**CÁCH ĐÁNH GIÁ ĐIỂM THỰC HÀNH**

**HỌC PHẦN: IT3040 - KỸ THUẬT LẬP TRÌNH – 2022.2**

1. **Quy định, yêu cầu:** 
   * Tài liệu và nội dung thực hành chấm điểm trên hệ thống:

https://lab.soict.hust.edu.vn/

* + 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.

1. **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: 40%

3. Trắc nghiệm – Form trên Teams: 10%

4. Kiểm tra thực hành: 40%. (Tiết 2,3 buổi thực hành thứ 5).

**Điểm thưởng: 5% 🡪 10% (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:

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Thời gian, địa điểm, Tuần học** | **Mã nhóm** | **Mã lớp** |
| **1** | Sáng T4: Tiết 645-915 , Địa điểm: B1-203, Tuần: 32,34,36,38,40 |  | 727638 |
| **2** | Sáng T4: Tiết 645-915 , Địa điểm: B1-203, Tuần: 31,35,37,39,41 |  | 727634 |
| **3** | Sáng T4: Tiết 915-1145 , Địa điểm: B1-203, Tuần: 31,35,37,39,41 |  | 727635 |
| **4** | Sáng T4: Tiết 915-1145 , Địa điểm: B1-203, Tuần: 32,34,36,38,40 |  | 727639 |

**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.**

Mục Lục

[Bài thực hành số 3 – Tuần 36 3](#_Toc137679274)

[**Bài tập 3.1.** Tính dãy Lucas. 3](#_Toc137679275)

[**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). 4](#_Toc137679276)

[**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 9](#_Toc137679277)

[**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. 13](#_Toc137679278)

[**Bài tập 3.5**. Tính hệ số tổ hợp C(n, k) 18](#_Toc137679279)

[**Bài tập 3.6**. Tìm ước chung lớn nhất của hai số nguyên a, b cho trước. 22](#_Toc137679280)

[**Bài tập 3.7**. Liệt kê xâu nhị phân: 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 24](#_Toc137679281)

[**Bài tập 3.8**. Cân đĩa 29](#_Toc137679282)

[**Bài tập 3.9**. Lập lịch cho y tá 32](#_Toc137679283)

[**Bài tập 3.10**. Khoảng cách Hamming 37](#_Toc137679284)

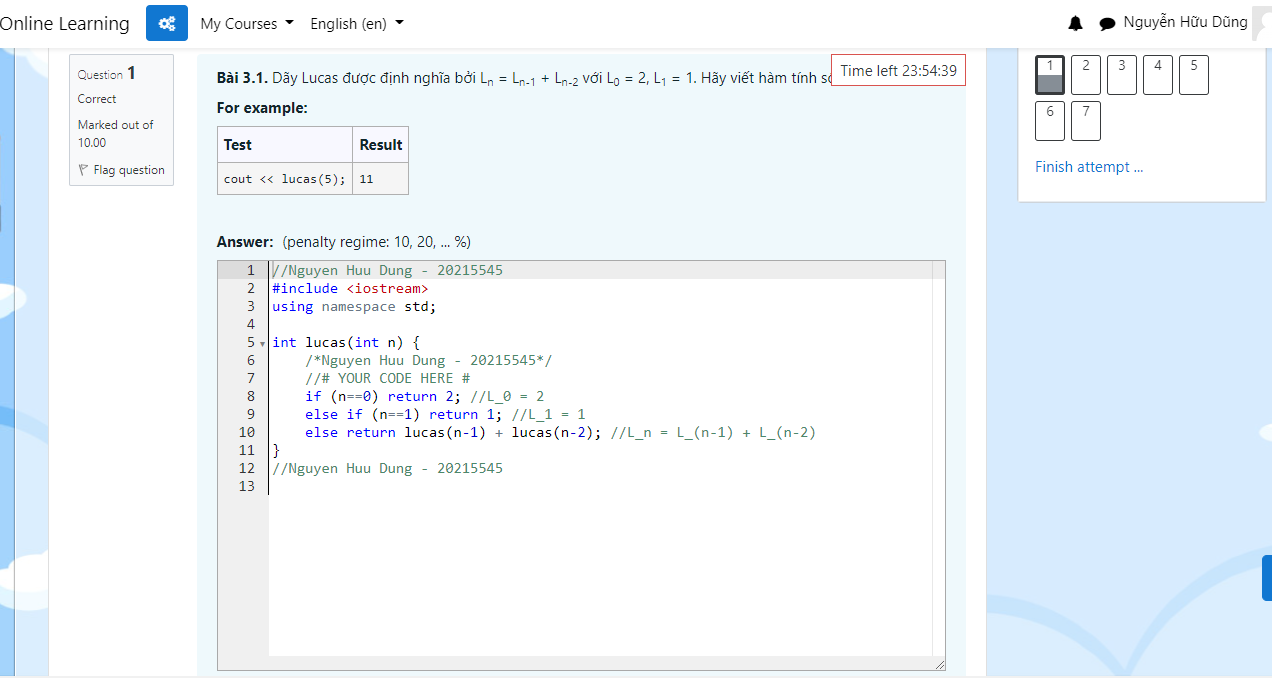
[**Bài tập 3.11**. Lịch trình chụp ảnh 41](#_Toc137679285)

[**Bài tập 3.12**. Đếm đường đi 47](#_Toc137679286)

# Bài thực hành số 3 – Tuần 36

## **Bài tập 3.1.** Tính dãy Lucas.

Dãy Lucas được định nghĩa bởi Ln = Ln-1 + Ln-2 và bắt đầu bởi L0 = 2, L1 = 1. Viết hàm tính số Lucas thứ n.



A screenshot of a computer

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#include <iostream>

using namespace std;

int lucas(int n) {

/\*Nguyen Huu Dung - 20215545\*/

//# YOUR CODE HERE #

if (n==0) return 2; //L\_0 = 2

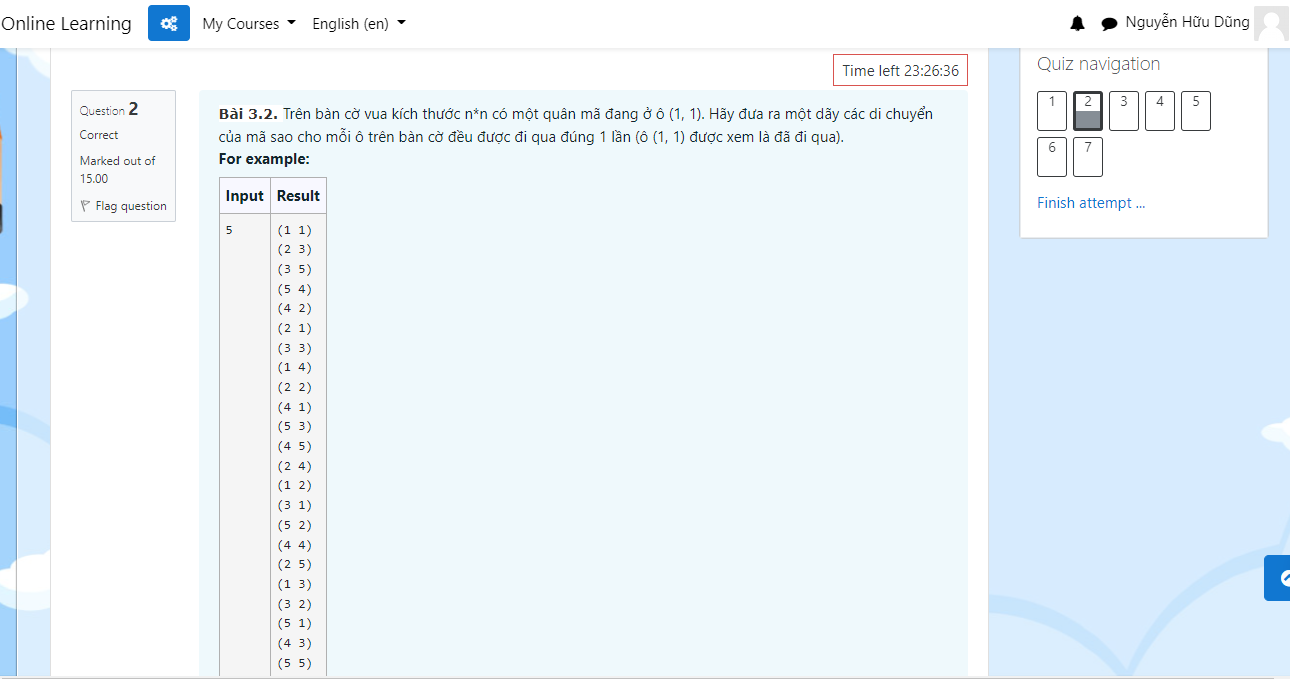
else if (n==1) return 1; //L\_1 = 1

else return lucas(n-1) + lucas(n-2); //L\_n = L\_(n-1) + L\_(n-2)

}

//Nguyen Huu Dung - 20215545

## **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).



A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#include <iostream>

using namespace std;

int n;

int X[100], Y[100]; //# Luu toa do cac buoc di chuyen cua quan ma

int mark[100][100]; //# Danh dau vi tri cac o ma quan ma da di chuyen qua

//# Mang hx, hy mo ta 8 vi tri quan ma co the di chuyen ke 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 cac di chuyen tim duoc

void print\_sol(){

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

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

}

exit(0);

}

//# Thuat toan 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];

/\*Nguyen Huu Dung - 20215545\*/

if (xx >= 1 && xx <= n && yy >= 1 && yy <= n && mark[xx][yy] != 1) {

//Neu toa do dang xet chua di qua va nam trong ban co thi

X[k] = xx;

Y[k] = yy;

//Cap nhat toa do hien tai

mark[X[k]][Y[k]] = 1; //Danh dau toa do hien tai da di qua

// Kiem tra neu da di qua tat ca cac o tren ban co

if (k == n \* n)

print\_sol();

else

TRY(k + 1);

// Tiep tuc xet den buoc tiep theo

mark[X[k]][Y[k]] = 0;

}

}

}

int main(){

cin >> n;

mark[1][1] = 1; //Vi tri ban dau la (1 1)

X[1] = Y[1] = 1; //Vi tri hien tai

TRY(2); //Bat dau tu buoc thu 2

return 0;

} //Nguyen Huu Dung - 20215545

## **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

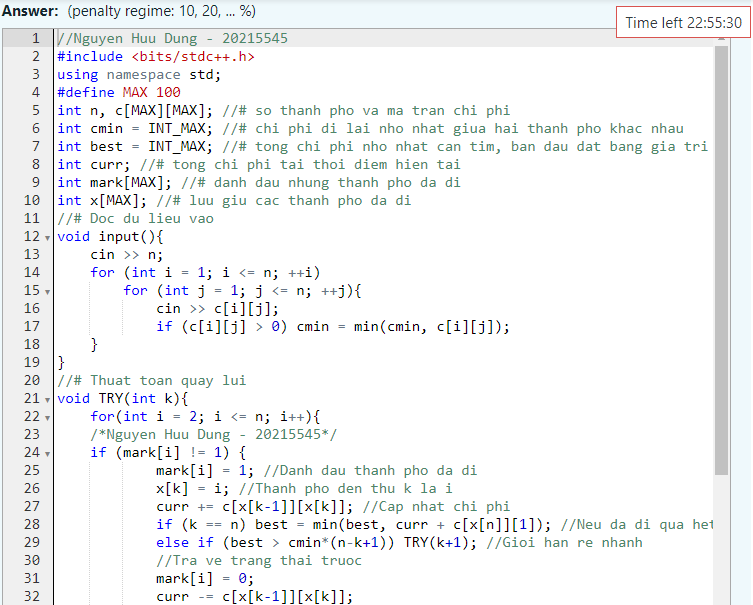
**Dữ liệu vào:**

* Dòng 1: Chứa số nguyên n (1 ≤ n ≤ 16)
* n dòng tiếp theo: Chứa ma trận c ( 0 ≤ cij ≤ 10000000)

**Kết quả:** Ghi tổng chi phí nhỏ nhất có thể

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a calendar

Description automatically generated

//Nguyen Huu Dung - 20215545

#include <bits/stdc++.h>

using namespace std;

#define MAX 100

int n, c[MAX][MAX]; //# so thanh pho va ma tran chi phi

int cmin = INT\_MAX; //# chi phi di lai nho nhat giua hai thanh pho khac nhau

int best = INT\_MAX; //# tong chi phi nho nhat can tim, ban dau dat bang gia tri vo cung lon INT\_MAX = 2^31-1

int curr; //# tong chi phi tai thoi diem hien tai

int mark[MAX]; //# danh dau nhung thanh pho da di

int x[MAX]; //# luu giu cac thanh pho da di

//# Doc du lieu vao

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]);

}

}

//# Thuat toan quay lui

void TRY(int k){

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

/\*Nguyen Huu Dung - 20215545\*/

if (mark[i] != 1) {

mark[i] = 1; //Danh dau thanh pho da di

x[k] = i; //Thanh pho den thu k la i

curr += c[x[k-1]][x[k]]; //Cap nhat chi phi

if (k == n) best = min(best, curr + c[x[n]][1]); //Neu da di qua het cac thanh pho

else if (best > cmin\*(n-k+1)) TRY(k+1); //Gioi han re nhanh

//Tra ve trang thai truoc

mark[i] = 0;

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

}

}

}

int main() {

input(); //Nhap vao

x[1] = 1; //Thanh pho 1 da di qua

TRY(2); //Bat dau tu buoc thu 2

cout << best; //In ra chi phi tot nhat

return 0;

} //Nguyen Huu Dung - 20215545

## **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.

**Dữ liệu vào:**

Dòng 1: Chứa số nguyên n (1≤ n ≤1000)

Dòng 2: Chứa n số nguyên a1, a2,..., an (|ai| ≤109)

**Kết quả:**

Dòng đầu tiên chứa độ dài dãy con tăng dài nhất

Dòng thứ 2 chứa chỉ số các phần tử được chọn vào dãy con đó

Nếu có nhiều dãy con tăng dài nhất, in ra dãy bất kỳ trong số đó

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

//Nguyen Huu Dung - 20215545

#include <bits/stdc++.h>

using namespace std;

int a[1000], n;

int mem[1000]; //# mang ghi nho loi giai cac bai toan con da duoc giai

void init(){

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

}

//# Quy hoach dong,

//# Ham lis(i) tra ve do dai day con tang dai nhat ket thuc boi a[i]

int lis(int i) {

/\*Nguyen Huu Dung - 20215545\*/

if (mem[i] != -1)

return mem[i]; //Neu da tim duoc day con tang dai nhat thi tra ve do dai day con

int res = 1; //Ban dau do dai day bang 1 (chi co a[i])

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

if (a[j] < a[i]){ //Neu tim duoc phan tu a[j] < a[i]

res = max(res, 1 + lis(j)); //thi tinh lai do dai

}

}

mem[i] = res;

return res;

}

//# 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]){

//Neu day ket thuc boi i co do dai = do dai day ket thuc boi j + 1

trace(j); //Truy vet day ket thuc boi j

break;

}

}

cout << a[i] << " "; //In ra phan tu cuoi

}

int main(){

init();

cin >> n;

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

int res = 1, pos = 0; //Ban dau do dai day bang 1, ket thuc boi a[0]

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

if (res < lis(i)){ //Tim do dai lon nhat va vi tri ket thuc

res = lis(i);

pos = i;

}

}

cout << res << endl; //In ra

trace(pos);

return 0;

} //Nguyen Huu Dung – 20215545

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

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#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 binom2(int n, int k){

//# Khu de quy

/\*Nguyen Huu Dung - 20215545\*/

//Cong thuc to hop: n!/(k!\*(n-k)!) = (n-k+1)\*...\*n/k!

long a = 1, b = 1; //a la mau so, b la tu so

for (int i = 1; i <= k; i++) a \*= i;

for (int i = n - k + 1; i <= n; i++) b \*= i;

return b/a;

}

int main() {

int m;

cin >> m;

for (int n = 1; n <= m; ++n){ //Giai theo cach de quy

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

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

printf("\n");

}

for (int n = 1; n <= m; ++n){ //Giai theo cach khong dung de quy

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

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

printf("\n");

}

return 0;

}//Nguyen Huu Dung - 20215545

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

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#include <iostream>

using namespace std;

int gcd(int a, int b){ //Tim uoc chung lon nhat bang de quy

if (b == 0) return a;

return gcd(b, a % b);

}

int gcd2(int a, int b) {

//# Khu de quy

/\*Nguyen Huu Dung - 20215545\*/

for (int i=a; i>0; i--) { //Do tu a den 1 de tim uoc chung lon nhat

if (a%i == 0 && b%i == 0) return i; //Tra ve uoc chung lon nhat

}

return 1;

}

int main() {

int a, b;

cin >> a >> b;

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

return 0;

}//Nguyen Huu Dung - 20215545

## **Bài tập 3.7**. Liệt kê xâu nhị phân: 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

**Dữ liệu vào:**

Một dòng duy nhất chứa hai số nguyên n k (1 ≤ k ≤ n ≤ 20)

**Kết quả:**

Với mỗi xâu tìm được, in ra n ký tự trên một dòng, các ký tự cách nhau bởi dấu cách.

Các xâu cần được liệt kê theo thứ tự từ điển

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

#include <bits/stdc++.h>

using namespace std;

struct state{

int i, j, old\_L;

//i : so vi tri hien tai trong xau

//j : gia tri cua bit hien tai

//old\_L : so bit 1 truoc do

//# constructor

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

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

};

int main() {

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;

}

//# Khử đệ quy

/\*Nguyen Huu Dung - 20215545\*/

if (top.j > 0) L = top.old\_L; //Neu bit hien tai >0 thi gan L=old.L

if (top.j > 1) { //Neu bit hien tai lon hon 1 thi nem ra

s.pop();

continue;

}

if (L+1 < k || top.j == 0) { //Neu L<k-1 hoac bit hien tai = 0

x[top.i] = top.j; //Luu bit hien tai vao mang x

top.old\_L = L; //So luong bit 1 lien tiep la L

if (top.j) L++; //Neu bit hien tai bang 1 thi L++

else L = 0; //Nguoc lai L=0

s.push(state(top.i+1,0)); //Them bit 0 vao

}

++top.j; //Bit hien tai tang 1 don vi

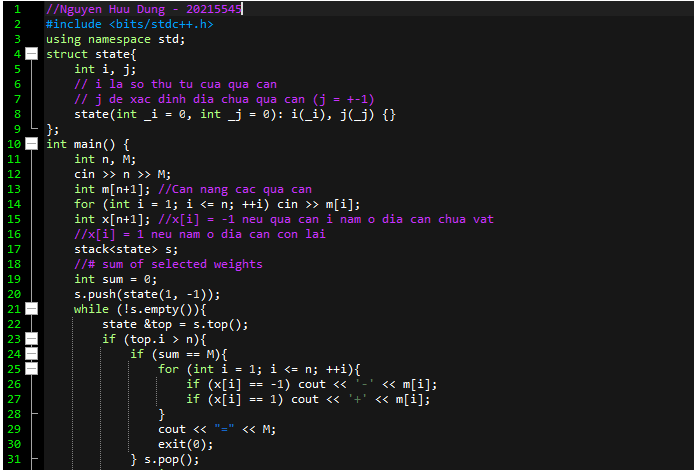
}

return 0;

} //Nguyen Huu Dung - 20215545

## **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.



A picture containing text, screenshot

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#include <bits/stdc++.h>

using namespace std;

struct state{

int i, j;

// i la so thu tu cua qua can

// j de xac dinh dia chua qua can (j = +-1)

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

};

int main() {

int n, M;

cin >> n >> M;

int m[n+1]; //Can nang cac qua can

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

int x[n+1]; //x[i] = -1 neu qua can i nam o dia can chua vat

//x[i] = 1 neu nam o dia can con lai

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;

}

//# Khu de quy

//Nguyen Huu Dung - 20215545

//# YOUR CODE HERE #

//Quay lui

if (top.j > -1) //Neu qua can nam o dia khong chua vat

sum -= m[top.i] \* x[top.i]; //Giam tong

//Moi truong hop con deu da duoc xet

if (top.j > 1) { //j > 1 thi push sai mau, pop ra

s.pop();

continue;

} //top.j = -1 hoac top.j =1

x[top.i] = top.j;

sum += m[top.i] \* x[top.i];

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

++top.j;

}

cout << -1;

return 0;

} //Nguyen Huu Dung - 20215545

## **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

**Dữ liệu vào:**

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

**Kết quả:**

Ghi danh sách các lịch tìm được theo thứ tự từ điển

A screen shot of a computer program

Description automatically generated with low confidence

A picture containing text, screenshot, software, multimedia software

Description automatically generated

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer screen

Description automatically generated with medium confidence

// Nguyen Huu Dung - 20215545

#include <bits/stdc++.h>

using namespace std;

struct state{

int i, j, old\_L;

// i: Thu tu ngay trong lich

// j: Ngay do lam hay nghi

// old\_L : So ngay lam viec lien tiep

//# constructor

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

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

};

int main() {

int N, K1, K2;

cin >> N >> K1 >> K2;

int x[N+1];

stack<state> s;

int L = 0;

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

while (!s.empty()){

state &top = s.top();

if (top.i > N){ //Neu da co lich trinh du cac ngay

if (x[N] && top.old\_L >= K1 || !x[N]) { //Neu ngay cuoi di lam va so ngay di lam lien tiep >= k1

//Hoac ngay cuoi nghi

for (int i = 1; i <= N; ++i) //In ra lich trinh

cout << x[i];

cout << endl;

}

s.pop(); //Xoa ra khoi stack

continue;

}

s.pop();

x[top.i] = top.j;

if (top.j == 1) { //Neu ngay hien tai di lam

L = top.old\_L + 1; //So ngay di lam = old\_L + 1

if (L >= K1) { //Neu so ngay di lam lien tiep >= K1

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

} else {

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

}

} else {

L = 0;

if (top.old\_L + 1 <= K2) { //Neu so ngay di lam lien tiep khong vuot qua K2

s.push(state(top.i, 1, top.old\_L));

}

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

}

}

return 0;

}// Nguyen Huu Dung – 20215545s

## **Bài tập 3.10**. 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 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

**Dữ liệu vào:**

Dòng đầu chứa T là số testcase

T dòng tiếp theo, mỗi dòng mô tả một testcase, ghi N và H (1≤ H ≤ N ≤ 16)

**Kết quả:**

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

A picture containing text, screenshot, software, multimedia software

Description automatically generated

A screenshot of a computer program

Description automatically generated with medium confidence

A picture containing text, screenshot, software, multimedia software

Description automatically generated

//Nguyen Huu Dung - 20215545

#include <bits/stdc++.h>

using namespace std;

inline void print(const vector<int>& v) { //Ham in ra Vector

for (int i : v) {

cout << i;

}

cout << endl;

}

int main() {

int t;

cin >> t; //t la so test case

int n[t], h[t];

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

cin >> n[i] >> h[i];

}

for (int i=0; i<t; i++) { //Voi moi test case, ta co bo input n[i], h[i]

vector<int> v(n[i]); //tao ra mot vector chieu dai n

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

v[j] = j < h[i] ? 1 : 0; //Tao mot thu tu ban dau voi h chu so 1 va n-h chu so 0

}

vector<int> v\_end(n[i]);

v\_end = v; //Danh dau so lon nhat

do {

next\_permutation(v.begin(), v.end()); //tao cac hoan vi

print(v); //in ra

} while (v != v\_end); //Cho toi khi in ra chuoi lon nhat

cout << endl;

}

return 0;

} //Nguyen Huu Dung - 20215545

## **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.

**Dữ liệu vào:**

* Dòng đầu chứa n và r
* Tiếp theo là ma trận n×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.
* r dòng tiếp theo chứa danh sách các địa điểm mà người r 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 r dòng ghi chi phí đi lại ít nhất của r người theo thứ tự đầu vào

A screen shot of a computer

Description automatically generated with medium confidence

A picture containing text, screenshot, software, multimedia software

Description automatically generated

A picture containing text, screenshot, software

Description automatically generated

A screenshot of a computer program

Description automatically generated with medium confidence

//Nguyen Huu Dung - 20215545

#include <iostream>

#include <vector>

#include <algorithm>

#include <sstream>

#include <climits>

using namespace std;

const int INF = INT\_MAX;

//Ham hoan doi 2 phan tu

void swap(int& x, int& y) {

int temp = x;

x = y;

y = temp;

}

//Ham tinh chi phi moi lich trinh

void price(const vector<int>& a, int x, const vector<vector<int>>& graph, vector<int>& cost) {

//a la lich trinh, x chi nguoi thu x, graph la mang chi phi, cost chua chi phi nho nhat cua lich trinh

int sum = 0;

for (int i = 0; i < a.size() - 1; i++) {

if (graph[a[i]][a[i + 1]] == 0) { //Neu khong the di chuyen thi chi phi bang vo cung

sum = INF;

break;

}

sum += graph[a[i]][a[i + 1]]; //Nguoc lai cong don chi phi

}

if (cost[x] > sum) {

cost[x] = sum; //Ghi lai chi phi

}

}

//Thuat toan nguoi du lich

void permute(vector<int>& a, int i, int k, int x, const vector<vector<int>>& graph, vector<int>& cost) {

//a la lich trinh, i/k la vi tri dau tien/cuoi cung co the thay doi trong lich trinh

//x chi nguoi thu x, graph la mang chi phi, cost chua chi phi nho nhat

if (i == k) {

price(a, x, graph, cost);

}

else {

for (int j = i; j <= k; j++) { //Thuat toan TSP

swap(a[i], a[j]);

permute(a, i + 1, k, x, graph, cost);

swap(a[i], a[j]);

}

}

}

int main() {

cout << "Input: " << endl;

int n, r;

cin >> n >> r;

vector<vector<int>> graph(n + 1, vector<int>(n + 1)); //Mang chi phi

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

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

cin >> graph[i][j];

}

}

cin.ignore(); //Xoa bo nho dem nhap vao

vector<vector<int>> preferences(r); //Mang chua lich trinh cua tung nguoi

for (int i = 0; i < r; ++i) { //Nhap lich trinh tung nguoi

string line;

getline(cin, line);

stringstream ss(line);

while (!ss.eof()) {

int temp;

ss >> temp;

preferences[i].push\_back(temp);

}

}

vector<int> cost(r, INF); //Mang cost chua chi phi can tim cua tung nguoi

for (int i = 0; i < r; ++i) { //Thuc hien TSP cho tung lich trinh

permute(preferences[i], 1, preferences[i].size() - 2, i, graph, cost);

if (cost[i] == INF) {

//Neu khong the di chuyen duoc thi chi phi bang vo cung, nhung theo de bai in ra 0

cost[i] = 0;

}

}

cout << "Output:" << endl;

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

cout << cost[i] << endl;

}

return 0;

} //Nguyen Huu Dung - 20215545

## **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.

**Dữ liệu vào:**

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

Dòng 2: Chứa số nguyên m (1≤m≤60) là số cạnh của G m dòng tiếp theo: Mỗi dòng chứa hai số nguyên là một cạnh của G

**Kết quả:**

Số lượng đường đi đơn độ dài k

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

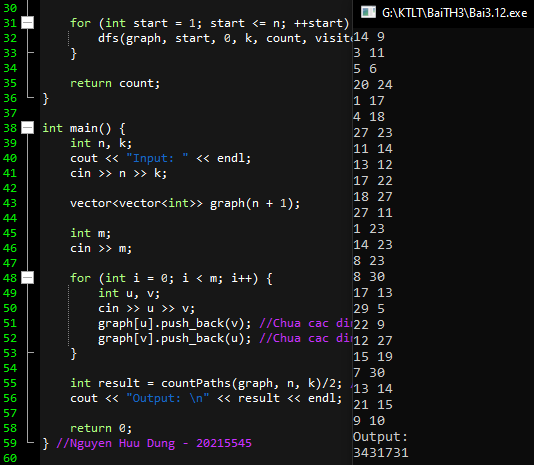
Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated with medium confidence



//Nguyen Huu Dung - 20215545

#include <iostream>

#include <vector>

using namespace std;

//Thuat toan duyet do thi theo chieu sau

void dfs(const vector<vector<int>>& graph, int current, int length, int k, int& count, vector<bool>& visited) {

//graph la ma tran ke, current la dinh bat dau, length la chieu dai duong di, count dem so canh, visited danh dau dinh da tham

visited[current] = true; //dinh hien tai da tham

if (length == k) { //Neu duong di da co k canh

count++; //Tang bien dem len

}

else if (length < k) { //Neu duong di co it hon k canh

for (int neighbor : graph[current]) { //Duyet cac dinh ke voi dinh hien tai

if (!visited[neighbor]) {

dfs(graph, neighbor, length + 1, k, count, visited);

}

}

}

visited[current] = false; //Sau khi duyet xong thi dua ve trang thai cu

} //Ham dfs se thay doi bien count tra ve so duong di chieu dai k bat dau tu dinh current

//Dem so duong di

int countPaths(const vector<vector<int>>& graph, int n, int k) {

int count = 0;

vector<bool> visited(n + 1, false);

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

dfs(graph, start, 0, k, count, visited); //Duyet dfs tung dinh mot

}

return count;

}

int main() {

int n, k;

cout << "Input: " << endl;

cin >> n >> k;

vector<vector<int>> graph(n + 1);

int m;

cin >> m;

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

int u, v;

cin >> u >> v;

graph[u].push\_back(v); //Chua cac dinh ke voi u

graph[v].push\_back(u); //Chua cac dinh ke voi v

}

int result = countPaths(graph, n, k)/2; //Do do thi la vo huong nen moi duong di da bi xet 2 lan

cout << "Output: \n" << result << endl;

return 0;

} //Nguyen Huu Dung - 20215545