# BÁO CÁO THỰC HÀNH 03 – TUẦN 12

**CÁCH ĐÁNH GIÁ ĐIỂM THỰC HÀNH**

**HỌC PHẦN: IT3040 - KỸ THUẬT LẬP TRÌNH – 20221**

1. **Quy định, yêu cầu:** 
   * Tài liệu và nội dung thực hành chấm điểm trên hệ thống: <http://www.bkict.org/moodle>
   * Bài tập trên lớp chấm điểm tự động (các bài không chấm trên hệ thống làm vào máy tính  làm báo cáo thực hành – Theo mẫu).
   * Hạn nộp báo cáo trên Teams (Bài tập trên lớp + Bài tập về nhà): 1 tuần.
   * Hạn nộp bài tập về nhà chấm điểm tự động trên hệ thống: 2 tuần.
2. **Đánh giá điểm thực hành**

1. Chuyên cần (đúng giờ, nghiêm túc trong giờ học) - Điểm danh trên Teams: 10%

2. Báo cáo thực hành (bài tập trên lớp + Về nhà) theo mẫu nộp trên Teams: 20%

3. Trắc nghiệm – Form trên Teams: 10%

4. Bài tập về nhà hệ thống chấm tự động: 40%

5. Kiểm tra thực hành: 20%. (Tiết 3 buổi thực hành thứ 5).

**Điểm thưởng: +1 (Cho Mục 1,2 điểm TB từ 9-10).**

Tham gia thực hành đúng giờ đầy đủ theo thời khóa biểu (nếu có lý do không đi thực hành đúng kíp được thì gửi mail xin phép thực hành bù trước 1 ngày qua mail [hoalt@soict.hust.edu.vn](mailto:hoalt@soict.hust.edu.vn), Tiêu đề: đăng ký học bù – IT3040 – MaLopTH.

Các kíp có thể bù:

Table

Description automatically generated

**Nếu nghỉ không có lý do 3 buổi, không thực hành bù thì điểm chuyên cần, báo cáo và BTVN coi như 0 điểm thực hành.**

Contents

[BÁO CÁO THỰC HÀNH 03 – TUẦN 12 1](#_Toc122727906)

[Bài thực hành số 3 – Tuần 12 4](#_Toc122727907)

[**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](#_Toc122727908)

[**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). 6](#_Toc122727909)

[**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 7](#_Toc122727910)

[**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. 8](#_Toc122727911)

[**Bài tập 3.5.** Tính hệ số tổ hợp C(n, k) 9](#_Toc122727912)

[**Bài tập 3.6.** Tìm ước chung lớn nhất của hai số nguyên a, b cho trước. 10](#_Toc122727913)

[**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 11](#_Toc122727914)

[**Bài tập 3.8.** 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 ii nặng mimi. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã nguồn dưới 12](#_Toc122727915)

[**Bài tập 3.9.** 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 14](#_Toc122727916)

[**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 16](#_Toc122727917)

[**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. 18](#_Toc122727918)

[**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. 21](#_Toc122727919)

DANH MỤC HÌNH ẢNH

[Hình ảnh 1 - Code bài 3.1 4](#_Toc122727942)

[Hình ảnh 2 - Kết quả bài 3.1 5](#_Toc122727943)

[Hình ảnh 3 - Code bài 3.2 6](#_Toc122727944)

[Hình ảnh 4 - Code bài 3.3 7](#_Toc122727945)

[Hình ảnh 5 -Code bài 3.4 8](#_Toc122727946)

[Hình ảnh 6 – Code bài 3.5 9](#_Toc122727947)

[hình ảnh 7 – code bài 3.6 10](#_Toc122727948)

[Hình ảnh 8 – code bài 3.7 11](#_Toc122727949)

[hình ảnh 9 – kết quả bài 3.8 13](#_Toc122727950)

[hình ảnh 10 – kết qủa bài 3.9 15](#_Toc122727951)

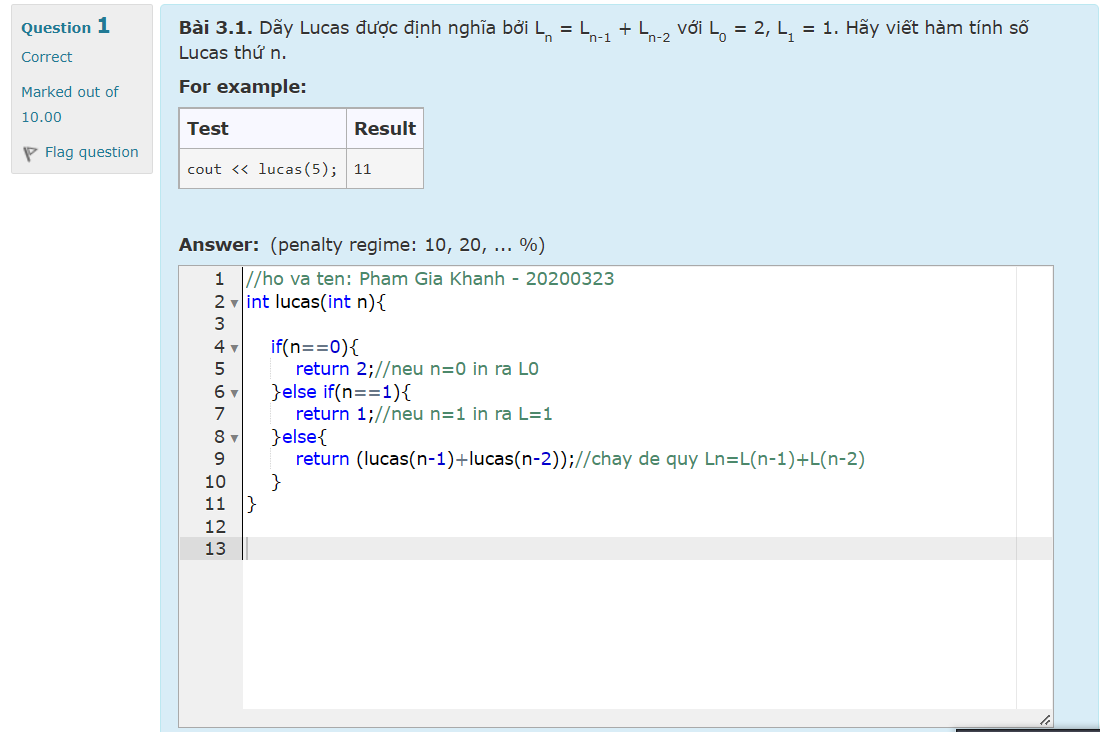
[hình ảnh 11 – kết quả bài 3.10 17](#_Toc122727952)

[hình ảnh 12 – kết quả bài 3.11 20](#_Toc122727953)

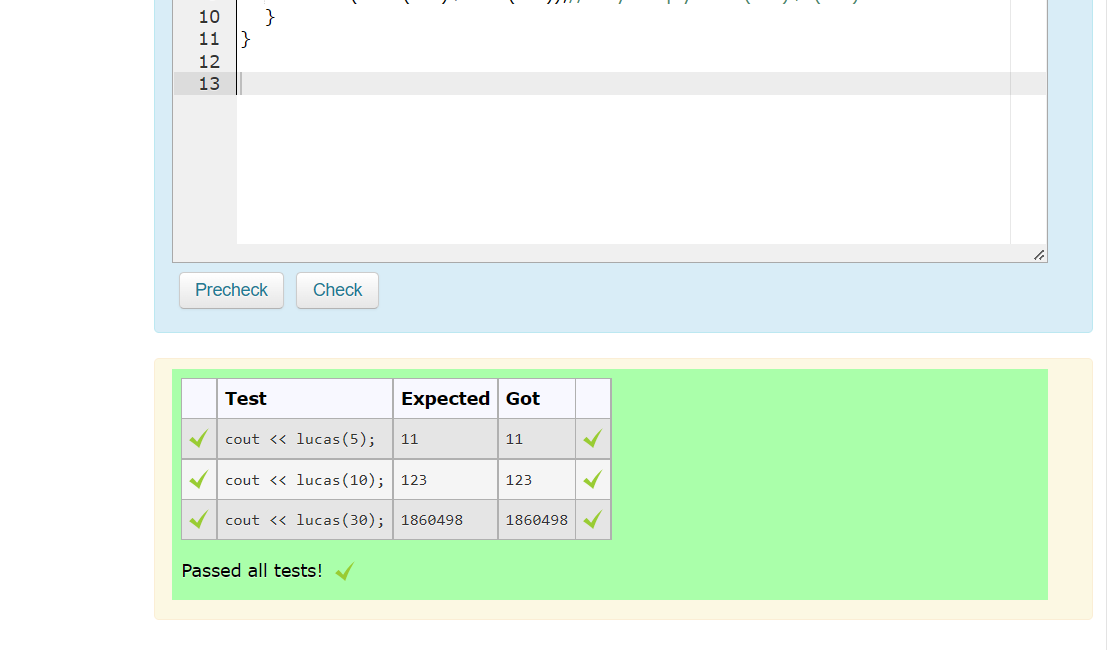
[hình ảnh 13 – kết quả bài 3.12 22](#_Toc122727954)

# Bài thực hành số 3 – Tuần 12

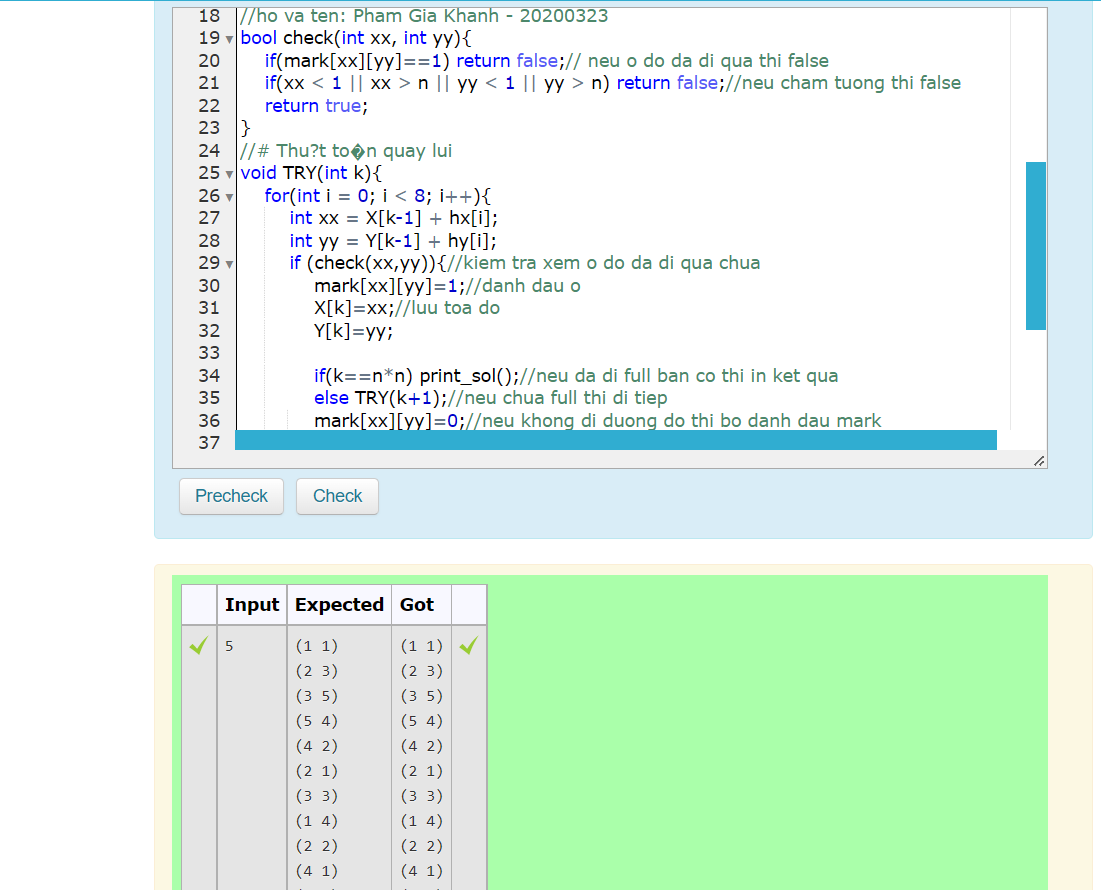
## **Bài tập 3.1.** Dãy Lucas được định nghĩa bởi Ln = Ln-1 + Ln-2 với L0 = 2, L1 = 1. Hãy viết hàm tính số Lucas thứ n.



Hình ảnh 1 - Code bài 3.1

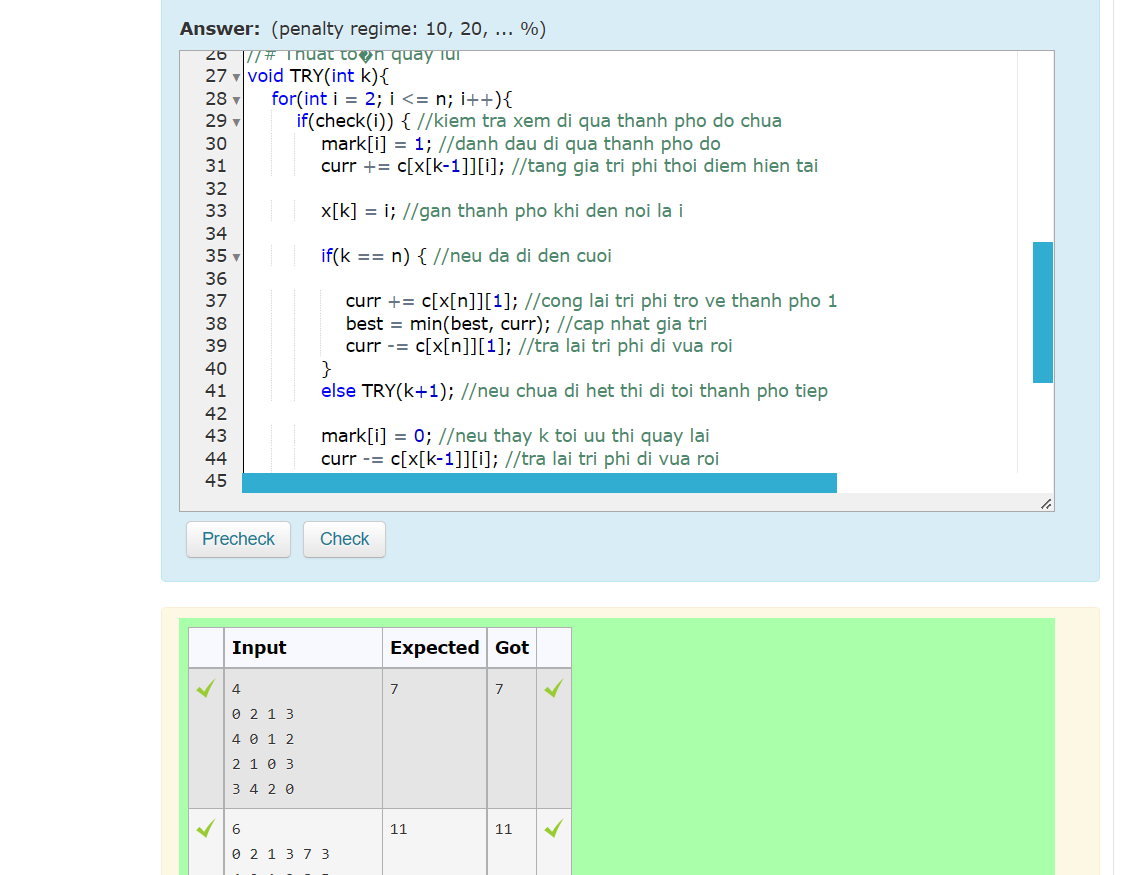


Hình ảnh 2 - Kết quả bài 3.1

**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). 

Hình ảnh 3 - Code bài 3.2

## **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



Hình ảnh 4 - Code bài 3.3

## **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.

Hình ảnh 5 -Code bài 3.4

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

Hình ảnh 6 – Code bài 3.5

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



hình ảnh 7 – code bài 3.6

## **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

Hình ảnh 8 – code bài 3.7

## **Bài tập 3.8.** 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 ii nặng mimi. 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;

struct *state*{

    int i, j, s;

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

};

//Pham Gia Khánh - 20200323

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

    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;

        }

        //#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));

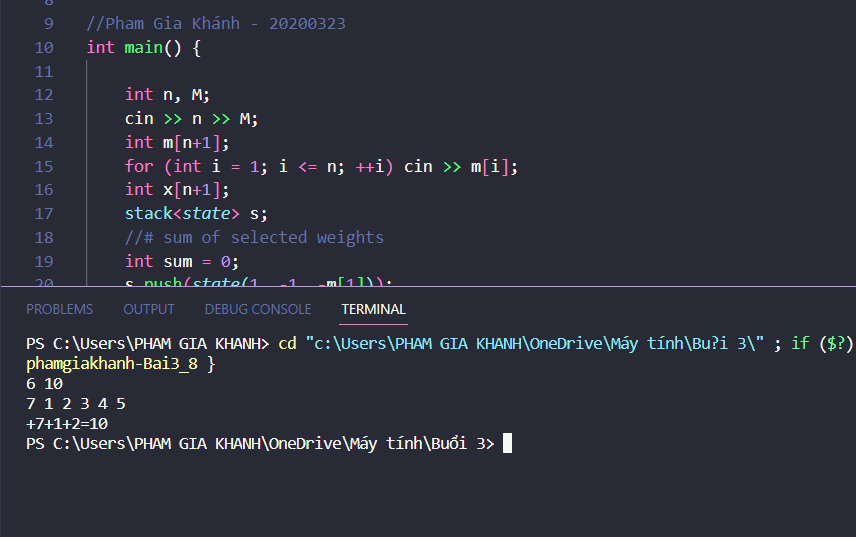
        s.push(*state*(top.i+1, 1,top.s+m[top.i+1]));

    }

    cout << -1;

    return 0;

}



hình ảnh 9 – kết quả bài 3.8

## **Bài tập 3.9.** 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

#include <iostream>

using namespace std;

//Pham Gia Khanh - 20200323

int n, k1, k2;

int x[100001];

int work = 0;

void input() {

    cin >> n >> k1 >> k2;

}

bool check(int *a*, int *i*) {

    if(*a* == 1) return true; //ngay dau tien chua xet

    else {

        if(*i* == 0) {

            if(x[*a*-1] == 0) return false; //neu ngay truoc da nghi

            else if(work < k1) return false; //chua lam du ngay lien tuc

        }

        else if(*i* == 1) {

            if(x[*a*-1] == 0) {

                if(n - *a* + 1 < k1) return false; //ngay cuoi cung ma chua du ngay lam viec lien tiep

            }

            else if(work >= k2) return false; //vuot qua so ngay lam viec

        }

        return true;

    }

}

void output() {

    for(int i = 1; i <= n; i++)

        cout << x[i];

    cout << '\n';

}

void TRY(int *a*) {

    for(int i = 0; i <= 1; i++) {

        if(check(*a*, i)) {

            x[*a*] = i;

            int pre = work;

            if(i == 1) {

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

                else work = 1;

            }

            else {

                work = 0;

            }

            if(*a* == n) output();

            else TRY(*a*+1);

            work = pre;

        }

    }

}

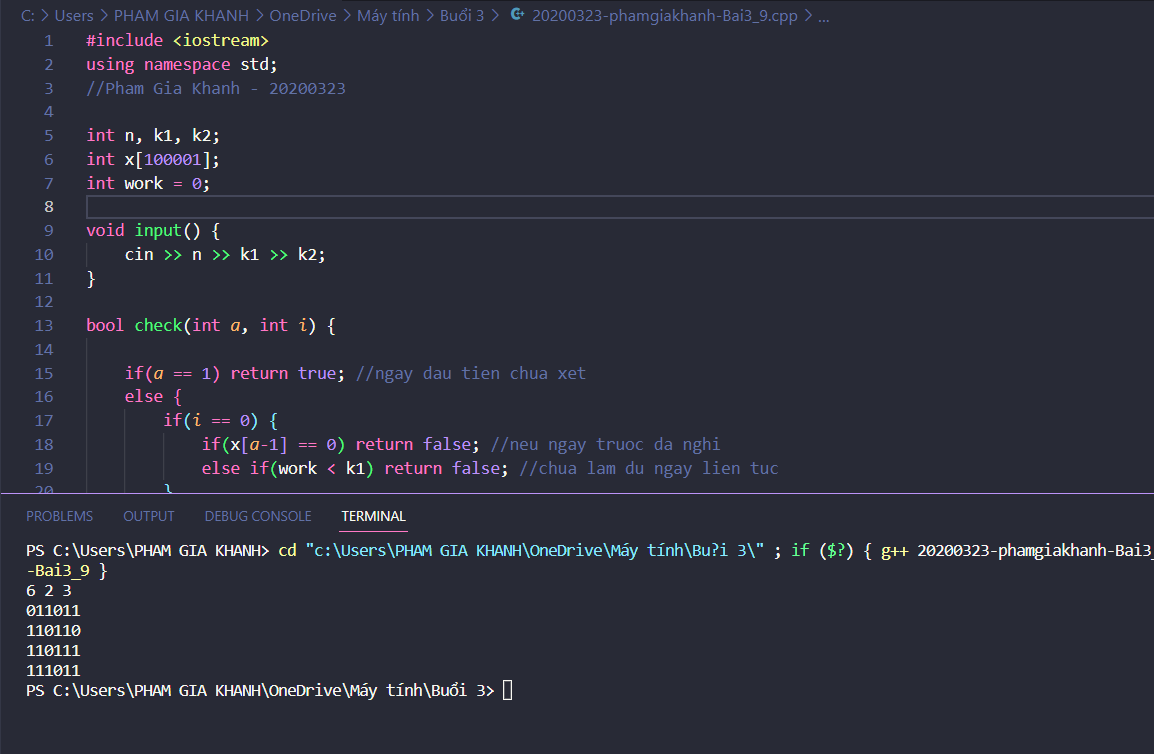
int main() {

    input();

    TRY(1);

    return 0;

}



hình ảnh 10 – kết qủa bài 3.9

## **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

#include<iostream>

using namespace std;

//Pham Gia Khanh - 20200323

int N, H;

int s[10001]; //xau mac dinh 000...

int t[10001]; //xau test

void input() {

    cin >> N >> H;

    for(int i = 0; i < N; i++) {

        s[i] = 0;

        t[i] = 0;

    }

}

//kiem tra co bao nhieu xau ki tu khac nhau

int checkHamming(int *s1*[], int *s2*[]) {

    int count = 0, lens = N;

    for(int i = 0; i< lens; i++) {

        if(*s1*[i] != *s2*[i]) count++;

    }

    return count;

}

void output() {

    if(checkHamming(s, t) == H) {

        for(int i = 0; i < N; i++) {

            cout << t[i];

        }

        cout << '\n';

    }

}

//liet ke cac day nhi phan

void TRY(int *a*) {

    for(int i = 0; i <=1; i++) {

        t[*a*] = i;

        if(*a* == N - 1) output();

        else TRY(*a* + 1);

    }

}

int main(){

    int T;

    cin >> T;

    while(T > 0){

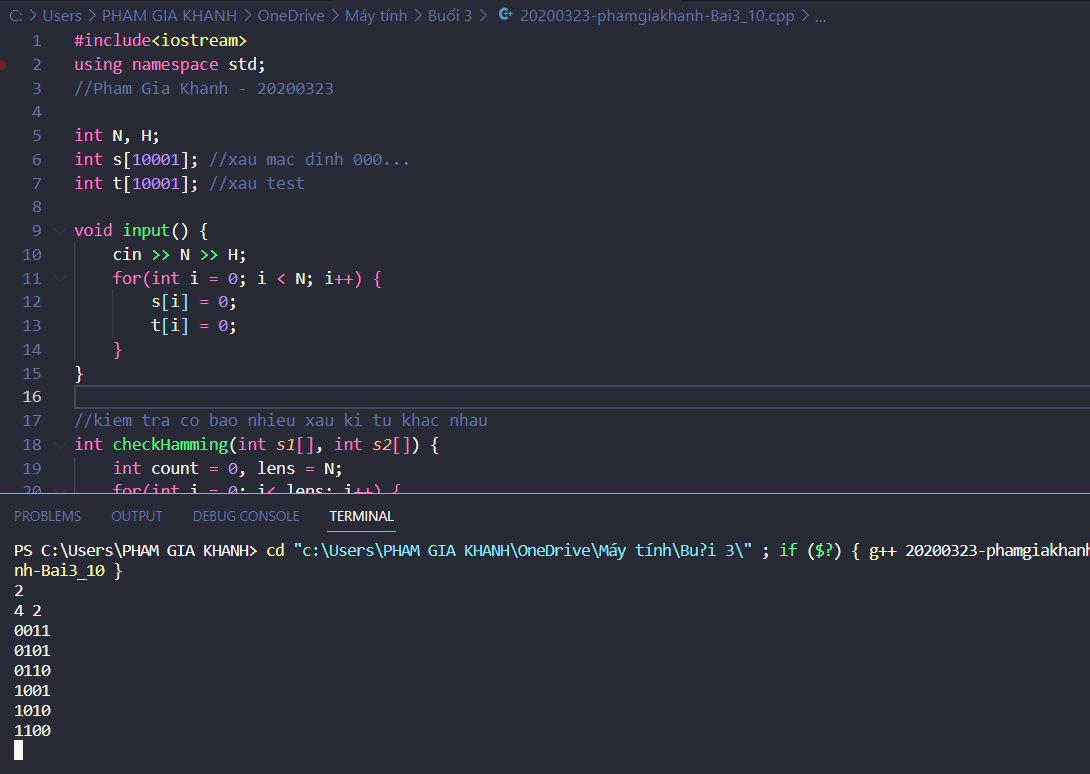
        input();

        TRY(0);

        T--;

    }

}



hình ảnh 11 – kết quả bài 3.10

## **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.

#include<bits/stdc++.h>

using namespace std;

//Phạm gia khánh - 20200323

int n, r;

int price[10001][10001];

int x[10001];

bool visited[10001]; //mảng lưu các địa điểm đã đi

vector<int> vt;

int min\_price; //chi phí nhỏ nhất

int sum\_price; //tổng chi phí

int start, destination, numberOfPoint;

void input(){

    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; //nếu đã đi thì false

    if(price[x[*a*-1]][vt[*i*]] == 0) return false; //không đi được

    return true;

}

void solution(){

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

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

}

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]];

        }

    }

}

int main(){

*string* str;

    input(); getline(cin,str);

    while(r > 0){

        min\_price = INT\_MAX;

        sum\_price = 0;

        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

            }

        }

        start = vt[0]; // điểm bắt đầu

        destination = vt[vt.size()-1]; // điểm đích

        numberOfPoint = vt.size(); // số điểm đi qua

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

        for(int i=0; i<n; i++)

            visited[i] = false;

        TRY(1);

        // In ra ket qua

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

        else cout << min\_price << endl;

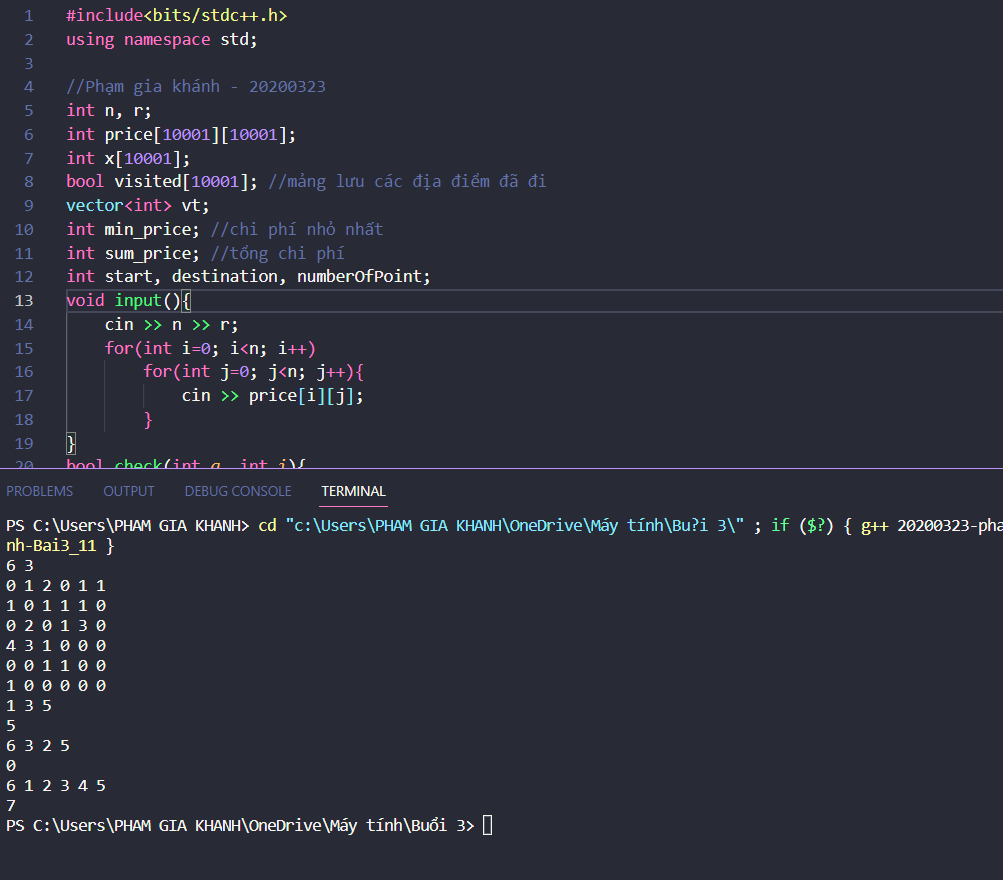
        // xóa vector chuyển sang khách tiếp theo

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

        r--;

    }

}



hình ảnh 12 – kết quả bài 3.11

## **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.

#include<bits/stdc++.h>

using namespace std;

//Pham Gia Khánh - 20200323

int n, k;

int m;

vector<vector<int> > vt; //

int x[10001];

bool visited[10001];

int res;

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); // đỉnh tmp2 kề với đỉnh tmp1

        vt[tmp2-1].push\_back(tmp1-1); // đỉnh tmp1 kề với đỉnh tmp2

    }

    for(int i = 0; i < n; i++){

        visited[i] = false;

    }

    res = 0;

}

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;

}

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;

        }

    }

}

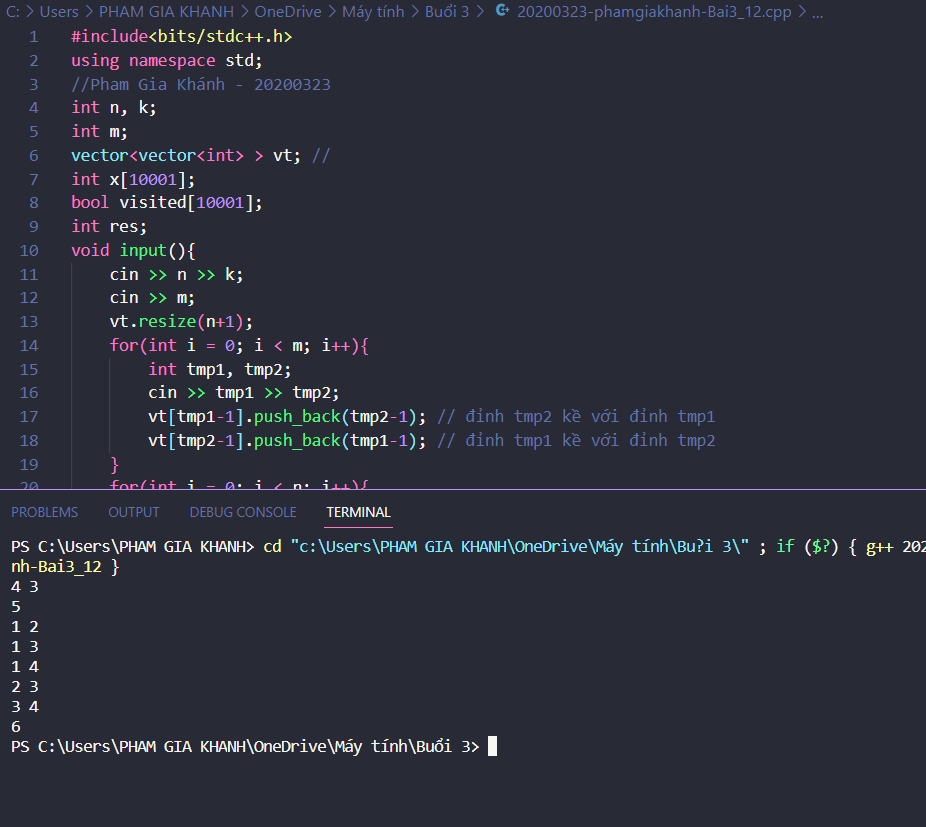
int main(){

    input();

    TRY(0);

    cout << res / 2;

}



hình ảnh 13 – kết quả bài 3.12