ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

**BÁO CÁO BÀI THỰC HÀNH SỐ 3**

**MÔN KỸ THUẬT LẬP TRÌNH**

Sinh viên: Lê Đình Trí Tuệ

MSSV: 20210909

Mã lớp: 732833

Contents

[Bài thực hành số 3 – Tuần 14 4](#_Toc153476114)

[**Bài tập 3.1.**Dãy Lucas được định nghĩa bởi Ln = Ln-1 + Ln-2 với L0 = 2, L1 = 1. Hãy viết hàm tính số Lucas thứ n. 4](#_Toc153476115)

[**Bài tập 3.2.** Trên bàn cờ vua kích thước n\*n có một quân mã đang ở ô (1, 1). Hãy đưa ra một dãy 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). 5](#_Toc153476116)

[**Bài tập 3.3.** 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. Chi phí để đi từ thành phố i sang thành phố j là cij. Hãy tìm tổng chi phí nhỏ nhất có thể. 8](#_Toc153476117)

[**Bài tập 3.4.** Cho dãy a có n phần tử. Một dãy con của a là dãy thu được bằng cách xóa đi một số phần tử của a 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 a. 11](#_Toc153476118)

[**Bài tập 3.5.** Tính hệ số tổ hợp C(n, k). 14](#_Toc153476119)

[**Bài tập 3.6.** Tìm ước chung lớn nhất của hai số nguyên a, b cho trước. 16](#_Toc153476120)

[**Bài tập 3.7.** Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp. 17](#_Toc153476121)

[**Bài tập 3.8.** Cân đĩa: Bạn đang muốn kiểm tra xem một vật cho trước có đúng nặng M như người ta nói hay không. Có một cân thăng bằng và n quả cân. Quả thứ i nặng mi. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã nguồn dưới. 19](#_Toc153476122)

[**Bài tập 3.9.** Lập lịch cho y tá: Một y tá cần lập lịch làm việc trong N 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]. Hãy liệt kê tất cả các cách lập lịch tốt, với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài n với bit 0/1 tương ứng là nghỉ/làm việc. Các xâu phải được in ra theo thứ tự từ điển. 22](#_Toc153476123)

[**Bài tập 3.10.** 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 S là xâu gồm n ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài n, có khoảng cách Hamming với S bằng H. Các xâu phải được liệt kê theo thứ tự từ điển. 26](#_Toc153476124)

[**Bài tập 3.11.** Lịch trình chụp ảnh Superior là một hòn đảo tuyệt đẹp với n địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau. Đoàn khách tham quan có r 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. 30](#_Toc153476125)

[**Bài tập 3.12.** Đếm đường đi Cho đồ thị vô hướng G, hãy đếm số đường đi đi qua k cạnh và không đi qua đỉnh nào quá một lần. 35](#_Toc153476126)

Figure 1 4

Figure 2 4

Figure 3 5

Figure 4 5

Figure 5 6

Figure 6 7

Figure 7 8

Figure 8 8

Figure 9 9

Figure 10 10

Figure 11 11

Figure 12 11

Figure 13 12

Figure 14 13

Figure 15 14

Figure 16 14

Figure 17 15

Figure 18 16

Figure 19 17

Figure 20 17

Figure 21 18

Figure 22 19

Figure 23 24

Figure 24 25

Figure 25 28

Figure 26 29

Figure 27 33

Figure 28 33

Figure 29 36

Figure 30 37

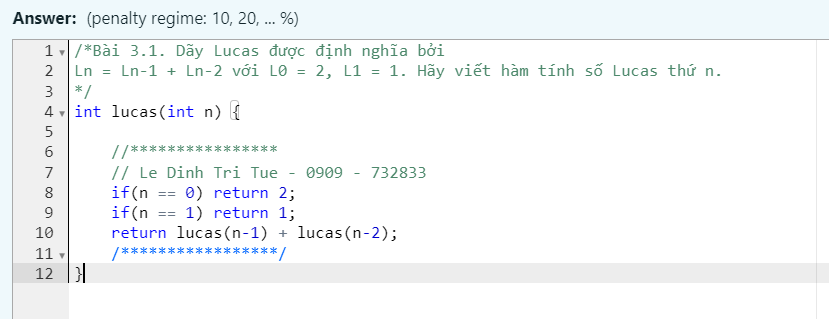
Figure 31 38

Figure 32 39

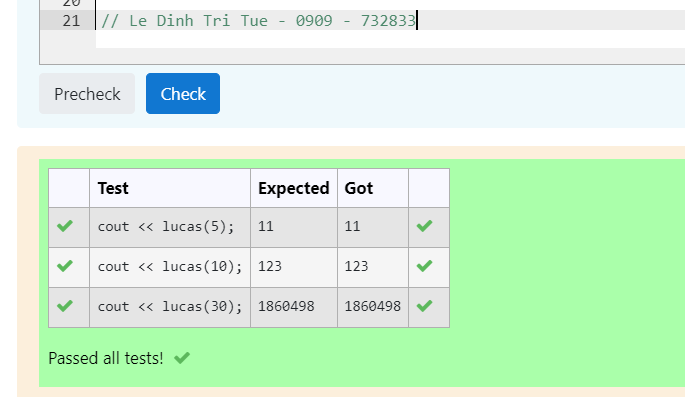
# Bài thực hành số 3 – Tuần 14

**Copy code (kèm ảnh code và kết quả vào )**

## **Bài tập 3.1.**Dãy Lucas được định nghĩa bởi Ln = Ln-1 + Ln-2 với L0 = 2, L1 = 1. Hãy viết hàm tính số Lucas thứ n.

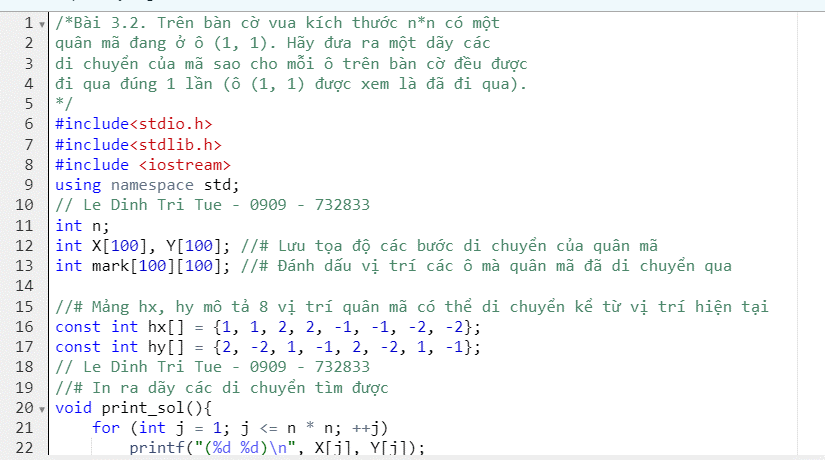


Figure

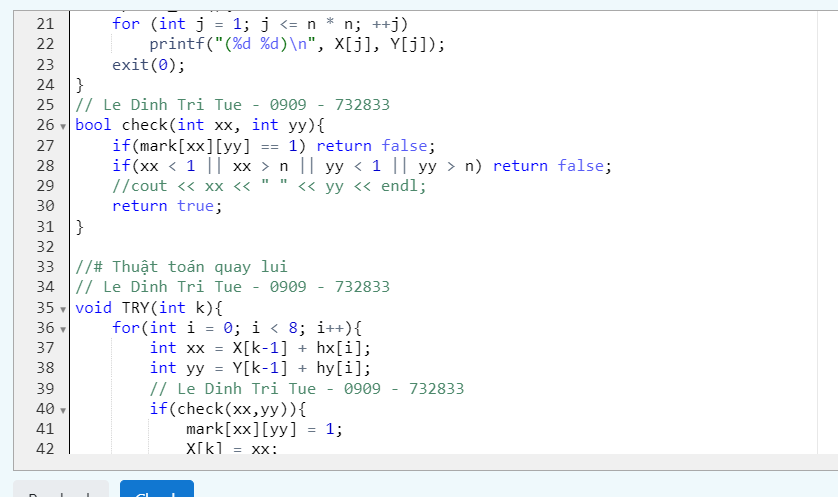


Figure

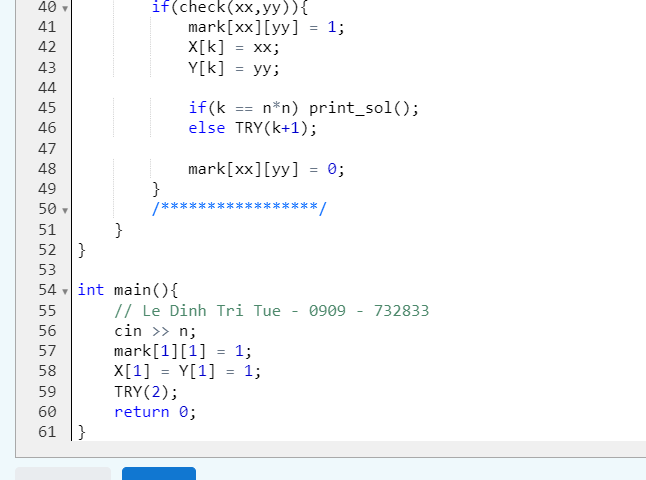
## **Bài tập 3.2.** Trên bàn cờ vua kích thước n\*n có một quân mã đang ở ô (1, 1). Hãy đưa ra một dãy các di chuyển của mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần (ô (1, 1) được xem là đã đi qua).



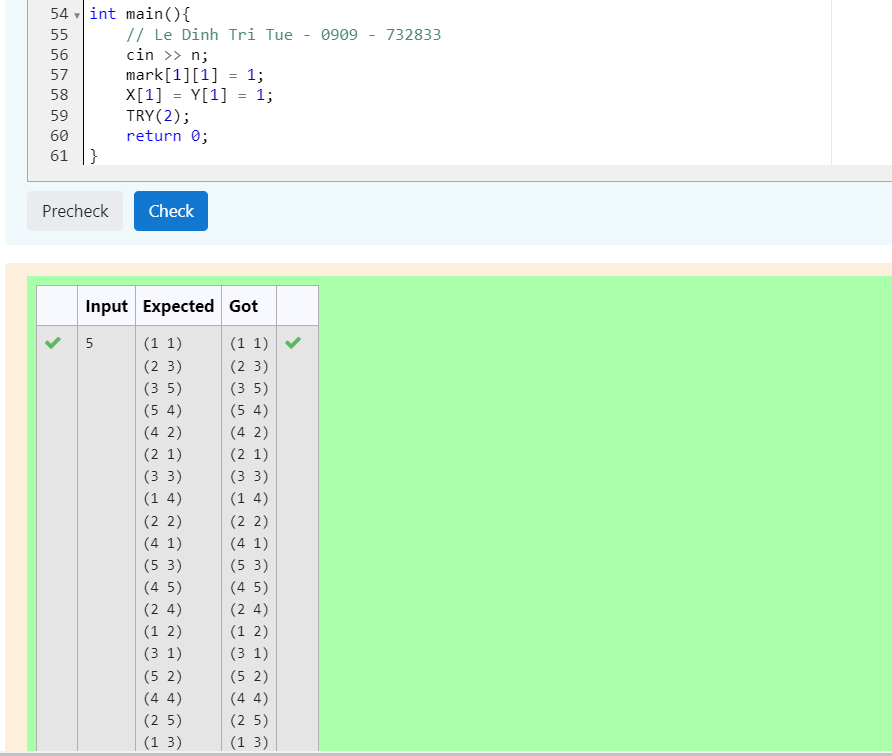
Figure



Figure

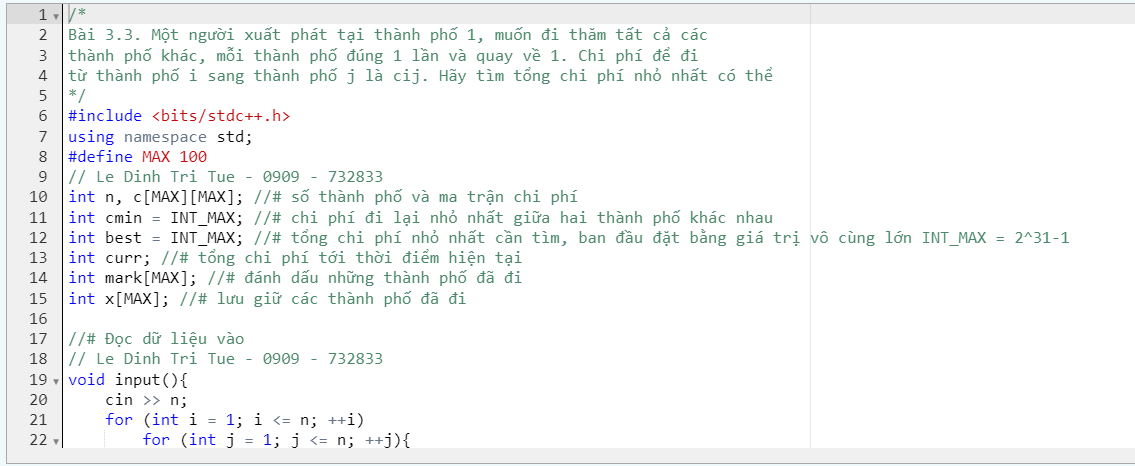


Figure

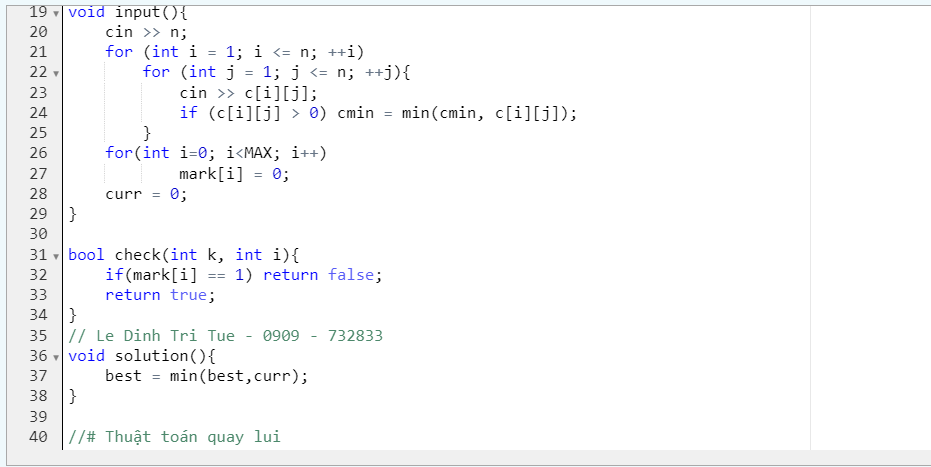


Figure

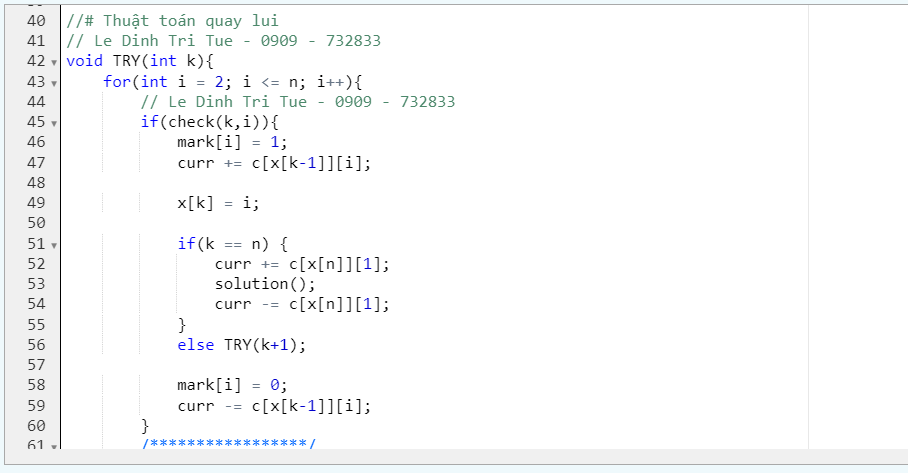
## **Bài tập 3.3.** Một người xuất phát tại thành phố 1, muốn đi thăm tất cả các thành phố khác, mỗi thành phố đúng 1 lần và quay về 1. Chi phí để đi từ thành phố i sang thành phố j là cij. Hãy tìm tổng chi phí nhỏ nhất có thể.



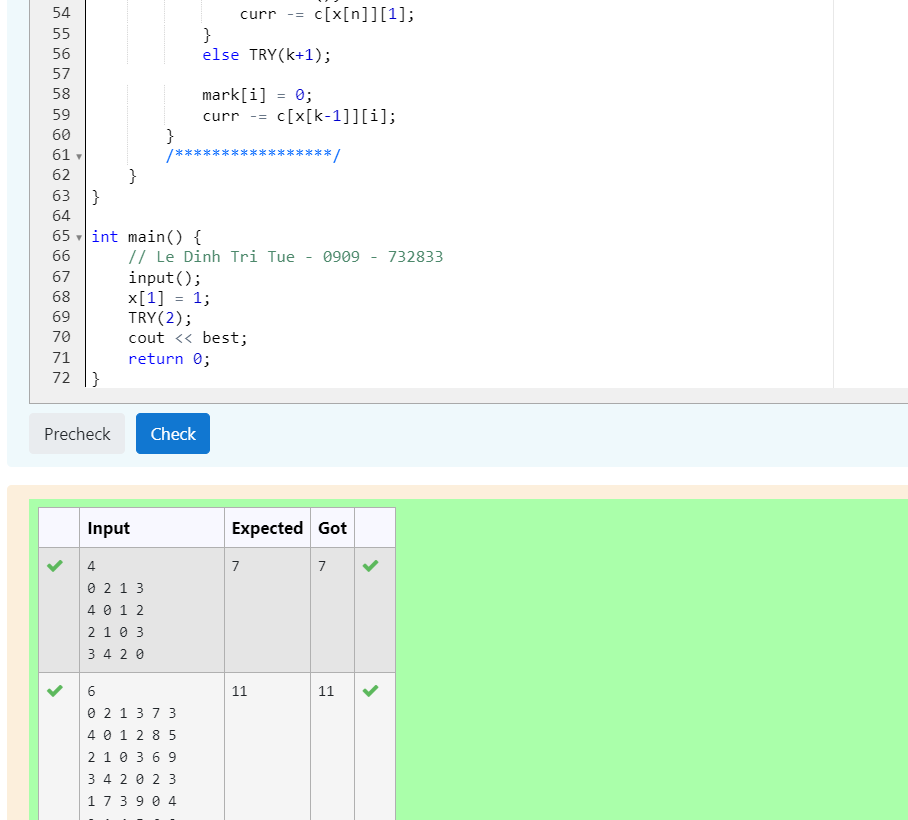
Figure



Figure

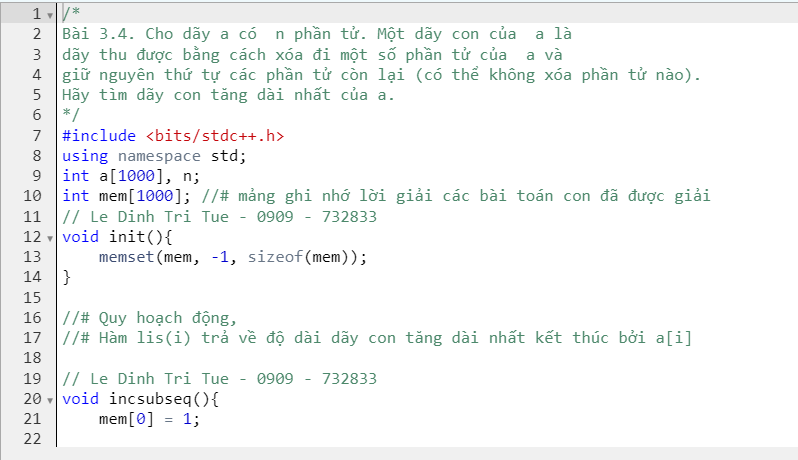


Figure

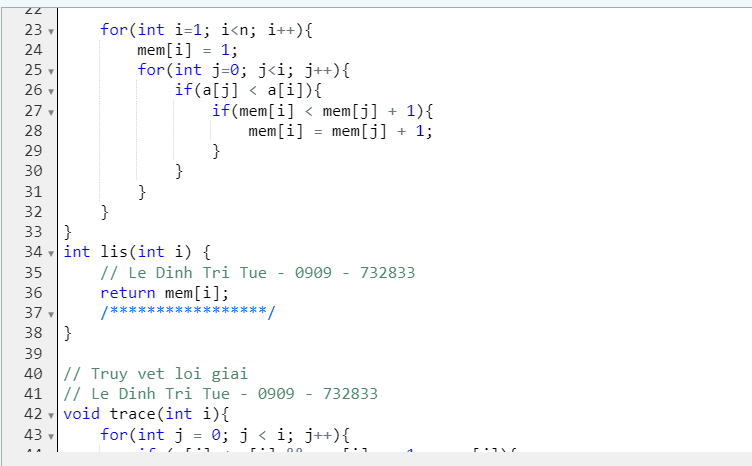


Figure

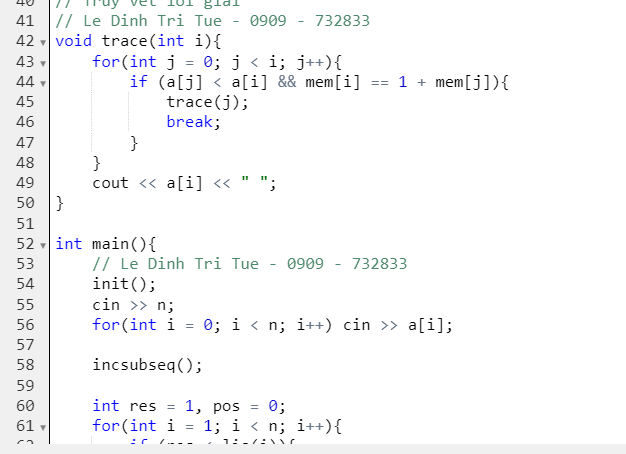
## **Bài tập 3.4.** Cho dãy a có n phần tử. Một dãy con của a là dãy thu được bằng cách xóa đi một số phần tử của a và giữ nguyên thứ tự các phần tử còn lại (có thể không xóa phần tử nào). Hãy tìm dãy con tăng dài nhất của a.



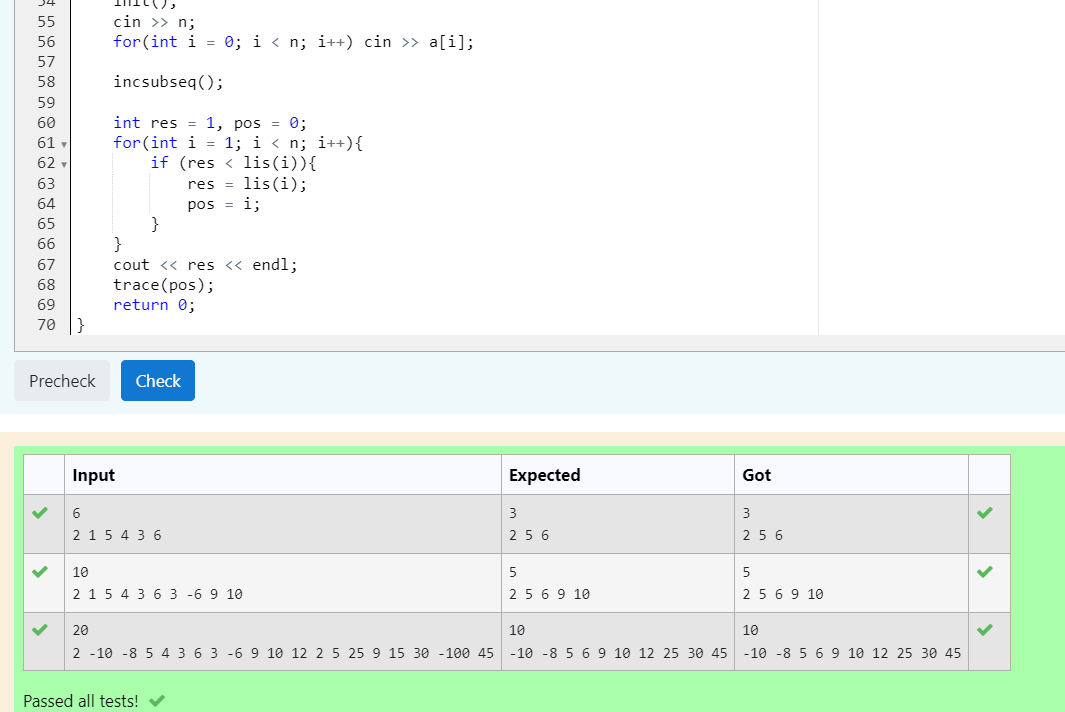
Figure



Figure

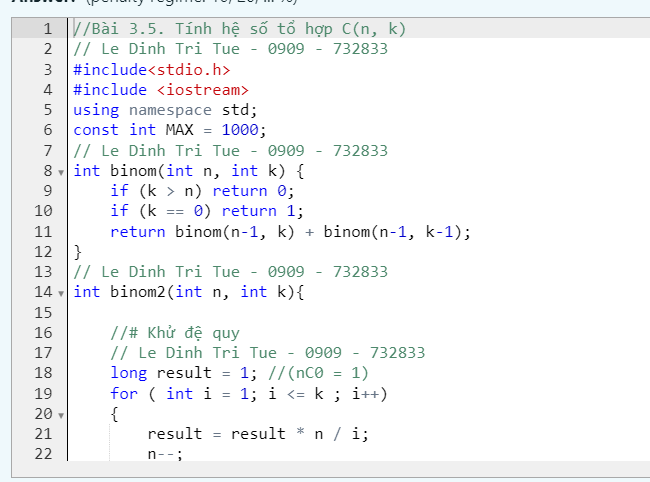


Figure

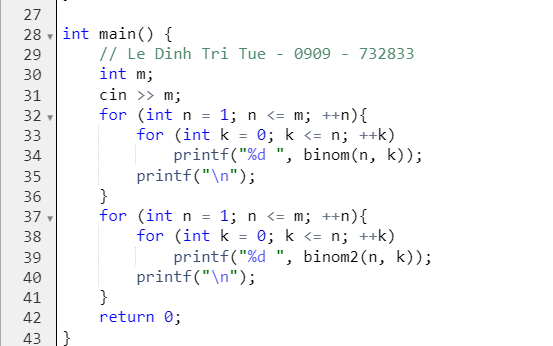


Figure

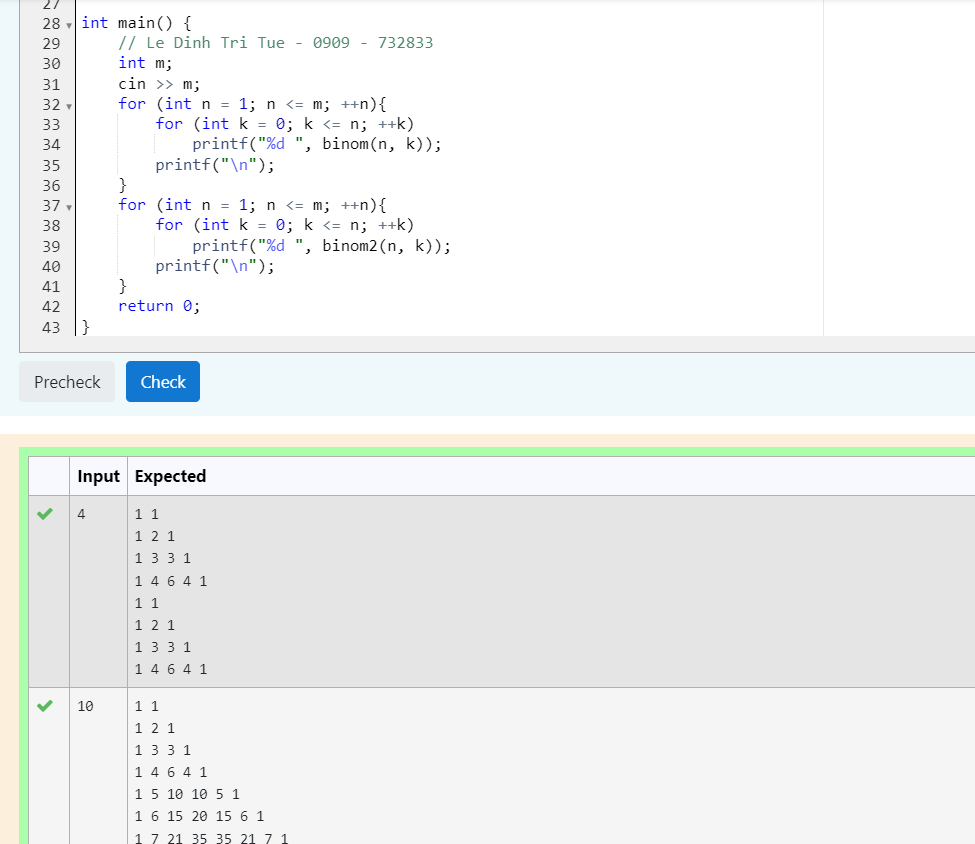
## **Bài tập 3.5.** Tính hệ số tổ hợp C(n, k).



Figure

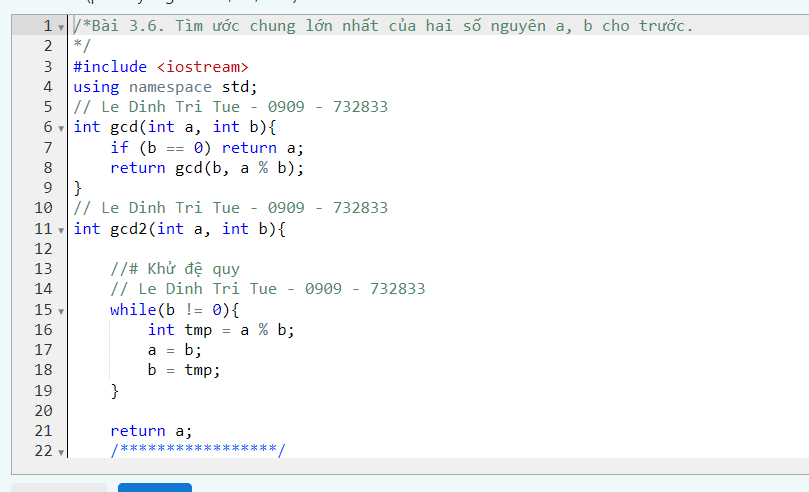


Figure

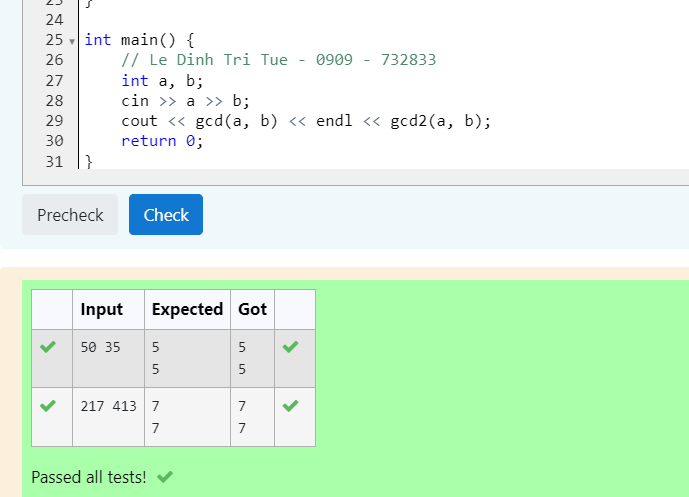


Figure

## **Bài tập 3.6.** Tìm ước chung lớn nhất của hai số nguyên a, b cho trước.

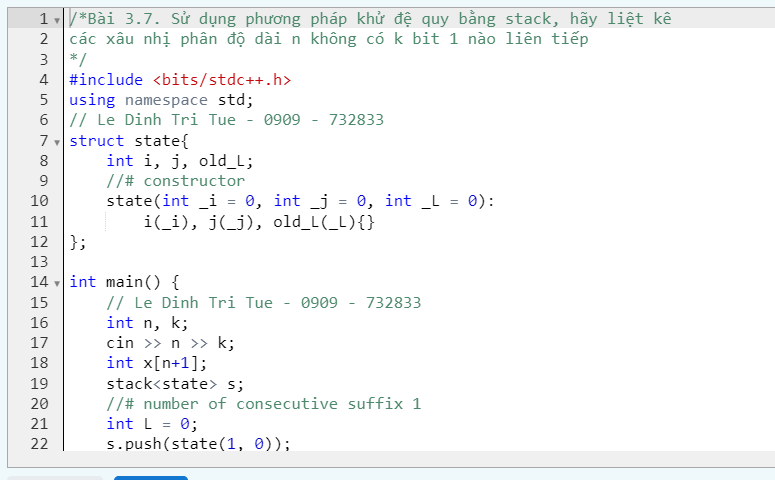


Figure

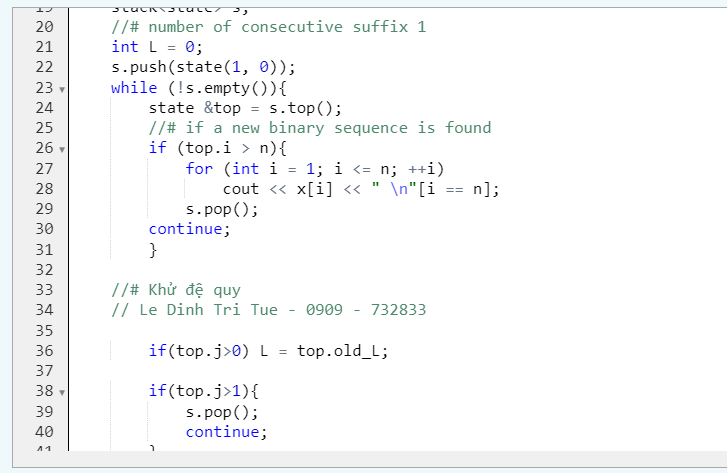


Figure

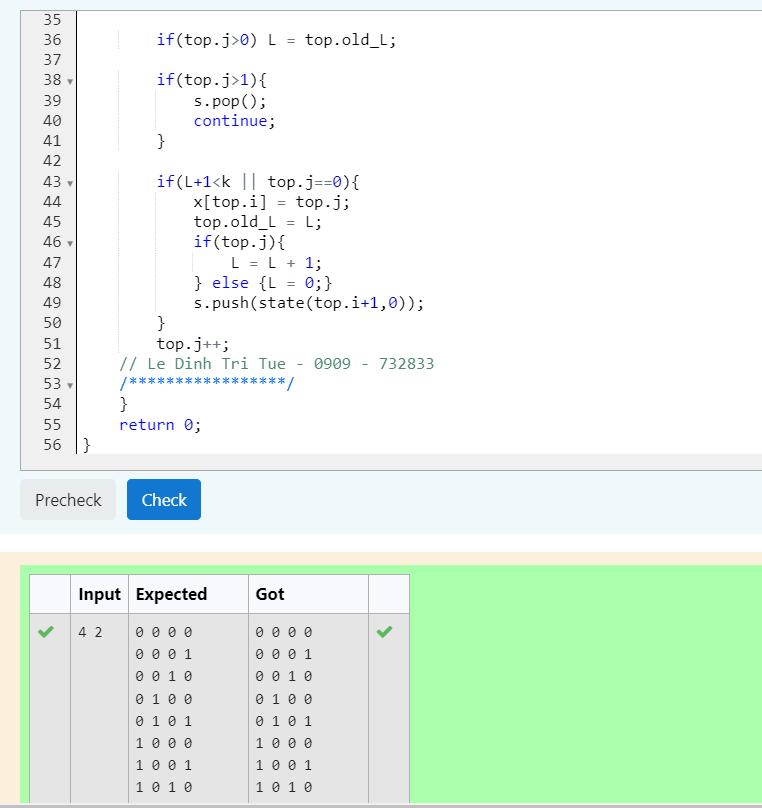
## **Bài tập 3.7.** Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp.



Figure



Figure



Figure

## **Bài tập 3.8.** Cân đĩa: Bạn đang muốn kiểm tra xem một vật cho trước có đúng nặng M như người ta nói hay không. Có một cân thăng bằng và n quả cân. Quả thứ i nặng mi. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã nguồn dưới.

**CODE:**

/\*

Bài tập 8: Cân đĩa

Bạn đang muốn kiểm tra xem một vật cho trước có đúng nặng M

như người ta nói hay không. Có một cân thăng bằng và n

quả cân. Quả thứ i nặng mi. Hãy chỉ ra một cách cân thỏa mãn.

Quy cách in ra đã được tích hợp trong mã nguồn dưới.\*/

#include <bits/stdc++.h>

using namespace std;

// Le Dinh Tri Tue - 0909 - 732833

struct state{

int i, j, s;

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

};

// Le Dinh Tri Tue - 0909 - 732833

int main() {

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

// Le Dinh Tri Tue - 0909 - 732833

int sum = 0;

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

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

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

while (!s.empty()){

state top = s.top();

if (top.i >= n){

if (top.s == 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;

}

// Le Dinh Tri Tue - 0909 - 732833

//# Khử đệ 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));

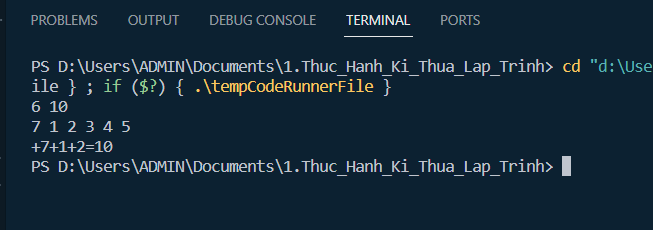
s.push(state(top.i+1, 1,top.s+m[top.i+1]));

}

cout << -1;

return 0;

}



## **Bài tập 3.9.** Lập lịch cho y tá: Một y tá cần lập lịch làm việc trong N 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]. Hãy liệt kê tất cả các cách lập lịch tốt, với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài n với bit 0/1 tương ứng là nghỉ/làm việc. Các xâu phải được in ra theo thứ tự từ điển.

**CODE:**

/\*

Bài tập 9: Lập lịch cho y tá

Một y tá cần lập lịch làm việc trong N 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].

Hãy liệt kê tất cả các cách lập lịch tốt,

với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài n

với bit 0/1 tương ứng là nghỉ/làm việc.

Các xâu phải được in ra theo thứ tự từ điển\*/

// Le Dinh Tri Tue - 0909 - 732833

#include<bits/stdc++.h>

using namespace std;

const int MAX = 1000;

const int du = 1000000000 + 7;

int n, k1, k2;

int x[MAX];

int cnt = 0;

int so0 = 0, so1 = 0;

// Le Dinh Tri Tue - 0909 - 732833

void inputData(){

cin >> n >> k1 >> k2;

}

bool check(int a, int i){

if(a==1) return true;

// Le Dinh Tri Tue - 0909 - 732833

else {

// neu ngay nay nghi -> kiem tra ngay truoc do, neu da nghi thi khong duoc nghi nua

if(i==0){

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

else { // x[a-1] == 1

if(so1<k1) return false;

}

} else { // i == 1

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

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

} else {

if(so1>=k2) return false;

}

}

// Le Dinh Tri Tue - 0909 - 732833

return true;

}

}

void solution(){

// Le Dinh Tri Tue - 0909 - 732833

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

cout << x[i];

cout << endl;

cnt++;

}

// Le Dinh Tri Tue - 0909 - 732833

void TRY(int a){

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

if(check(a,i)){

x[a] = i; // ngay thu a lam viec hoac khong lam viec

int pre = so1;

if(i == 1){

if(x[a-1] == 1) so1++;

else so1 = 1;

} else {

so1 = 0;

}

if(a==n) solution();

else TRY(a+1);

so1 = pre;

}

}

}

// Le Dinh Tri Tue - 0909 - 732833

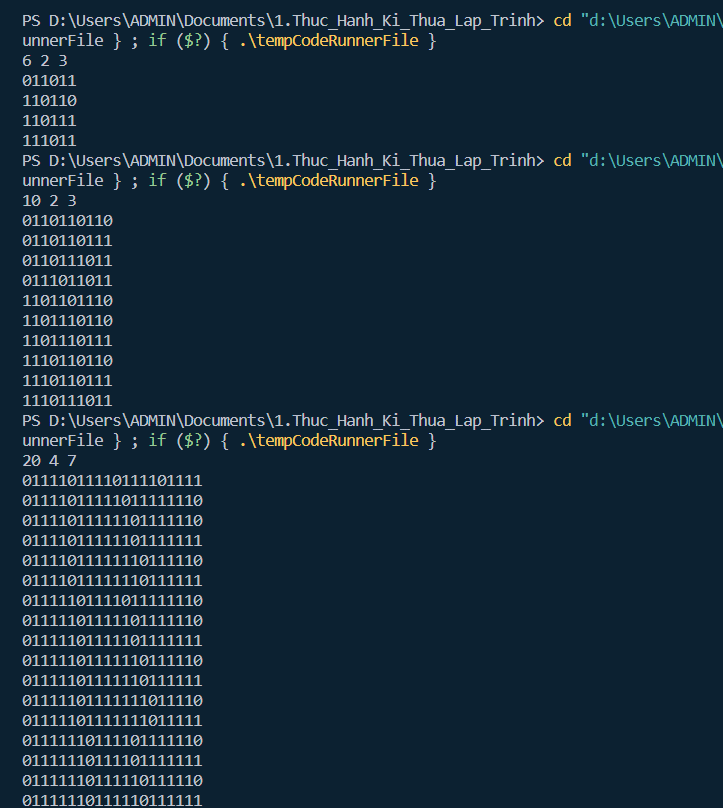
int main(){

inputData();

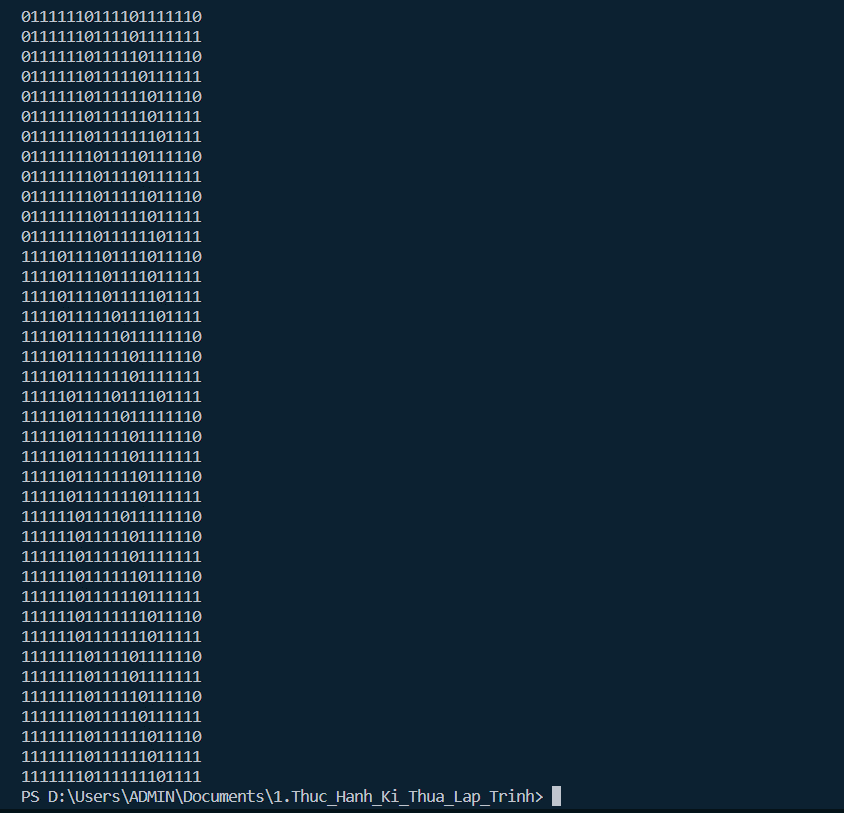
TRY(1);

return 0;

}



Figure



Figure

## **Bài tập 3.10.** 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 S là xâu gồm n ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài n, có khoảng cách Hamming với S bằng H. Các xâu phải được liệt kê theo thứ tự từ điển.

**CODE:**

/\*

Bài tập 10: 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 S là xâu gồm n

ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài n, có

khoảng cách Hamming với S bằng H. Các xâu phải được liệt kê theo thứ tự từ điển\*/

#include<bits/stdc++.h>

using namespace std;

const int MAX = 20;

int N, H;

int x[MAX];

int S[MAX]; // Xau mac dinh de so sanh

int cnt = 0;

// Le Dinh Tri Tue - 0909 - 732833

void input(){

cin >> N >> H;

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

S[i] = 0;

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

x[i] = 0;

}

// Le Dinh Tri Tue - 0909 - 732833

// Kiem tra 2 co khoang cach Hamming la bao nhieu

int checkHamming(int str1[], int str2[]){

int lens = N;

int cnt = 0;

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

if(str1[i] != str2[i]) cnt++;

}

return cnt;

}

// Le Dinh Tri Tue - 0909 - 732833

bool check(int a, int i){

return true;

}

// Le Dinh Tri Tue - 0909 - 732833

void solution(){

if(checkHamming(x,S) == H){

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

cout << x[i];

cout << endl;

}

}

// Le Dinh Tri Tue - 0909 - 732833

void TRY(int a){

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

x[a] = i;

if(a == N-1) solution();

else

TRY(a+1);

}

}

// Le Dinh Tri Tue - 0909 - 732833

int main(){

int T;

cin >> T;

while(T > 0){

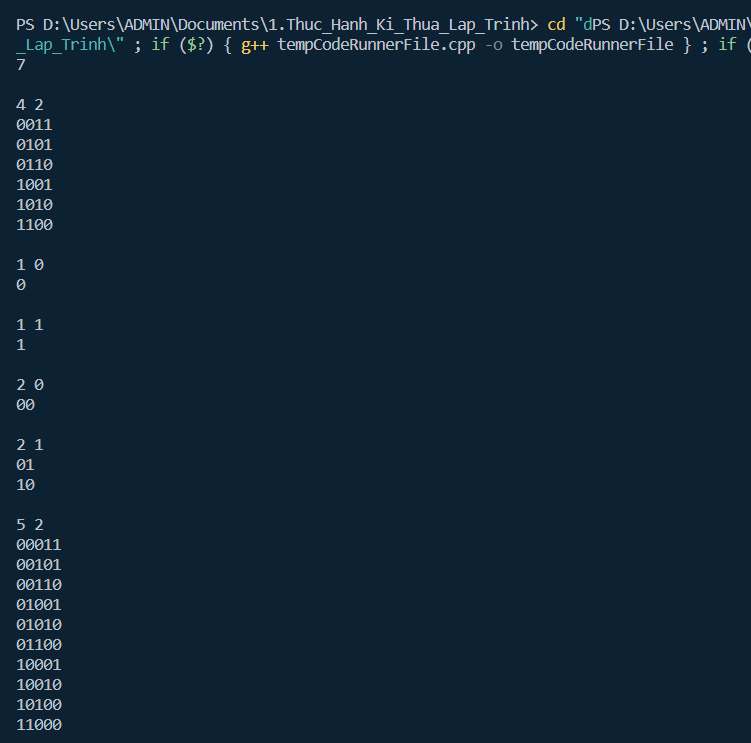
input();

TRY(0);

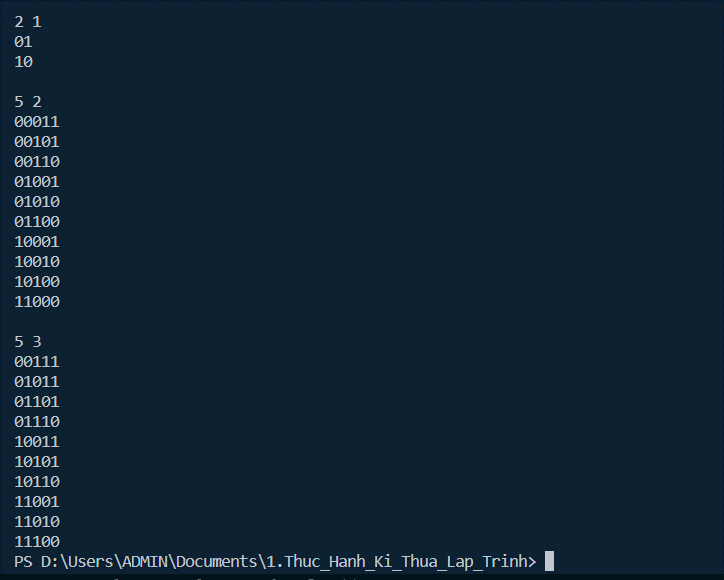
T--;

}

}



Figure



Figure

## **Bài tập 3.11.** Lịch trình chụp ảnh Superior là một hòn đảo tuyệt đẹp với n địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau. Đoàn khách tham quan có r 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.

**CODE:**

/\*Bài tập 11: Lịch trình chụp ảnh Superior là một hòn đảo tuyệt đẹp với n

địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau.

Đoàn khách tham quan có r 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.\*/

// Le Dinh Tri Tue - 0909 - 732833

#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 min\_price;

int sum\_price;

int start, destination, numberOfPoint;

// Le Dinh Tri Tue - 0909 - 732833

void input(){

cin >> n >> r;

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

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

cin >> price[i][j];

}

}

// Le Dinh Tri Tue - 0909 - 732833

bool check(int a, int i){

if(visited[vt[i]]) return false;

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

return true;

}

// Le Dinh Tri Tue - 0909 - 732833

void solution(){

if(price[x[numberOfPoint-2]][destination] == 0) return;

min\_price = min(min\_price, sum\_price + price[x[numberOfPoint-2]][destination]);

}

// Le Dinh Tri Tue - 0909 - 732833

void TRY(int a){

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

if(check(a, i)){

visited[vt[i]] = true;

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

x[a] = vt[i];

if(a == numberOfPoint-2) solution();

else TRY(a+1);

visited[vt[i]] = false;

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

}

}

}

// Le Dinh Tri Tue - 0909 - 732833

int main(){

string str;

input(); getline(cin,str);

while(r > 0){

min\_price = INT\_MAX;

sum\_price = 0;

// Le Dinh Tri Tue - 0909 - 732833

getline(cin, str);

while (!str.empty()){

stringstream convert(str.substr(0, str.find(" ")));

int tmp = 0;

convert >> tmp;

vt.push\_back(tmp - 1);

if (str.find(" ") > str.size()){

break;

} else {

str.erase(0, str.find(" ") + 1); // Update string

}

}

// Le Dinh Tri Tue - 0909 - 732833

// Bat dau khoi tao cac du lieu can thiet truoc khi quay lui

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

destination = vt[vt.size()-1]; // diem dich

numberOfPoint = vt.size(); // so diem phai di qua

x[0] = start; x[numberOfPoint-1] = destination;

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

visited[i] = false;

TRY(1);

// Le Dinh Tri Tue - 0909 - 732833

// In ra ket qua

if(min\_price == INT\_MAX) cout << "0" << endl;

else cout << min\_price << endl;

// Xoa vector va chuyen sang khach tiep theo

vt.erase(vt.begin(), vt.end());

r--;

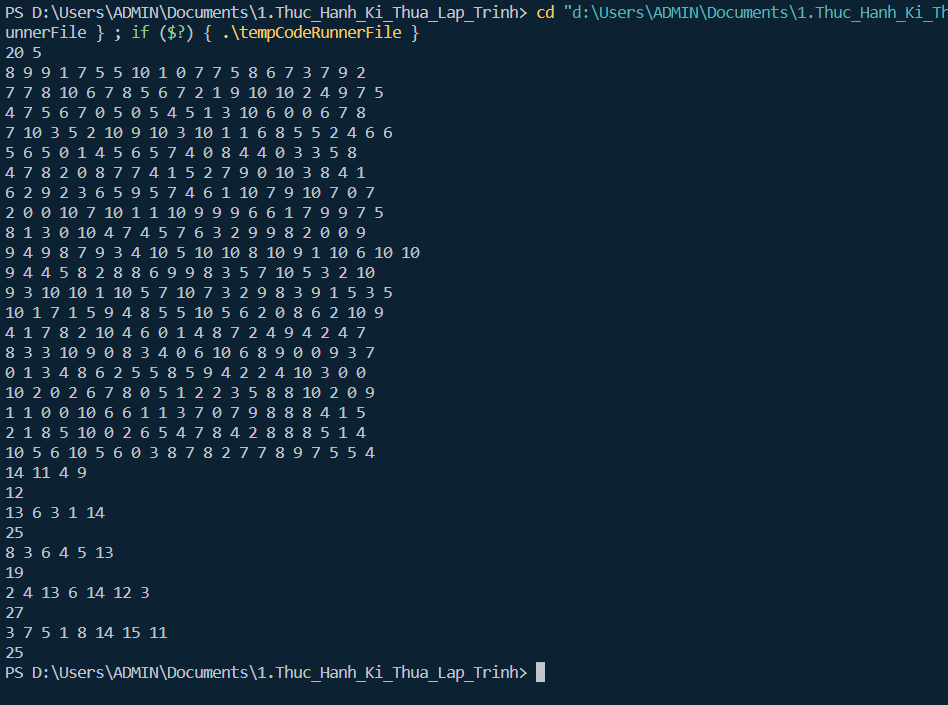
}

}

A screen shot of a computer code

Description automatically generated

Figure



Figure

## **Bài tập 3.12.** Đếm đường đi Cho đồ thị vô hướng G, hãy đếm số đường đi đi qua k cạnh và không đi qua đỉnh nào quá một lần.

**CODE:**

/\*Bài tập 12: Đếm đường đi Cho đồ thị vô hướng G, hãy đếm số đường đi đi qua k

cạnh và không đi qua đỉnh nào quá một lần.\*/

#include<bits/stdc++.h>

using namespace std;

const int MAX = 100;

int n, k;

int m;

vector<vector<int> > vt; // chuyen tu luu tru cac canh sang luu tru danh sach ke

int x[MAX];

bool visited[MAX];

int res;

// Le Dinh Tri Tue - 0909 - 732833

void input(){

cin >> n >> k;

cin >> m;

vt.resize(n+1);

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

int tmp1, tmp2;

cin >> tmp1 >> tmp2;

vt[tmp1-1].push\_back(tmp2-1); // dinh tmp2 ke voi dinh tmp1

vt[tmp2-1].push\_back(tmp1-1); // dinh tmp1 ke voi dinh tmp2

}

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

visited[i] = false;

}

res = 0;

}

// Le Dinh Tri Tue - 0909 - 732833

bool check(int a, int i){

if(a == 0) return true;

if(visited[i]) return false;

int index = 0;

for(int j=0; j<vt[x[a-1]].size(); j++){

if(i == vt[x[a-1]][j]) index++;

}

if(index == 0) return false;

return true;

}

// Le Dinh Tri Tue - 0909 - 732833

void solution(){

res++;

}

void TRY(int a){

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

if(check(a, i)){

visited[i] = true;

x[a] = i;

if(a == k) solution();

else TRY(a+1);

visited[i] = false;

}

}

}

// Le Dinh Tri Tue - 0909 - 732833

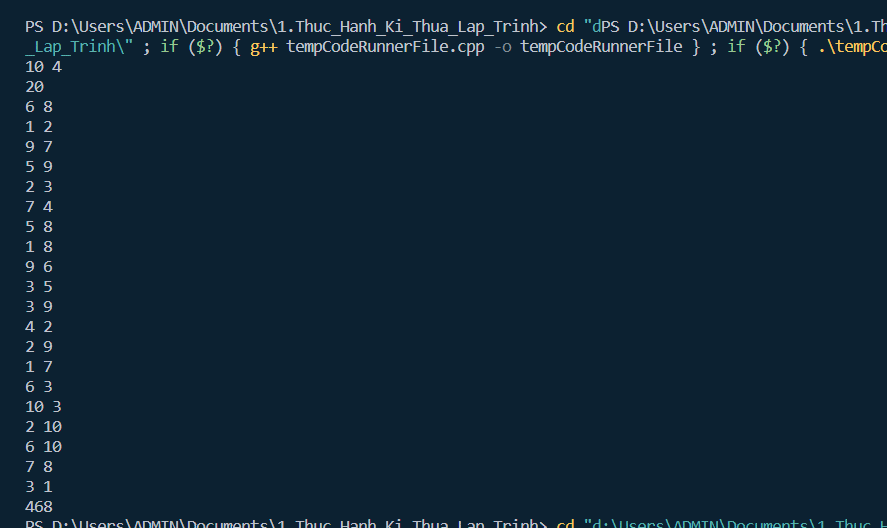
int main(){

input();

TRY(0);

cout << res / 2;

}



Figure

A screen shot of a computer

Description automatically generated

Figure

A computer screen shot of a code

Description automatically generated

Figure

A blue screen with white text

Description automatically generated

Figure