pos = i;

}

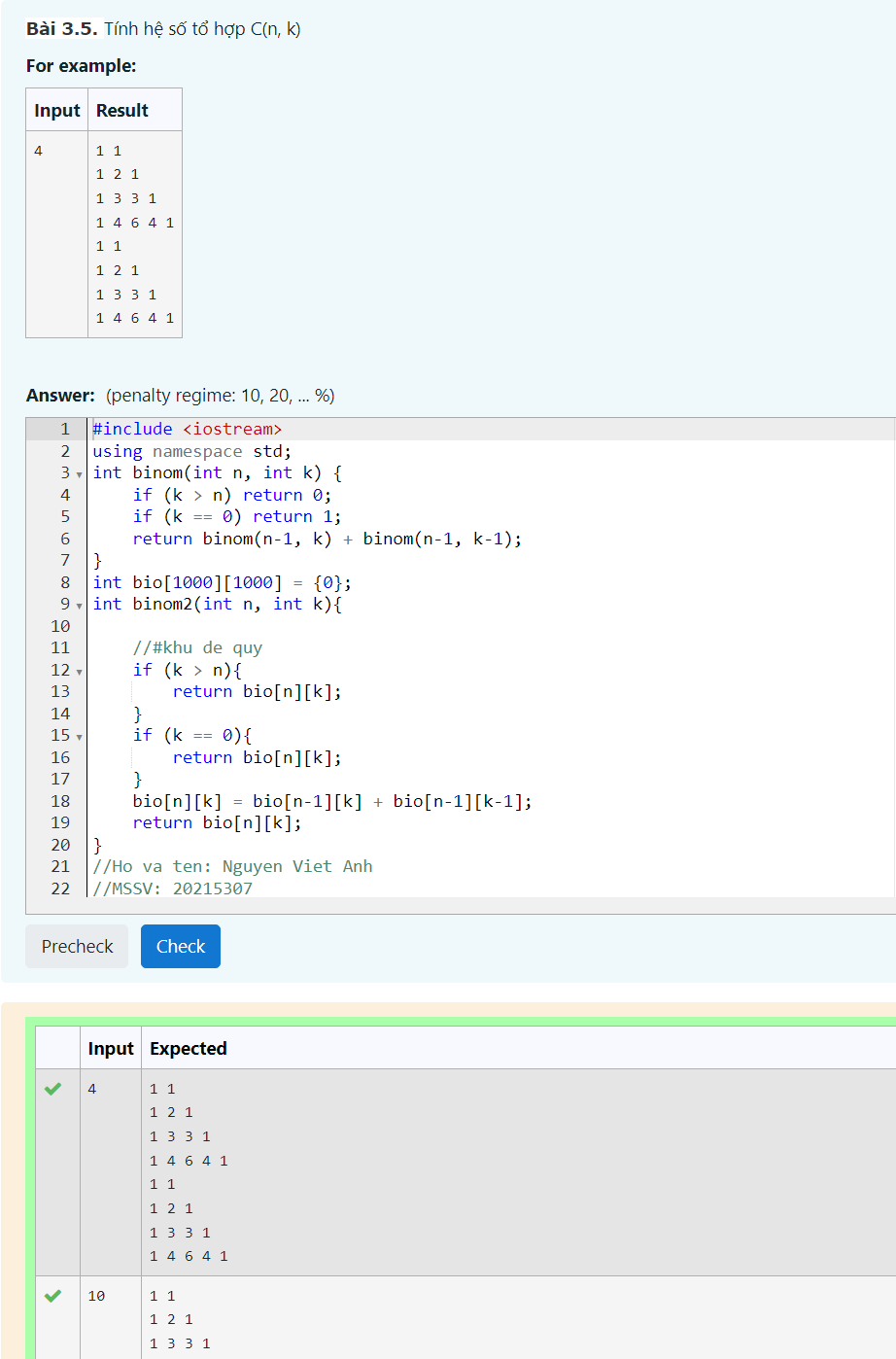
}

cout << res << endl;

trace(pos);

return 0;

}



Hình Bài 3.5 Tính hệ số tổ hợp C(n,k)

#include <iostream>

using namespace std;

int binom(int n, int k) {

if (k > n) return 0;

if (k == 0) return 1;

return binom(n-1, k) + binom(n-1, k-1);

}

int bio[1000][1000] = {0};

int binom2(int n, int k){

//#khu de quy

if (k > n){

return bio[n][k];

}

if (k == 0){

return bio[n][k];

}

bio[n][k] = bio[n-1][k] + bio[n-1][k-1];

return bio[n][k];

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

int main() {

printf("Ho va ten: Nguyen Viet Anh\n");

printf("MSSV:20215307\n");

int m;

cin >> m;

for (int n = 1; n <= m; ++n){

for (int k = 0; k <= n; ++k)

printf("%d ", binom(n, k));

printf("\n");

}

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

bio[i][0] = 1;

bio[0][i] = 1;

}

for (int n = 1; n <= m; ++n){

for (int k = 0; k <= n; ++k)

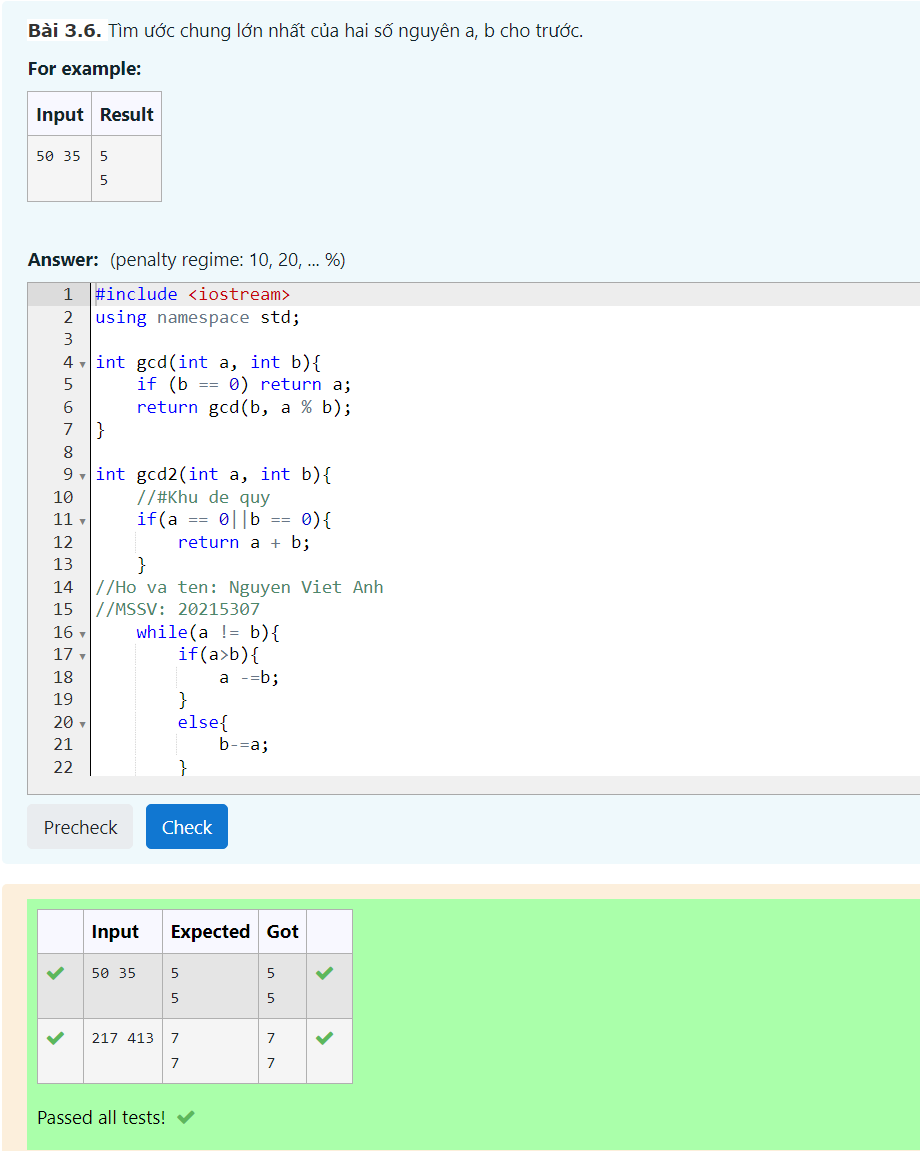
printf("%d ", binom2(n, k));

printf("\n");

}

return 0;

}}



Hình Bài 3.6 Tìm ước chung lớn nhất của 2 số a,b cho trước

#include <iostream>

using namespace std;

int gcd(int a, int b){

if (b == 0) return a;

return gcd(b, a % b);

}

int gcd2(int a, int b){

//#Khu de quy

if(a == 0||b == 0){

return a + b;

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

while(a != b){

if(a>b){

a -=b;

}

else{

b-=a;

}

}

return a;

}

int main() {

printf("Ho va ten: Nguyen Viet Anh\n");

printf("MSSV:20215307\n");

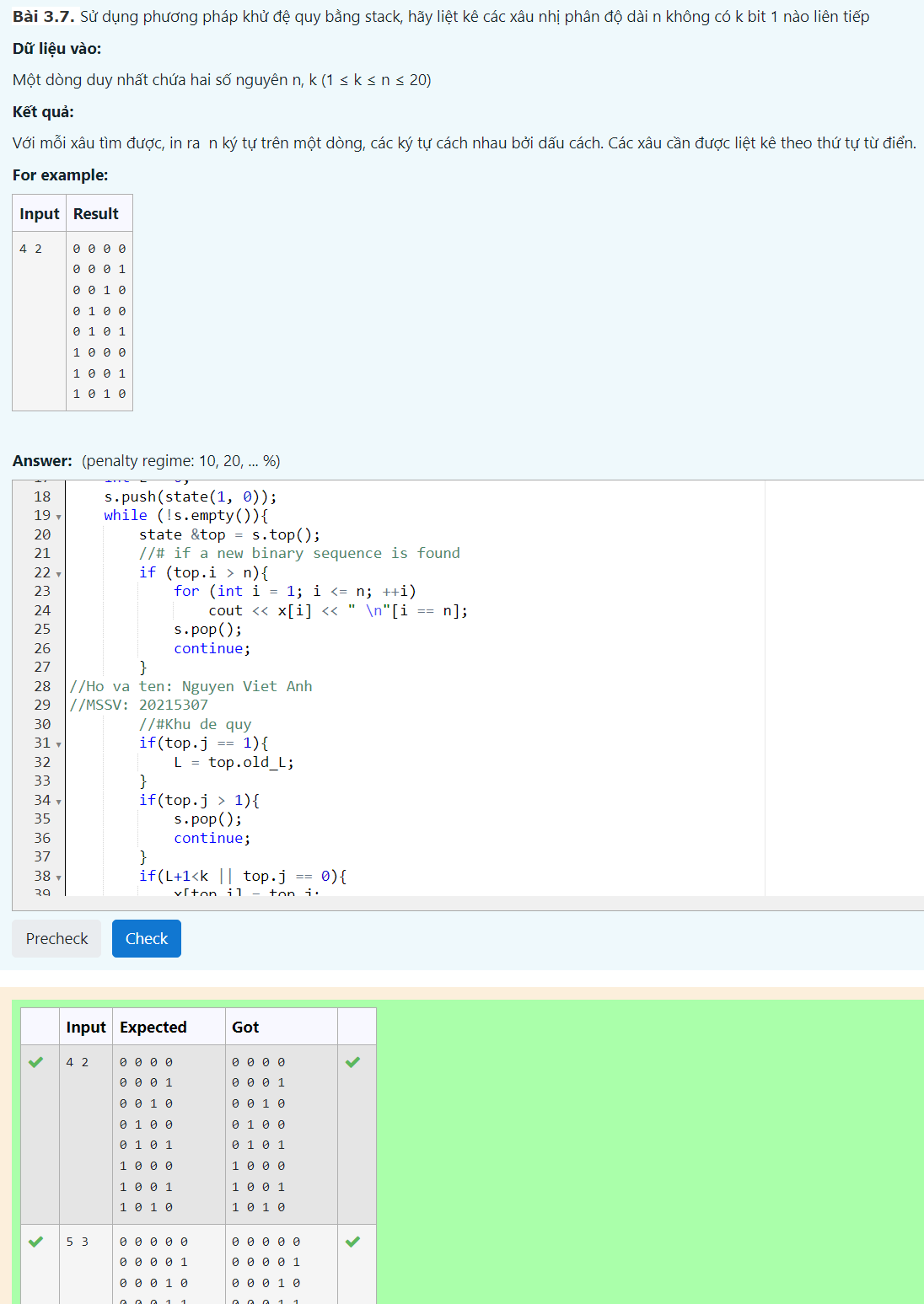
int a, b;

cin >> a >> b;

cout << gcd(a, b) << endl << gcd2(a, b);

return 0;

}



Hình Bài 3.7 Viết hàm liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp

#include <bits/stdc++.h>

using namespace std;

struct state{

int i, j, old\_L;

//# constructor

state(int \_i = 0, int \_j = 0, int \_L = 0):

i(\_i), j(\_j), old\_L(\_L){}

};

int main() {

printf("Ho va ten: Nguyen Viet Anh\n");

printf("MSSV:20215307\n");

int n, k;

cin >> n >> k;

int x[n+1];

stack<state> s;

//# number of consecutive suffix 1

int L = 0;

s.push(state(1, 0));

while (!s.empty()){

state &top = s.top();

//# if a new binary sequence is found

if (top.i > n){

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

cout << x[i] << " \n"[i == n];

s.pop();

continue;

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

//#Khu de quy

if(top.j == 1){

L = top.old\_L;

}

if(top.j > 1){

s.pop();

continue;

}

if(L+1<k || top.j == 0){

x[top.i] = top.j;

top.old\_L = L;

if(top.j){

L += 1;

}

else L = 0;

s.push(state(top.i+1,0));

}

++top.j;

}

return 0;

}

**Bài 3.8.**Viết hàm kiểm tra quả cân



Hình Bài 3.8 Viết hàm kiểm tra quả cân

#include <bits/stdc++.h>

using namespace std;

struct state{

    int i, j, s;

    state(int \_i = 0, int \_j = 0, int \_s=0): i(\_i), j(\_j), s(\_s){}

};

int main() {

    cout << "Nguyen Viet Anh\n";

    cout << "MSSV:20215307\n";

    int n, M;

    cin >> n >> M;

    int m[n+1];

    for (int i = 1; i <= n; ++i) cin >> m[i];

    int x[n+1];

    stack<state> s;

    //# sum of selected weights

    int sum = 0;

    s.push(state(1, -1));

    while (!s.empty()){

        state &top = s.top();

        if (top.i > n){

            if (sum == M){

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

                    if (x[i] == -1) cout << '-' << m[i];

                    if (x[i] == 1) cout << '+' << m[i];

                }

                cout << "=" << M;

                exit(0);

            }

            s.pop();

            continue;

        }

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

        //# Khu de quy

        s.pop();

        x[top.i]=top.j;

        s.push(state(top.i+1, -1, top.s-m[top.i+1]));

        s.push(state(top.i+1, 0,top.s));

        s.push(state(top.i+1, 1,top.s+m[top.i+1]));

    }

    cout << -1;

    return 0;

}

# **BÀI TẬP VỀ NHÀ**

Chụp ảnh kết quả của tất cả các test.

**Bài tập 9: Lập lịch cho y tá**

**Một y tá cần lập lịch làm việc trong NN ngày, mỗi ngày chỉ có thể là làm việc hay nghỉ ngơi. Một lịch làm việc là tốt nếu không có hai ngày nghỉ nào liên tiếp và mọi chuỗi ngày tối đại làm việc liên tiếp đều có số ngày thuộc đoạn [K1,K2][K1,K2]. Hãy liệt kê tất cả các cách lập lịch tốt, với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài nn với bit 0/1 tương ứng là nghỉ/làm việc. Các xâu phải được in ra theo thứ tự từ điển**

**Dữ liệu vào:**

Ghi 3 số nguyên N,K1,K2N,K1,K2 (N≤200,K1<K2≤70N≤200,K1<K2≤70)

**Kết quả:** Ghi danh sách các lịch tìm được theo thứ tự từ điển

**Ví dụ:**

**Dữ liệu mẫu:**

6 2 3

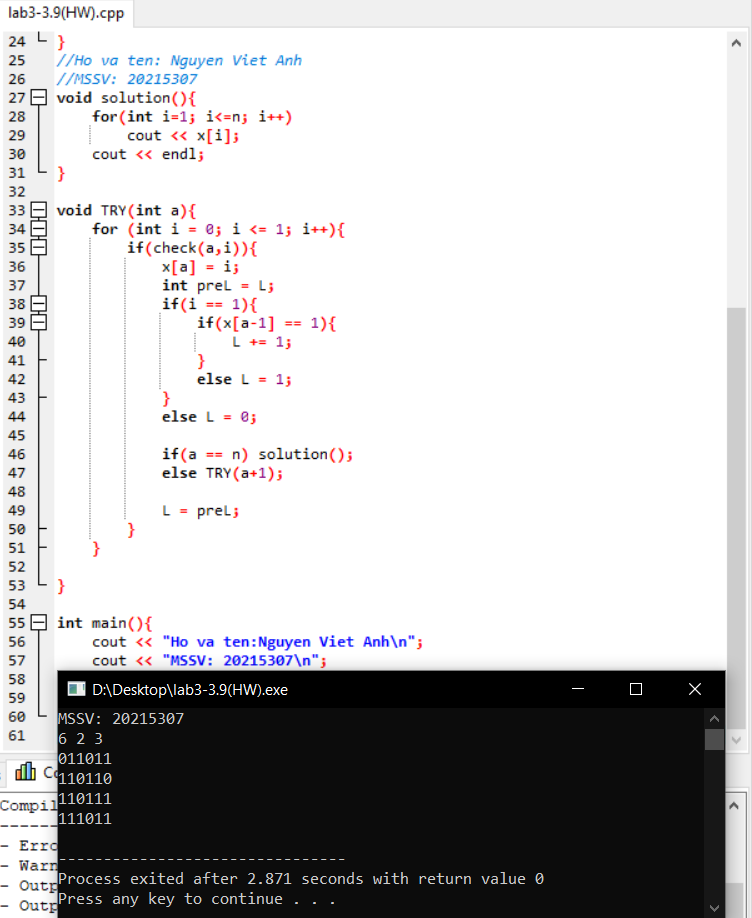
**Kết quả mẫu:**

011011

110110

110111

111011



#include<bits/stdc++.h>

using namespace std;

int n, k1 ,k2;

int x[1000];

int L;

bool check(int a,int i){

if(a == 1) return true;

else{

if(i == 0){

if(x[a-1] == 0) return false;

else if(L<k1) return false;

}

else{

if(x[a-1] == 0){

if(n-a+1 < k1) return false;

}

else{

if(L+1>k2) return false;

}

return true;

}

}

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

void solution(){

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

cout << x[i];

cout << endl;

}

void TRY(int a){

for (int i = 0; i <= 1; i++){

if(check(a,i)){

x[a] = i;

int preL = L;

if(i == 1){

if(x[a-1] == 1){

L += 1;

}

else L = 1;

}

else L = 0;

if(a == n) solution();

else TRY(a+1);

L = preL;

}

}

}

int main(){

cout << "Ho va ten:Nguyen Viet Anh\n";

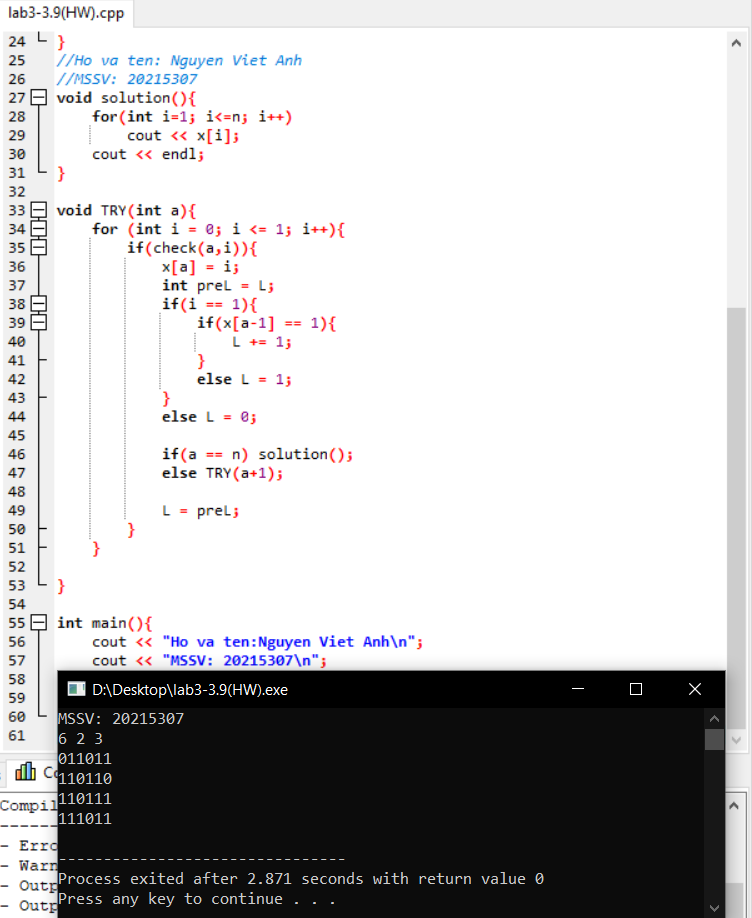
cout << "MSSV: 20215307\n";

cin >> n >> k1 >> k2;

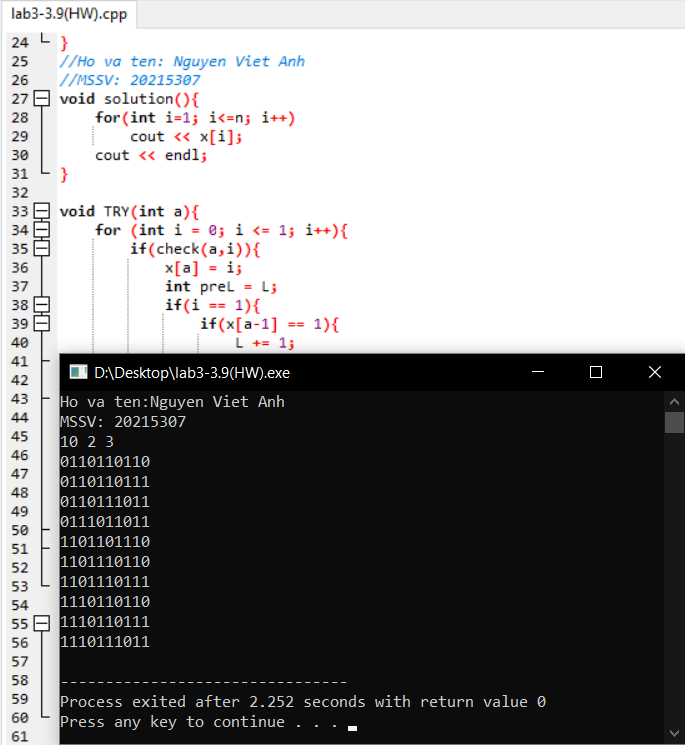
TRY(1);

}

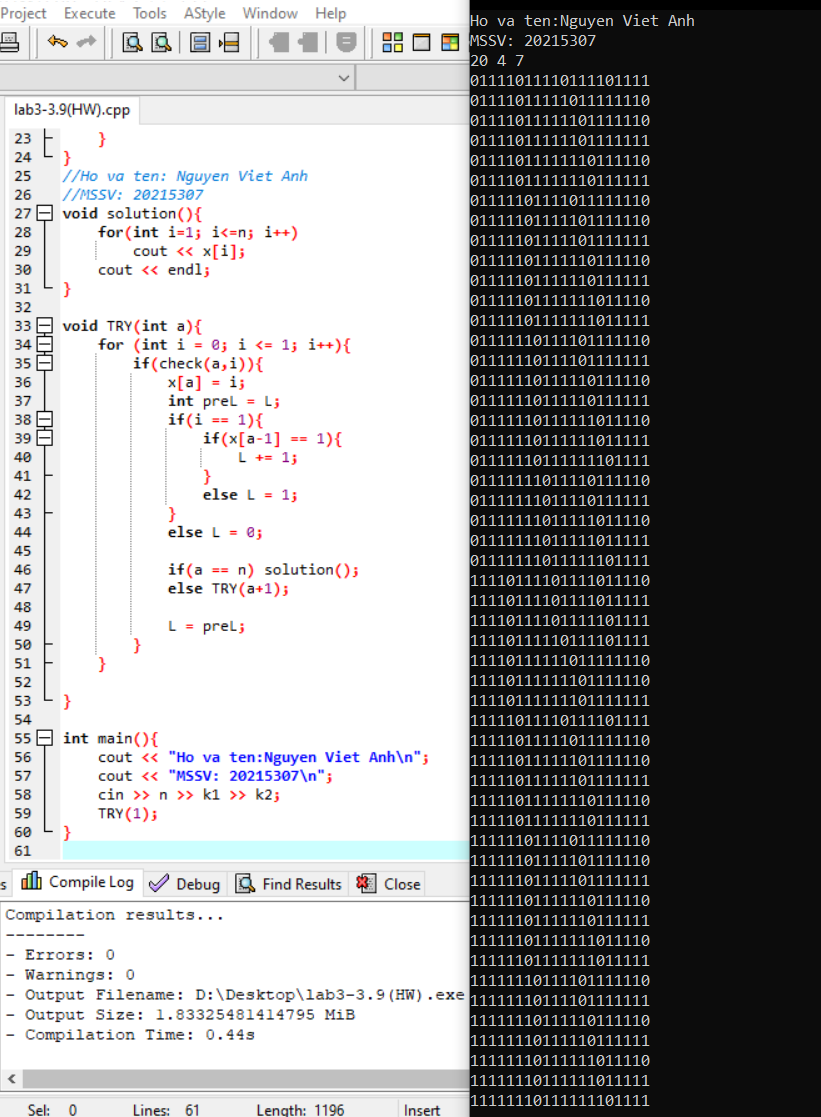
**Case 1**



**Case 2**

****

**Case 3**

****

**Bài tập 10: Khoảng cách Hamming**

**Khoảng cách Hamming giữa hai xâu cùng độ dài là số vị trí mà ký tự tại vị trí đó là khác nhau trên hai xâu. Cho SS là xâu gồm nn ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài nn, có khoảng cách Hamming với SS bằng HH. Các xâu phải được liệt kê theo thứ tự từ điển**

**Dữ liệu vào:** Dòng đầu chứa TT là số testcase TT dòng tiếp theo, mỗi dòng mô tả một testcase, ghi NN và HH (1≤H≤N≤161≤H≤N≤16)

**Kết quả:** Với mỗi testcase, in ra danh sách các xâu thỏa mãn. In ra một dòng trống giữa hai testcase

**Ví dụ:**

**Dữ liệu mẫu:**

2 4 2 1 0

**Kết quả mẫu:**

0011

0101

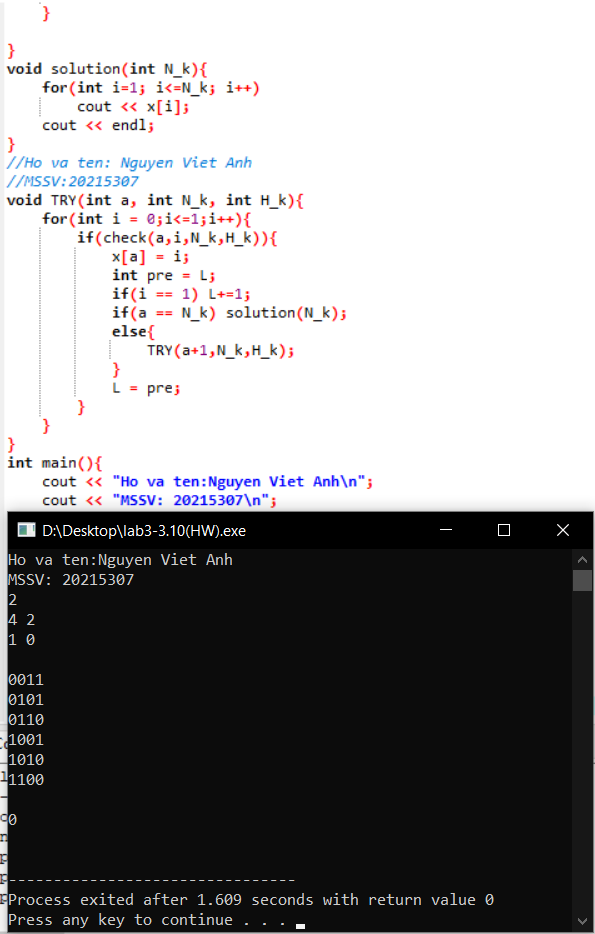
0110

1001

1010

1100

0



#include<bits/stdc++.h>

using namespace std;

int n;

const int MAX = 100;

int N[MAX],H[MAX];

int x[20];

int L;

bool check(int a, int i,int N\_k, int H\_k){

if(i == 1){

if(L + 1 > H\_k) return false;

else return true;

}

else{

if(N\_k - a + L < H\_k) return false;

else return true;

}

}

void solution(int N\_k){

for(int i=1; i<=N\_k; i++)

cout << x[i];

cout << endl;

}

//Ho va ten: Nguyen Viet Anh

//MSSV:20215307

void TRY(int a, int N\_k, int H\_k){

for(int i = 0;i<=1;i++){

if(check(a,i,N\_k,H\_k)){

x[a] = i;

int pre = L;

if(i == 1) L+=1;

if(a == N\_k) solution(N\_k);

else{

TRY(a+1,N\_k,H\_k);

}

L = pre;

}

}

}

int main(){

cout << "Ho va ten:Nguyen Viet Anh\n";

cout << "MSSV: 20215307\n";

cin >> n;

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

cin >> N[i] >> H[i];

}

cout << endl;

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

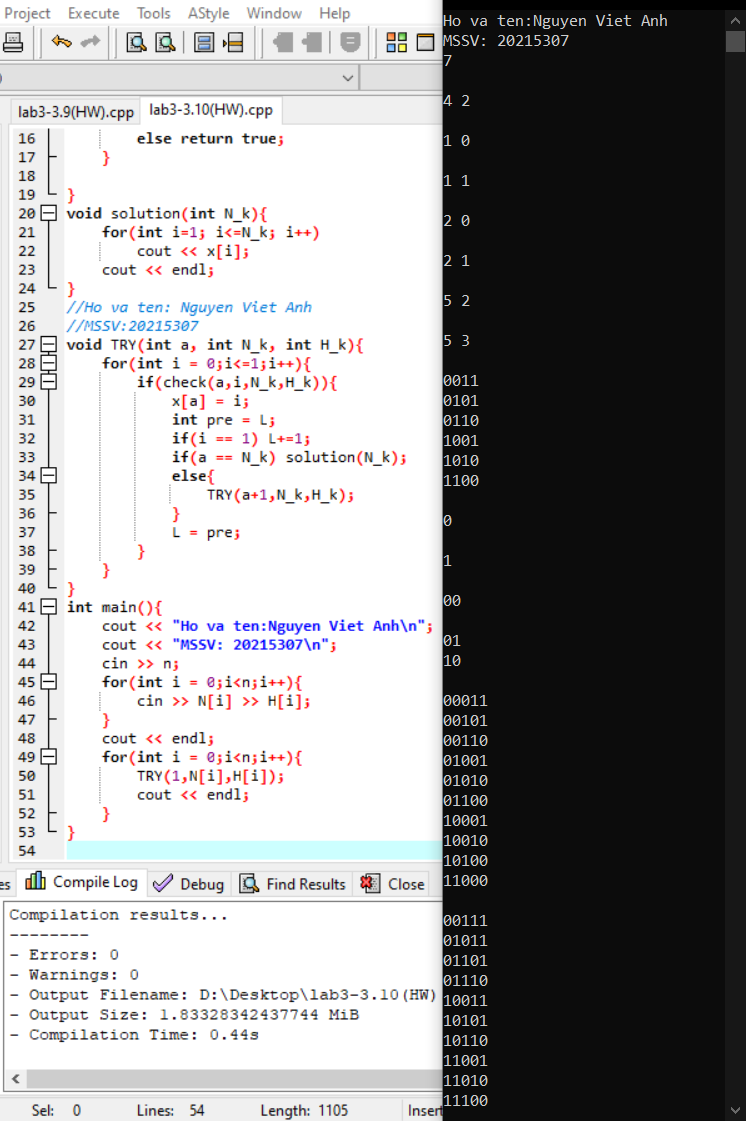
TRY(1,N[i],H[i]);

cout << endl;

}

}

**Case 1**

****

**Bài tập 11: Lịch trình chụp ảnh**

**Superior là một hòn đảo tuyệt đẹp với nn địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau. Đoàn khách tham quan có rr người với sở thích chụp ảnh khác nhau. Theo đó, mỗi người sẽ đưa ra danh sách các địa điểm mà họ muốn chụp. Bạn cần giúp mỗi người trong đoàn lập lịch di chuyển sao cho đi qua các điểm họ yêu cầu đúng một lần, không đi qua điểm nào khác, bắt đầu tại điểm đầu tiên và kết thúc tại điểm cuối cùng trong danh sách mà họ đưa ra, và có tổng khoảng cách đi lại là nhỏ nhất.**

**Dữ liệu vào:**

Dòng đầu chứa nn và rr

Tiếp theo là ma trận n×nn×n mô tả chi phí đi lại giữa các địa điểm. Chi phí bằng 0 có nghĩa là không thể đi lại giữa hai địa điểm đó. rr dòng tiếp theo chứa danh sách các địa điểm mà người rr đưa ra. Lưu ý là hành mỗi hành trình cần phải bắt đầu và kết thúc bởi hai đỉnh đầu và cuối của danh sách, còn các địa điểm còn lại có thể thăm theo bất kỳ thứ tự nào

**Kết quả:** Gồm rr dòng ghi chi phí đi lại ít nhất của rr người theo thứ tự đầu vào

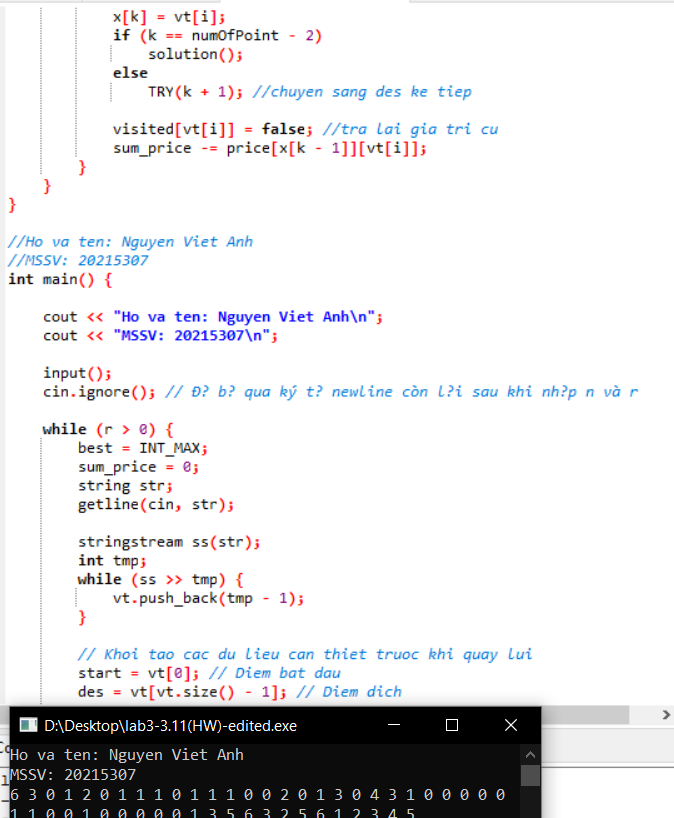
**Ví dụ:**

**Dữ liệu mẫu:**

6 3 0 1 2 0 1 1 1 0 1 1 1 0 0 2 0 1 3 0 4 3 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 3 5 6 3 2 5 6 1 2 3 4 5

**Kết quả mẫu:**

5 0 7

****

#include <bits/stdc++.h>

using namespace std;

const int MAX = 10000;

int n, r;

int price[MAX][MAX];

int x[MAX];

bool visited[MAX];

vector<int> vt;

int best;

int sum\_price;

int start, des, numOfPoint;

void input() { //nhap ma tran

cin >> n >> r;

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

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

cin >> price[i][j];

}

}

}

bool check(int a, int i) {

if (visited[vt[i]]) return false; //neu visited thi tra ve false

if (price[x[a - 1]][vt[i]] == 0) return false; //neu price = 0 tra ve false

return true;

}

void solution() {

if (price[x[numOfPoint - 2]][des] == 0) return;

best = min(best, sum\_price + price[x[numOfPoint - 2]][des]);

}

void TRY(int k) {

for (int i = 1; i < numOfPoint - 1; i++) {

if (check(k, i)) { //kiem tra kha nang

visited[vt[i]] = true; //set visited

sum\_price += price[x[k - 1]][vt[i]]; //cap nhat price

x[k] = vt[i];

if (k == numOfPoint - 2)

solution();

else

TRY(k + 1); //chuyen sang des ke tiep

visited[vt[i]] = false; //tra lai gia tri cu

sum\_price -= price[x[k - 1]][vt[i]];

}

}

}

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

int main() {

cout << "Ho va ten: Nguyen Viet Anh\n";

cout << "MSSV: 20215307\n";

input();

cin.ignore(); // Ð? b? qua ký t? newline còn l?i sau khi nh?p n và r

while (r > 0) {

best = INT\_MAX;

sum\_price = 0;

string str;

getline(cin, str);

stringstream ss(str);

int tmp;

while (ss >> tmp) {

vt.push\_back(tmp - 1);

}

// Khoi tao cac du lieu can thiet truoc khi quay lui

start = vt[0]; // Diem bat dau

des = vt[vt.size() - 1]; // Diem dich

numOfPoint = vt.size(); // So diem phai di qua

x[0] = start;

x[numOfPoint - 1] = des;

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

visited[i] = false;

TRY(1);

// In ket qua

if (best == INT\_MAX)

cout << "0" << endl;

else

cout << best << endl;

// Xoa vector va chuyen sang khach tiep theo

vt.clear();

r--;

}

return 0;

}

**Bài tập 12: Đếm đường đi**

**Cho đồ thị vô hướng GG, hãy đếm số đường đi đi qua kk cạnh và không đi qua đỉnh nào quá một lần.**

**Dữ liệu vào:**

Dòng 1: Chứa hai số nguyên nn và kk (1≤n≤301≤n≤30, 1≤k≤101≤k≤10) với nn là số đỉnh của GG. Các đỉnh sẽ được đánh số từ 1 đến nn

Dòng 2: Chứa số nguyên mm (1≤m≤601≤m≤60) là số cạnh của GG mm dòng tiếp theo: Mỗi dòng chưa hai số nguyên là một cạnh của GG

**Kết quả:** Số lượng đường đi đơn độ dài kk

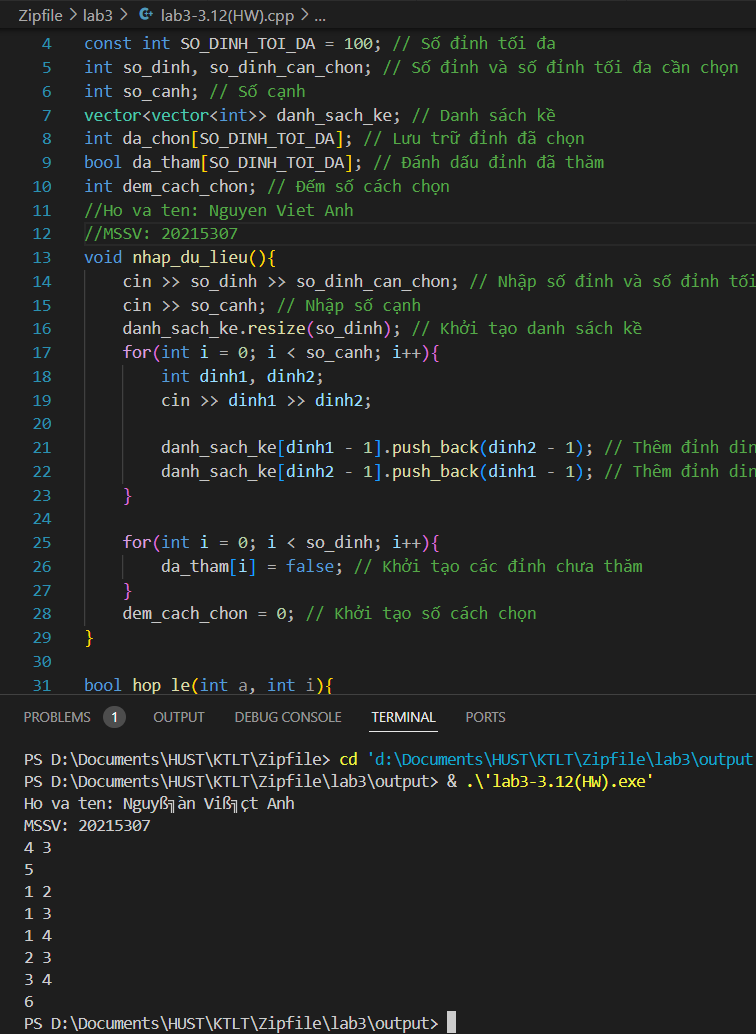
**Ví dụ:**

**Dữ liệu mẫu:**

4 3 5 1 2 1 3 1 4 2 3 3 4

**Kết quả mẫu:**

6



#include<bits/stdc++.h>

using namespace std;

const int SO\_DINH\_TOI\_DA = 100; // Số đỉnh tối đa

int so\_dinh, so\_dinh\_can\_chon; // Số đỉnh và số đỉnh tối đa cần chọn

int so\_canh; // Số cạnh

vector<vector<int>> danh\_sach\_ke; // Danh sách kề

int da\_chon[SO\_DINH\_TOI\_DA]; // Lưu trữ đỉnh đã chọn

bool da\_tham[SO\_DINH\_TOI\_DA]; // Đánh dấu đỉnh đã thăm

int dem\_cach\_chon; // Đếm số cách chọn

//Ho va ten: Nguyen Viet Anh

//MSSV: 20215307

void nhap\_du\_lieu(){

    cin >> so\_dinh >> so\_dinh\_can\_chon; // Nhập số đỉnh và số đỉnh tối đa cần chọn

    cin >> so\_canh; // Nhập số cạnh

    danh\_sach\_ke.resize(so\_dinh); // Khởi tạo danh sách kề

    for(int i = 0; i < so\_canh; i++){

        int dinh1, dinh2;

        cin >> dinh1 >> dinh2;

        danh\_sach\_ke[dinh1 - 1].push\_back(dinh2 - 1); // Thêm đỉnh dinh2 vào danh sách kề của dinh1

        danh\_sach\_ke[dinh2 - 1].push\_back(dinh1 - 1); // Thêm đỉnh dinh1 vào danh sách kề của dinh2

    }

    for(int i = 0; i < so\_dinh; i++){

        da\_tham[i] = false; // Khởi tạo các đỉnh chưa thăm

    }

    dem\_cach\_chon = 0; // Khởi tạo số cách chọn

}

bool hop\_le(int a, int i){

    if(a == 0) return true; // Nếu a = 0, luôn hợp lệ

    if(da\_tham[i]) return false; // Nếu đỉnh đã thăm, không hợp lệ

    int index = 0;

    for(int j = 0; j < danh\_sach\_ke[da\_chon[a-1]].size(); j++){

        if(i == danh\_sach\_ke[da\_chon[a-1]][j]) index++;

    }

    if(index == 0) return false; // Nếu không kề với đỉnh trước đó, không hợp lệ

    return true; // Nếu đủ điều kiện, hợp lệ

}

void ghi\_nhan\_cach\_chon(){

    dem\_cach\_chon++; // Tăng số cách chọn

}

void THU(int a){

    for(int i = 0; i < so\_dinh; i++){

        if(hop\_le(a, i)){

            da\_tham[i] = true; // Đánh dấu đỉnh đã thăm

            da\_chon[a] = i; // Lưu lại đỉnh đã chọn

            if(a == so\_dinh\_can\_chon) ghi\_nhan\_cach\_chon(); // Nếu đã chọn đủ số đỉnh cần thiết, thực hiện ghi\_nhan\_cach\_chon

            else THU(a + 1); // Thử tiếp tục chọn đỉnh tiếp theo

            da\_tham[i] = false; // Trả lại trạng thái ban đầu

        }

    }

}

int main(){

    cout << "Ho va ten: Nguyễn Việt Anh\n";

    cout << "MSSV: 20215307\n";

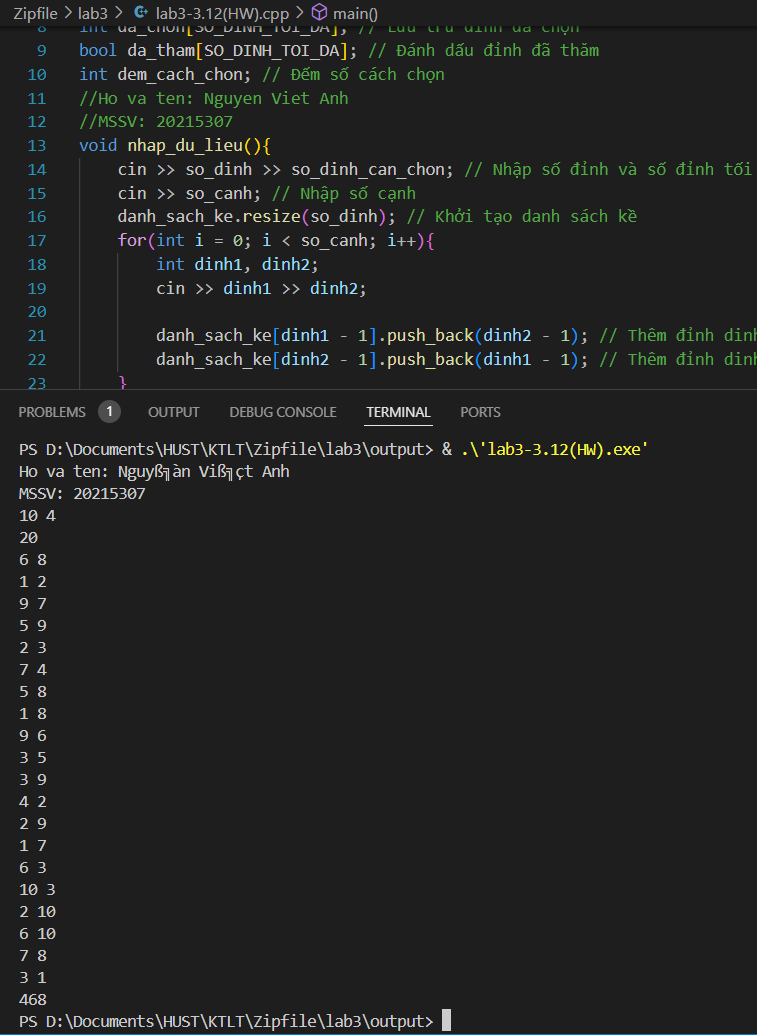
    nhap\_du\_lieu(); // Nhập dữ liệu

    THU(0); // Bắt đầu thử các cách chọn

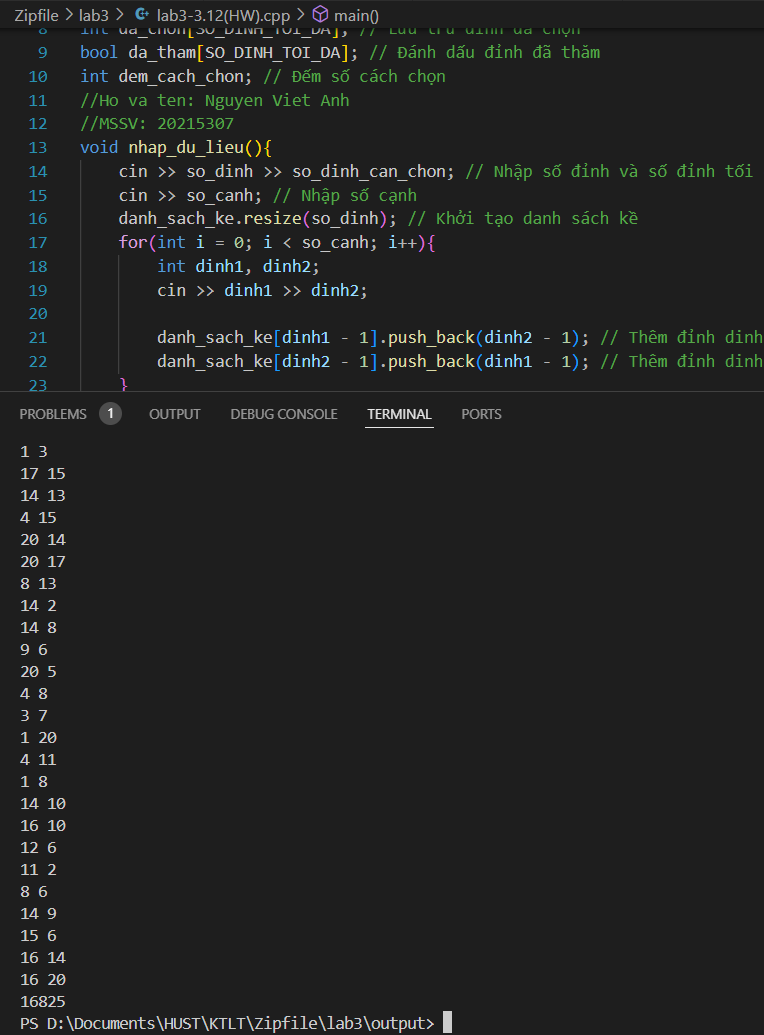
    cout << dem\_cach\_chon / 2; // In ra kết quả, chia 2 vì mỗi cách chọn được đếm hai lần

}

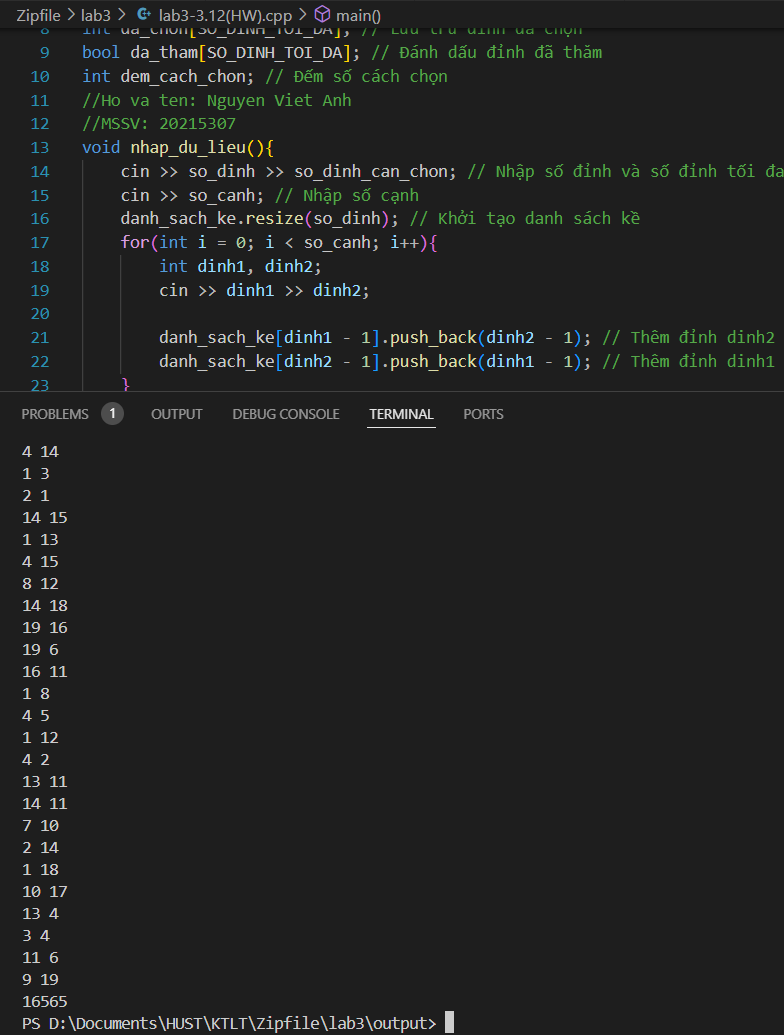
**Case 1**



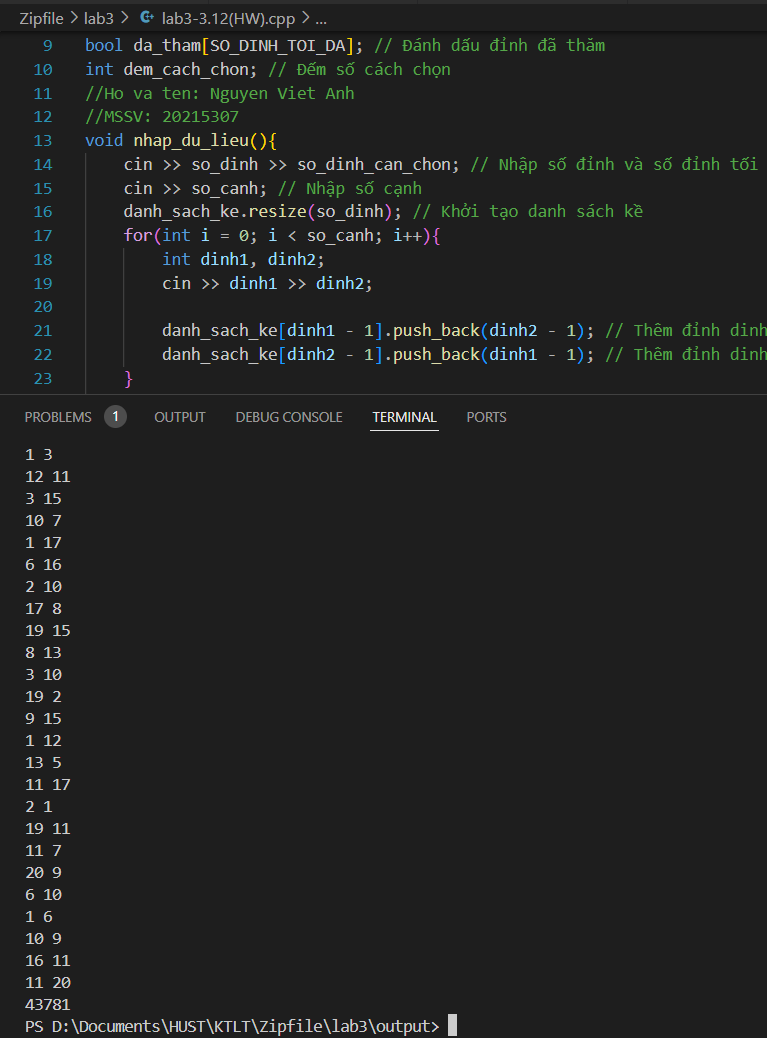
**Case 2**



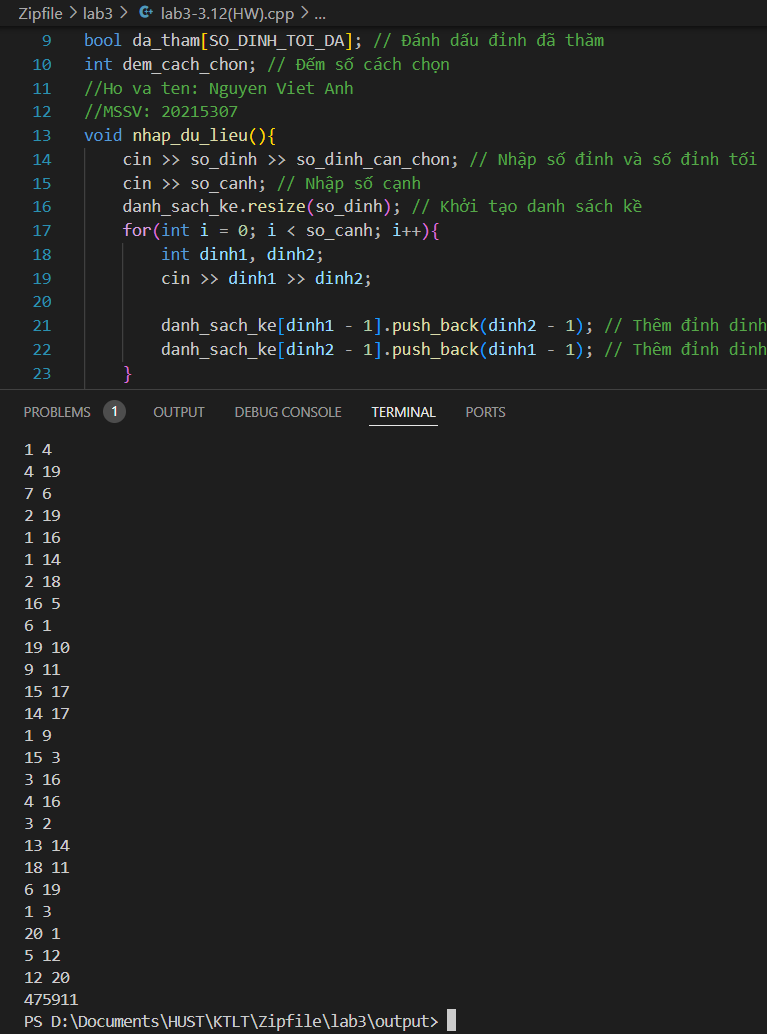
**Case 3**



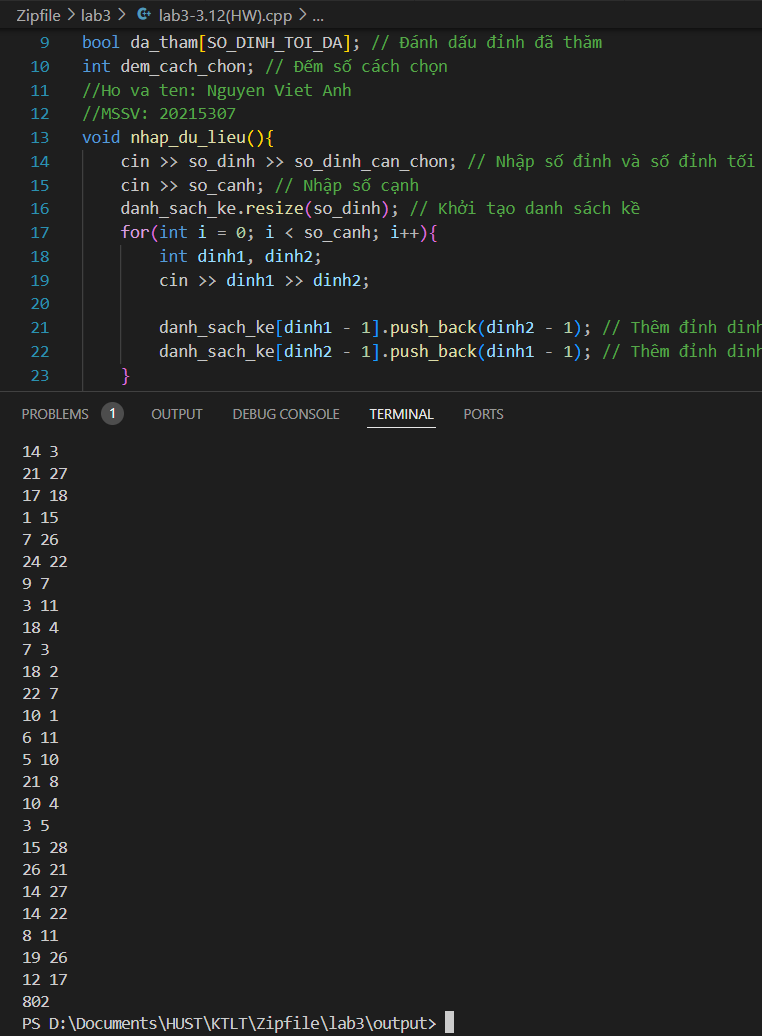
**Case 4**



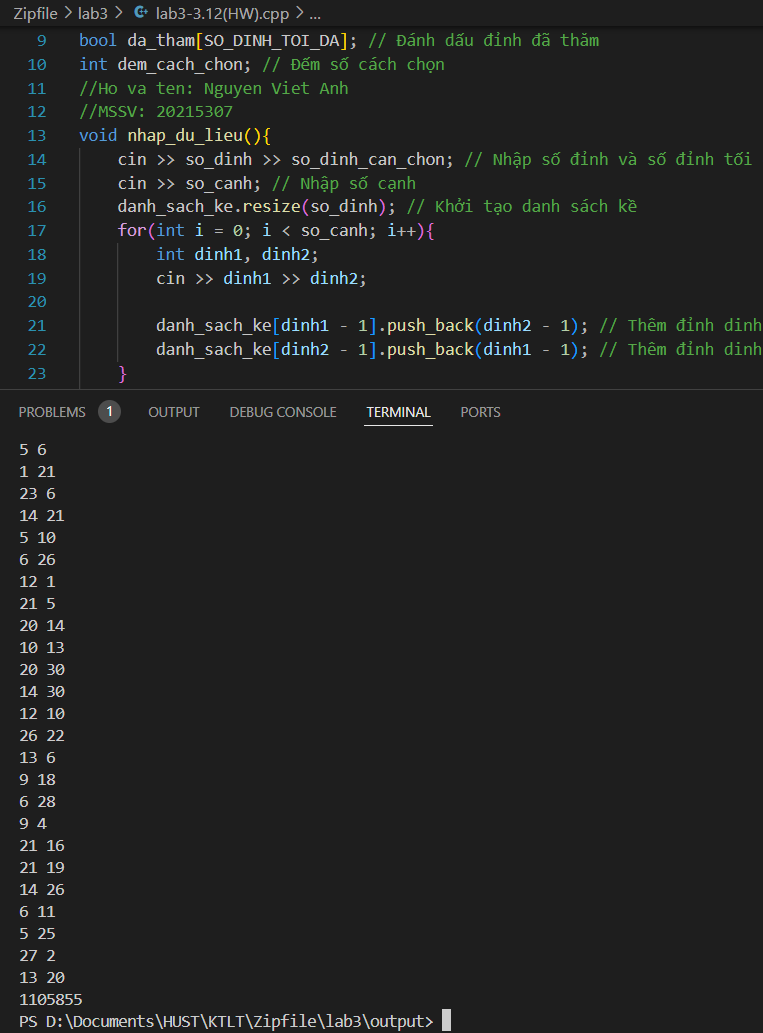
**Case 5**

****

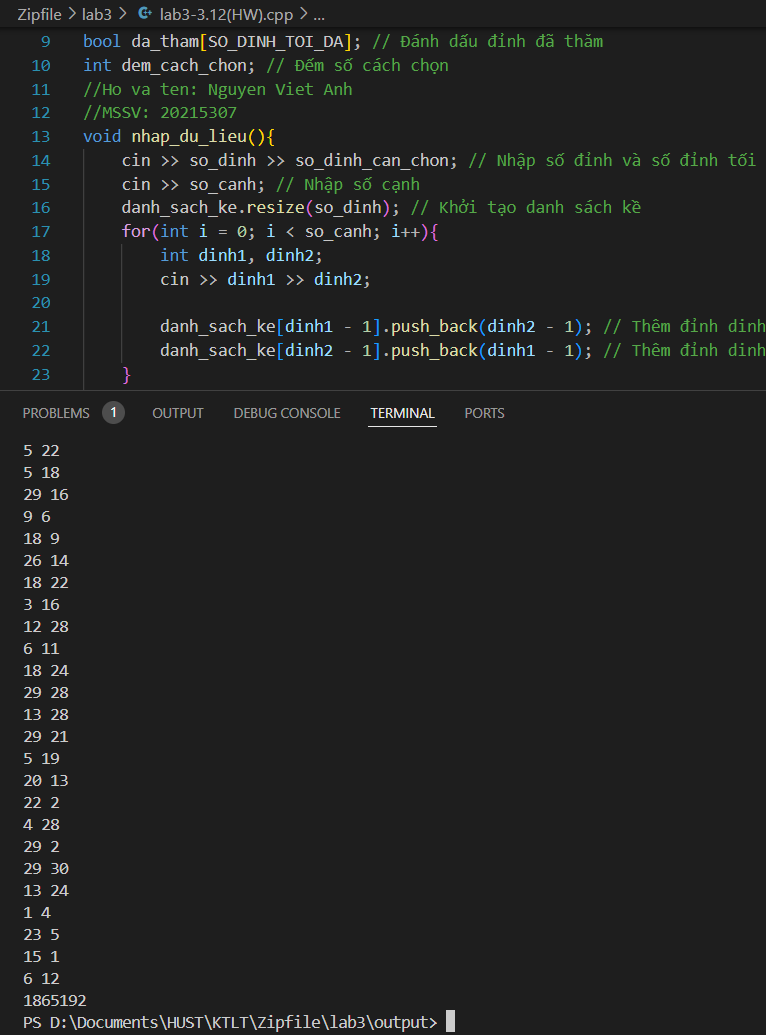
**Case 6**

****

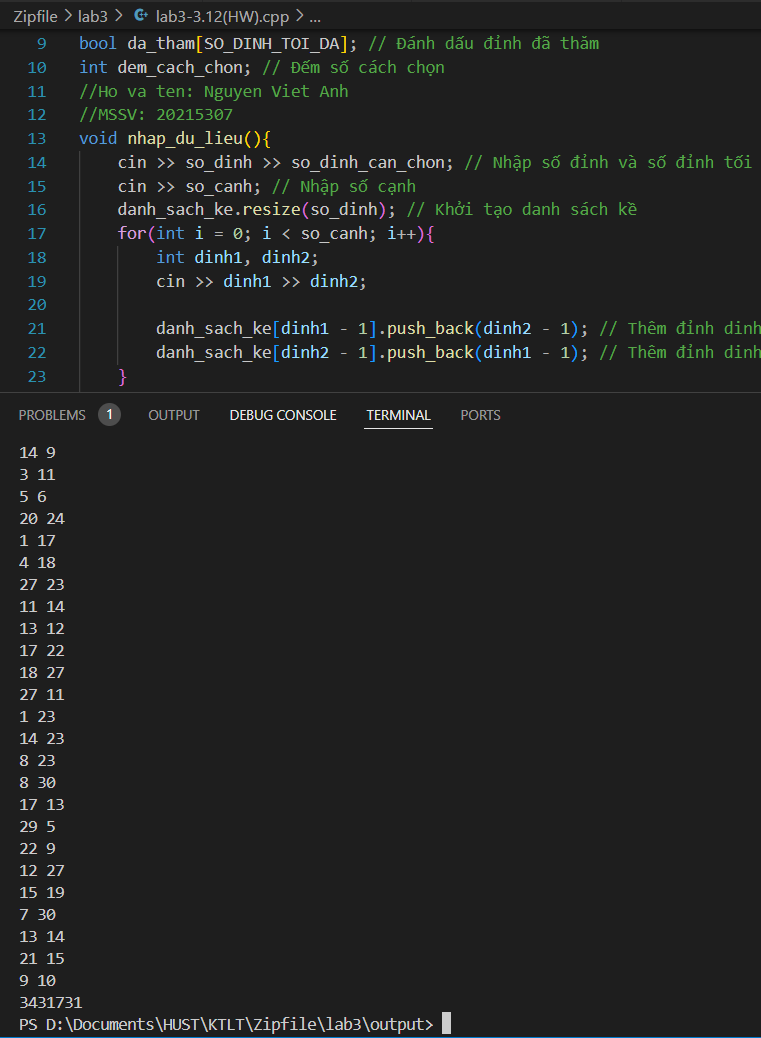
**Case 7**

****

**Case 8**

****

**Case 9**

****