**DANH MỤC**

[Phần 1. Đệ quy, quay lui 3](#_Toc57403805)

[Bài tập 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 3](#_Toc57403806)

[Bài tập 2: Quân mã đi tuần Trên bàn cờ vua kích thước n x n 2 có 1 quân mã đang ở ô (1,1). Hãy đưa ra 1 dãy các di chuyển của quân mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần( ô (1,1) coi như đã được đi qua) 5](#_Toc57403807)

[Phần 2. Kỹ thuật nhánh cận 8](#_Toc57403808)

[Bài tập 3: Bài toán người du lịch 1 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 đến thành phố j là ci,j. Hãy tìm tổng chi phí nhỏ nhất có thể 8](#_Toc57403809)

[Phần 3. Đệ quy có nhớ 11](#_Toc57403810)

[Bài tập 4: LIS Cho bài 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](#_Toc57403811)

[Phần 4. Khử đệ quy 14](#_Toc57403812)

[Bài tập 5: Tính C(k)n 14](#_Toc57403813)

[Bài tập 6: Tìm ước chung lớn nhất Tính ước chung lớn nhất của 2 số cho trước 17](#_Toc57403814)

[Bài tập 7: Liệt kê xâu nhị phân Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân có độ dài n mà không có k bit 1 nào liên tiếp. 18](#_Toc57403815)

[Bài tập 8: Cân đĩa Bạn đang muốn kiểm tra xem một vật cho trước có nặng đúng M như người ta nói hay không. Có 1 cân thang bằng n quả cân. Quả thứ i nặng mi. Hãy chỉ ra 1 cách cân thỏa mãn. Quy cách in ra được tích hợp trong mã nguồn. 21](#_Toc57403816)

[Phần 5. Bài tập về nhà 24](#_Toc57403817)

[Bài tập 9: Một y tá 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 2 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 1 dòng một sâu nhị phân độ dài n vói bit 0/1 tương ứng là nghỉ và làm việc. Các xâu phải được in ra theo thứ tự từ điển. 24](#_Toc57403818)

[Bài tập 10: Hàm khoảng cách Hamming Khoảng cách hamming giữa 2 xâu cùng độ dài là số vị trí mà kí tự tại đó trên 2 xâu là khác nhau. Cho S là xâu gồm n kí tự 0. Hãy liệt kê tất cả các xâu nhị phân có độ dài n, có khoảng cách hamming với S bằng H. Các xâu phải được liệt kê theo thứ tự từ điển. 28](#_Toc57403819)

[Bài tập 11: Lịch trình chụp ảnh Superior là một hòn đảo tuyệt vời với n địa điểm chụp ảnh và các đường 1 chiều nối các điểm chụp ảnh với nhau. Đoàn khác tham quan có n người với sở thích chụp ảnh khác nhau. Theo đó mỗi người sẽ đưa ra danh sách địa điểm mà họ muốn chụp. Bạn cần gíup mỗi người trong đoàn lập lịch di chuyển sao cho đi qua các điểm mà họ yêu cầu đúng 1 lần, không đi qua điểm nào khác, bắt đầu tại điểm đầu tiên và kết thúc tại điểm cuối cùng trong danh sách và tổng khoảng cách đi lại là nhỏ nhất 31](#_Toc57403820)

[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 1 lần 36](#_Toc57403821)

DANH MỤC HÌNH ẢNH

[Hình 1 dãy lucas 4](#_Toc57403988)

[Hình 2 Bài 2 : Quân mã đi tuần 7](#_Toc57403989)

[Hình 3 Bài 3 : Bài toán người du lịch 11](#_Toc57403990)

[Hình 4 Bài 4 : LIS 14](#_Toc57403991)

[Hình 5 Bài 5 : Tính C(k)n 16](#_Toc57403992)

[Hình 6 Bài 6 : Tìm ước chung lớn nhất 18](#_Toc57403993)

[Hình 7 Bài 7 : Liệt kê xâu nhị phân 21](#_Toc57403994)

[Hình 8 Bài 8 cân đĩa 24](#_Toc57403995)

[Hình 9 Bài 9 liệt kê xâu nhị phân 27](#_Toc57403996)

[Hình 10 Bài 10 : Hàm khoảng cách hamming 30](#_Toc57403997)

[Hình 11 Bài 11 : Lịch trình chụp ảnh 35](#_Toc57403998)

[Hình 12 Bài 12 : Đếm đường đi trên đồ thị 39](#_Toc57403999)

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

Phần 1. Đệ quy, quay lui

Bài tập 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

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_1.cpp**

#include<stdio.h>

#include<iostream>

using namespace std;

int lucas(int n) {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ho va ten : Nguyen Quang Huy

MSSV : 20183554

\*/

if(n == 0) return 2;

if(n == 1) return 1;

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

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

int main(){

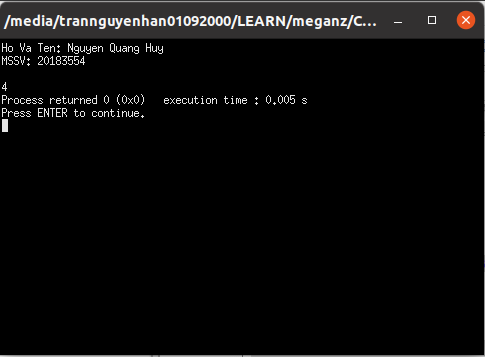
printf("Ho Va Ten: Nguyen Quang Huy\n");

printf("MSSV: 20183554\n\n");

cout << lucas(3);

}

Kết quả:



Hình 1 dãy lucas

Bài tập 2: Quân mã đi tuần  
Trên bàn cờ vua kích thước n x n 2 có 1 quân mã đang ở ô (1,1). Hãy đưa ra 1 dãy các di chuyển của quân mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần( ô (1,1) coi như đã được đi qua)

### Tên file: 20183554-NguyenQuangHuy\_Bai 3\_2.cpp

### #include<stdio.h>

### #include<stdlib.h>

### #include <iostream>

### using namespace std;

### int n;

### int X[100], Y[100]; //# Lưu tọa độ các bước di chuyển của quân mã

### int mark[100][100]; //# Đánh dấu vị trí các ô mà quân mã đã di chuyển qua

### //# Mảng hx, hy mô tả 8 vị trí quân mã có thể di chuyển kể từ vị trí hiện tại

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

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

### //# In ra dãy các di chuyển tìm được

### void print\_sol(){

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

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

### exit(0);

### }

### bool check(int xx, int yy){

### if(mark[xx][yy] == 1) return false;

### if(xx < 1 || xx > n || yy < 1 || yy > n) return false;

### //cout << xx << " " << yy << endl;

### return true;

### }

### //# Thuật 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];

### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Ho va ten : Nguyen Quang Huy

### MSSV : 20183554

### \*/

### if(check(xx,yy)){

### mark[xx][yy] = 1;

### X[k] = xx;

### Y[k] = yy;

### 

### if(k == n\*n) print\_sol();

### else TRY(k+1);

### mark[xx][yy] = 0;

### }

### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

### }

### }

### int main(){

### printf("Ho Va Ten: Nguyen Quang Huy\n");

### printf("MSSV: 20183554\n\n");

### cin >> n;

### mark[1][1] = 1;

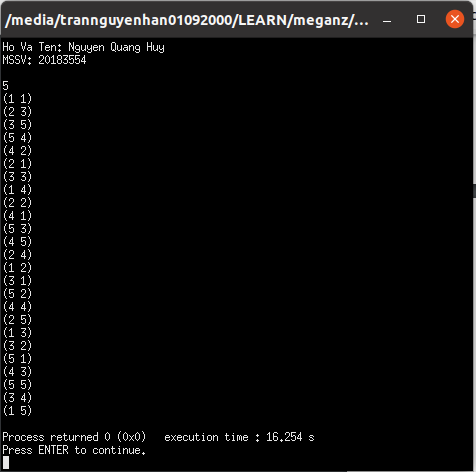
### X[1] = Y[1] = 1;

### TRY(2);

### return 0;

### }

### Kết quả:



Hình 2 Bài 2 : Quân mã đi tuần

Phần 2. Kỹ thuật nhánh cận

Bài tập 3: Bài toán người du lịch  
1 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 đến thành phố j là ci,j. Hãy tìm tổng chi phí nhỏ nhất có thể

### Tên file: 20183554-NguyenQuangHuy\_Bai 3\_3.cpp

### #include <bits/stdc++.h>

### using namespace std;

### #define MAX 100

### 

### int n, c[MAX][MAX]; //# số thành phố và ma trận chi phí

### int cmin = INT\_MAX; //# chi phí đi lại nhỏ nhất giữa hai thành phố khác nhau

### int best = INT\_MAX; //# tổng chi phí nhỏ nhất cần tìm, ban đầu đặt bằng giá trị vô cùng lớn INT\_MAX = 2^31-1

### int curr; //# tổng chi phí tới thời điểm hiện tại

### int mark[MAX]; //# đánh dấu những thành phố đã đi

### int x[MAX]; //# lưu giữ các thành phố đã đi

### 

### //# Đọc dữ liệu 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);

### }

### 

### //# Thuật toán quay lui

### void TRY(int k){

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

### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Ho va ten : Nguyen Quang Huy

### MSSV : 20183554

### \*/

### 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];

### }

### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

### }

### }

### 

### int main() {

### printf("Ho Va Ten: Nguyen Quang Huy\n");

### printf("MSSV: 20183554\n\n");

### input();

### x[1] = 1;

### TRY(2);

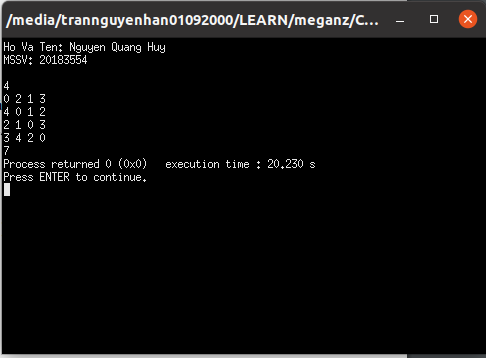
### cout << best;

### return 0;

### }

### 

### Kết quả:



Hình 3 Bài 3 : Bài toán người du lịch

Phần 3. Đệ quy có nhớ

Bài tập 4: LIS  
Cho bài 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.

### Tên file: 20183554-NguyenQuangHuy\_Bai 3\_4.cpp

#include <bits/stdc++.h>

using namespace std;

int a[1000], n;

int mem[1000]; //# mảng ghi nhớ lời giải các bài toán con đã được giải

void init(){

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

}

//# Quy hoạch động,

//# Hàm lis(i) trả về độ dài dãy con tăng dài nhất kết thúc bởi a[i]

void incsubseq(){

mem[0] = 1;

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

mem[i] = 1;

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

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

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

mem[i] = mem[j] + 1;

}

}

}

}

}

int lis(int i) {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ho va ten : Nguyen Quang Huy

MSSV : 20183554

\*/

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] << " ";

}

int main(){

printf("Ho Va Ten: Nguyen Quang Huy\n");

printf("MSSV: 20183554\n\n");

init();

cin >> n;

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

incsubseq();

int res = 1, pos = 0;

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

if (res < lis(i)){

res = lis(i);

pos = i;

}

}

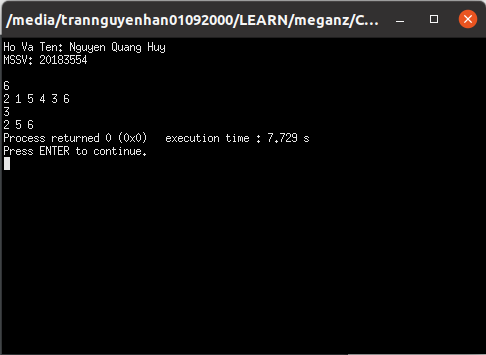
cout << res << endl;

trace(pos);

return 0;

}

Kết quả :



Hình 4 Bài 4 : LIS

Phần 4. Khử đệ quy

Bài tập 5: Tính C(k)n

### Tên file: 20183554-NguyenQuangHuy\_Bai 3\_5.cpp

#include<stdio.h>

#include <iostream>

using namespace std;

const int MAX = 1000;

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){

//# Khử đệ quy

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ho va ten : Nguyen Quang Huy

MSSV : 20183554

\*/

long result = 1; //(nC0 = 1)

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

{

result = result \* n / i;

n--;

}

return result;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

int main() {

printf("Ho Va Ten: Nguyen Quang Huy\n");

printf("MSSV: 20183554\n\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 n = 1; n <= m; ++n){

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

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

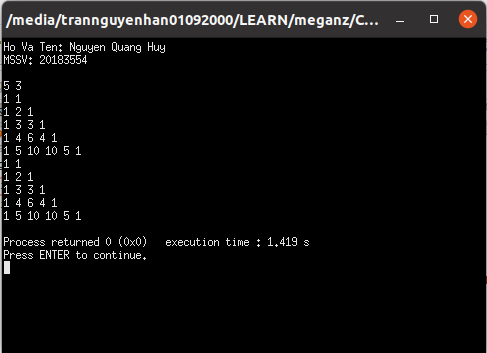
printf("\n");

}

return 0;

}

Kết quả:



Hình 5 Bài 5 : Tính C(k)n

## 

Bài tập 6: Tìm ước chung lớn nhất  
Tính ước chung lớn nhất của 2 số cho trước

### Tên file: 20183554-NguyenQuangHuy\_Bai 3\_6.cpp

#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){

//# Khử đệ quy

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ho va ten : Nguyen Quang Huy

MSSV : 20183554

\*/

while(b != 0){

int tmp = a % b;

a = b;

b = tmp;

}

return a;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

int main() {

printf("Ho Va Ten: Nguyen Quang Huy\n");

printf("MSSV: 20183554\n\n");

int a, b;

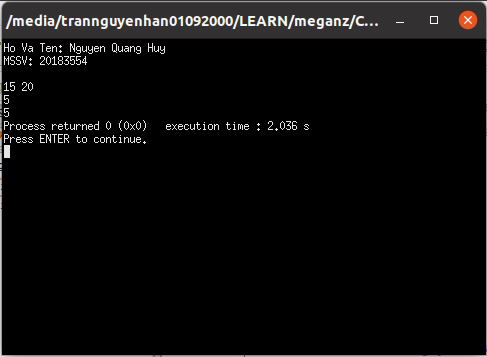
cin >> a >> b;

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

return 0;

}

Đáp án:



Hình 6 Bài 6 : Tìm ước chung lớn nhất

Bài tập 7: Liệt kê xâu nhị phân  
Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân có độ dài n mà không có k bit 1 nào liên tiếp.

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_7.cpp**

#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 Quang Huy\n");

printf("MSSV: 20183554\n\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;

}

//# Khử đệ quy

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ho va ten : Nguyen Quang Huy

MSSV : 20183554

\*/

if(top.j>0) 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 = L + 1;

} else {L = 0;}

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

}

top.j++;

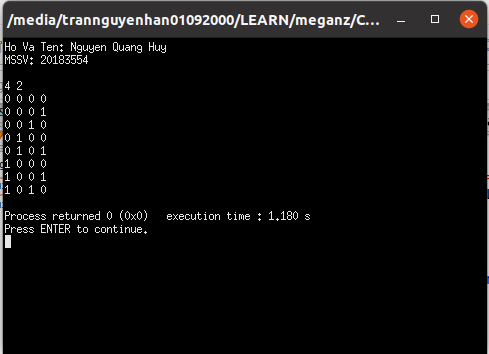
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

return 0;

}

Đáp án:



Hình 7 Bài 7 : Liệt kê xâu nhị phân

## 

Bài tập 8: Cân đĩa  
Bạn đang muốn kiểm tra xem một vật cho trước có nặng đúng M như người ta nói hay không. Có 1 cân thang bằng n quả cân. Quả thứ i nặng mi. Hãy chỉ ra 1 cách cân thỏa mãn. Quy cách in ra được tích hợp trong mã nguồn.

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_8.cpp**

#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() {

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;

}

//# 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]));

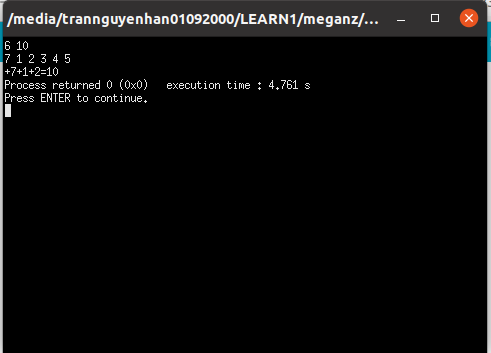
}

cout << -1;

return 0;

}

Kết quả:



Hình 8 Bài 8 cân đĩa

Phần 5. Bài tập về nhà

Bài tập 9: Một y tá 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 2 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 1 dòng một sâu nhị phân độ dài n vói bit 0/1 tương ứng là nghỉ và làm việc. Các xâu phải được in ra theo thứ tự từ điển.

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_9.cpp**

#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;

void inputData(){

cin >> n >> k1 >> k2;

}

bool check(int a, int i){

if(a==1) return true;

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;

}

}

return true;

}

}

void solution(){/\*

int so1 = 0;

for(int i=n; i>0; i--)

if(x[i] == 1) so1++;

else break;

if(so1!=0 && so1<k1) return; \*/

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

cout << x[i];

cout << endl;

cnt++;

//if(cnt == du) cnt = 0;

}

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;

}

}

}

int main(){

inputData();

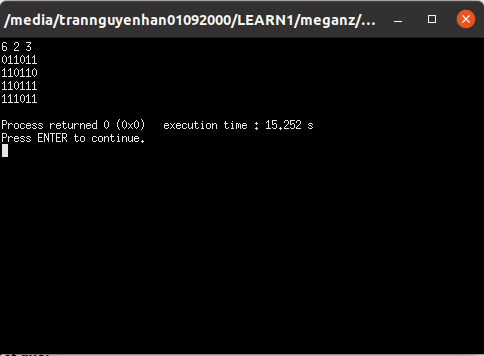
TRY(1);

//cout << cnt % du;

return 0;

}

Kết quả:



Hình 9 Bài 9 liệt kê xâu nhị phân

Bài tập 10: Hàm khoảng cách Hamming  
Khoảng cách hamming giữa 2 xâu cùng độ dài là số vị trí mà kí tự tại đó trên 2 xâu là khác nhau. Cho S là xâu gồm n kí tự 0. Hãy liệt kê tất cả các xâu nhị phân có độ dài n, có khoảng cách hamming với S bằng H. Các xâu phải được liệt kê theo thứ tự từ điển.

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_10.cpp**

#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;

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;

}

// Kiem tra 2 co khoang cach Hamming la bnh

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;

}

bool check(int a, int i){

return true;

}

void solution(){

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

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

cout << x[i];

cout << endl;

}

}

void TRY(int a){

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

x[a] = i;

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

else

TRY(a+1);

}

}

int main(){

int T;

cin >> T;

while(T > 0){

input();

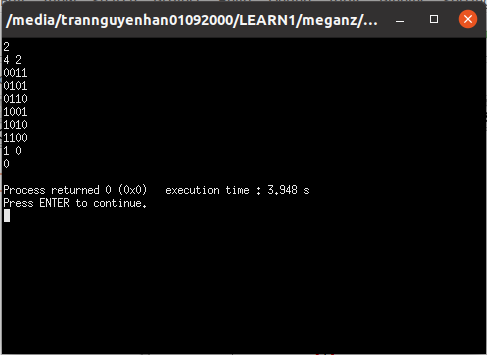
TRY(0);

T--;

}

}

Kết quả:



Hình 10 Bài 10 : Hàm khoảng cách hamming

Bài tập 11: Lịch trình chụp ảnh  
Superior là một hòn đảo tuyệt vời với n địa điểm chụp ảnh và các đường 1 chiều nối các điểm chụp ảnh với nhau. Đoàn khác tham quan có n người với sở thích chụp ảnh khác nhau. Theo đó mỗi người sẽ đưa ra danh sách địa điểm mà họ muốn chụp. Bạn cần gíup mỗi người trong đoàn lập lịch di chuyển sao cho đi qua các điểm mà họ yêu cầu đúng 1 lần, không đi qua điểm nào khác, bắt đầu tại điểm đầu tiên và kết thúc tại điểm cuối cùng trong danh sách và tổng khoảng cách đi lại là nhỏ nhất

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_11.cpp**

#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;

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;

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

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);

// Tach str thanh cac so va ghi vao vector vt

/\*

int pre = 0;

for(int i=0; i<str.length(); i++){

if(str[i] == ' '){

string tmp = str.substr(pre,i);

pre = i + 1;

stringstream convert(tmp);

int tmp\_int = 0;

convert >> tmp\_int;

vt.push\_back(tmp\_int - 1);

}

}

string tmp = str.substr(pre,str.length());

stringstream convert(tmp);

int tmp\_int = 0;

convert >> tmp\_int;

vt.push\_back(tmp\_int - 1);

\*/

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

}

}

// 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);

// 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

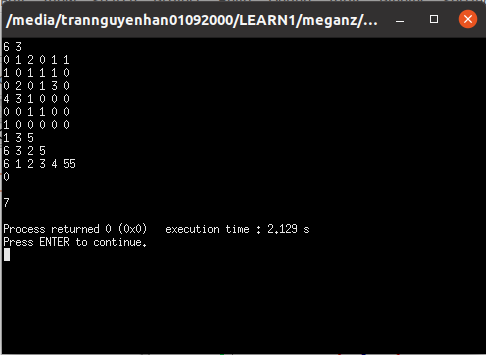
vt.erase(vt.begin(), vt.end());

r--;

}

}

Kết quả :



Hình 11 Bài 11 : Lịch trình chụp ảnh

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 1 lần

**Tên file: 20183554-NguyenQuangHuy\_Bai 3\_12.cpp**

#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;

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;

}

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(){

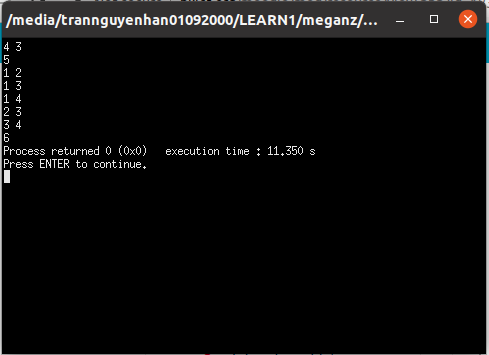
input();

TRY(0);

cout << res / 2;

}

Kết quả :



Hình 12 Bài 12 : Đếm đường đi trên đồ thị