Contents

[Danh mục hình ảnh 2](#__RefHeading___Toc349_2734074527)

[Bài thực hành số 3: Đệ quy và khử đệ quy để giải quyết một số bài toán 3](#__RefHeading___Toc1901_2139985059)

[Phần 1. Thực hành về đệ quy 3](#__RefHeading___Toc353_2734074527)

[1. Bài tập 1: Tính dãy Lucas 3](#__RefHeading___Toc355_2734074527)

[2. Bài tập 2: Quân mã đi tuần 4](#__RefHeading___Toc357_2734074527)

[3. Bài tập 3: Bài toán người du lịch 5](#__RefHeading___Toc359_2734074527)

[4. Bài tập 4: LIS 7](#__RefHeading___Toc2013_1126995683)

[Phần 2. Thực hành về đệ quy 9](#__RefHeading___Toc1881_1126995683)

[1. Bài tập 5: Tính tổ hợp 9](#__RefHeading___Toc365_2734074527)

[2. Bài tập 6: Tính ước chung lớn nhất 11](#__RefHeading___Toc367_2734074527)

[3. Bài tập 7: Liệt kê xâu nhị phân 12](#__RefHeading___Toc2015_1126995683)

[4. Bài tập 8: Cân đĩa 14](#__RefHeading___Toc2017_1126995683)

[Phần 3. Bài tập về nhà 17](#__RefHeading___Toc375_2734074527)

[1. Bài tập 9: Lập lịch cho y tá 17](#__RefHeading___Toc377_2734074527)

[2. Bài tập 10: Khoảng cách hamming 17](#__RefHeading___Toc379_2734074527)

[3. Bài tập 11: Lịch trình chụp ảnh 18](#__RefHeading___Toc2019_1126995683)

[4. Bài tập 12: Đếm đường đi 19](#__RefHeading___Toc2021_1126995683)

# Danh mục hình ảnh

[Hình 1 Bài 1.1 Tính dãy Lucas 3](#Hình!0|sequence)

[Hình 2 Bài 1.2 Quân mã đi tuần 5](#Hình!1|sequence)

[Hình 3 Bài 1.3 Bài toán người du lịch 7](#Hình!2|sequence)

[Hình 4 Bài 1.4 LIS 9](#Hình!3|sequence)

[Hình 5 Bài 1.5 Tính tổ hợp 11](#Hình!4|sequence)

[Hình 6 Bài 1.6 Tính ước chung lớn nhất 12](#Hình!5|sequence)

[Hình 7 Bài 1.7 Liệt kê xâu nhị phân 14](#Hình!6|sequence)

[Hình 8 Bài 1.8 Cân đĩa 16](#Hình!7|sequence)

[Hình 9 Bài 1.9 Lập lịch cho y tá 17](#Hình!8|sequence)

[Hình 10 Bài 1.10 Khoảng cách hamming 18](#Hình!9|sequence)

[Hình 11 Bài 1.11 Lịch trình chụp ảnh 18](#Hình!10|sequence)

[Hình 12 Bài 1.12 Đếm đường đi 19](#Hình!11|sequence)

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

## Phần 1.**Thực hành về đệ quy**

### Bài tập 1: **Tính dãy Lucas**

*Dãy Lucas được định nghĩa bởi* *L**n**=**L**n**−**1**+**L**n**−**2 và bắt đầu bởi* *L**0**=**2,* *L**1**=**1. Viết hàm tính số Lucas thứ* *n.*

#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);

}

int main(){

int n;

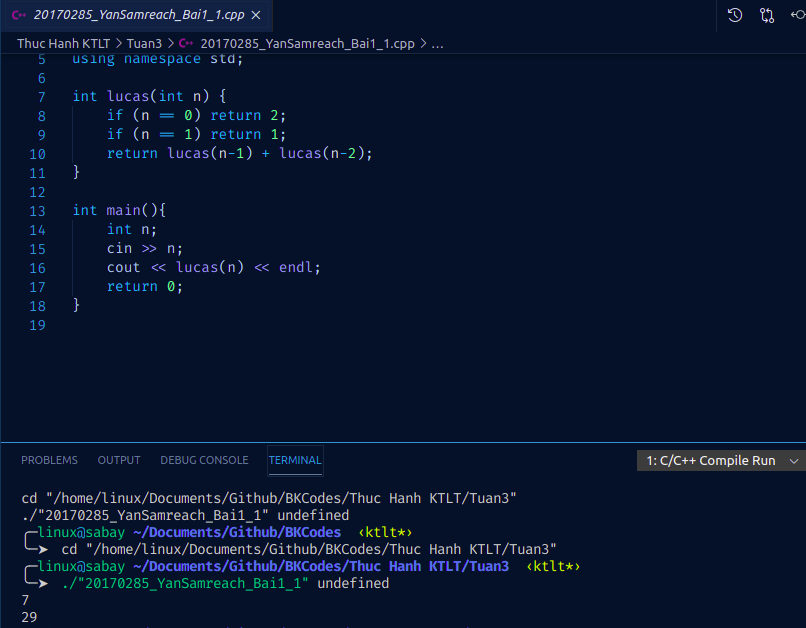
cin >> n;

cout << lucas(n) << endl;

return 0;

}

Kết quả:



Hình 1 Bài 1.1 Tính dãy Lucas

### Bài tập 2: **Quân mã đi tuần**

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)

#include <iostream>

using namespace std;

int n;

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

int mark[100][100]; //# Đánh dấu vị trí các ô mà quân mã đi 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]);

}

int isOk(int n){

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

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

if(!mark[i][j]) return 0;

return 1;

}

//# 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];

if(!mark[xx][yy] && xx > 0 && yy > 0 && xx <= n && yy <= n){

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(){

cin >> n;

mark[1][1] = 1;

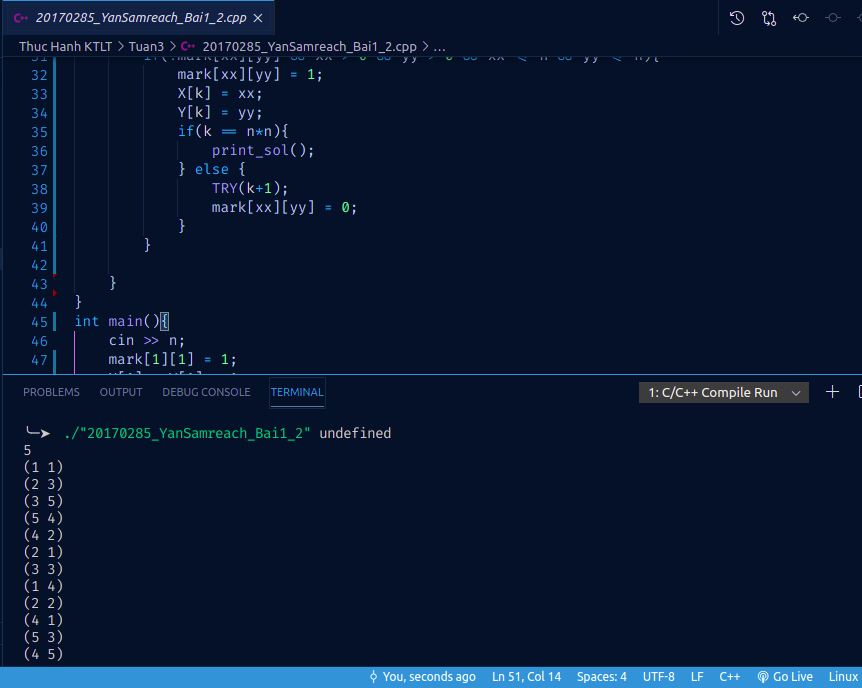
X[1] = Y[1] = 1;

TRY(2);

return 0;

}

Kết quả:



Hình 2 Bài 1.2 Quân mã đi tuần

### 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. Chi phí để đi từ thành phố *i* sang thành phố jlà ci,j. Hãy tìm tổng chi phí nhỏ nhất có thể

#include <bits/stdc++.h>

#define MAX 1000

using namespace std;

int n;

int a[MAX];

int c[MAX][MAX];

bool visitted[MAX];

int c\_min = MAX\*MAX;

int f = 0;

int f\_min = MAX\*MAX;

void input() {

cin >> n;

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

visitted[i] = false;

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

cin >> c[i][j];

if(c[i][j] != 0) c\_min = min(c\_min, c[i][j]);

}

}

}

void Try(int k){

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

if(!visitted[i]){

a[k] = i;

visitted[i] = true;

f += c[a[k-1]][a[k]];

if(k == n){

f\_min = min(f + c[a[n]][a[1]], f\_min);

} else{

int g = f + (n-k+1)\*c\_min;

if(g < f\_min) Try(k+1);

}

visitted[i] = false;

f -= c[a[k-1]][a[k]];

}

}

}

int main(){

input();

a[1] = 1;

visitted[a[1]] = true;

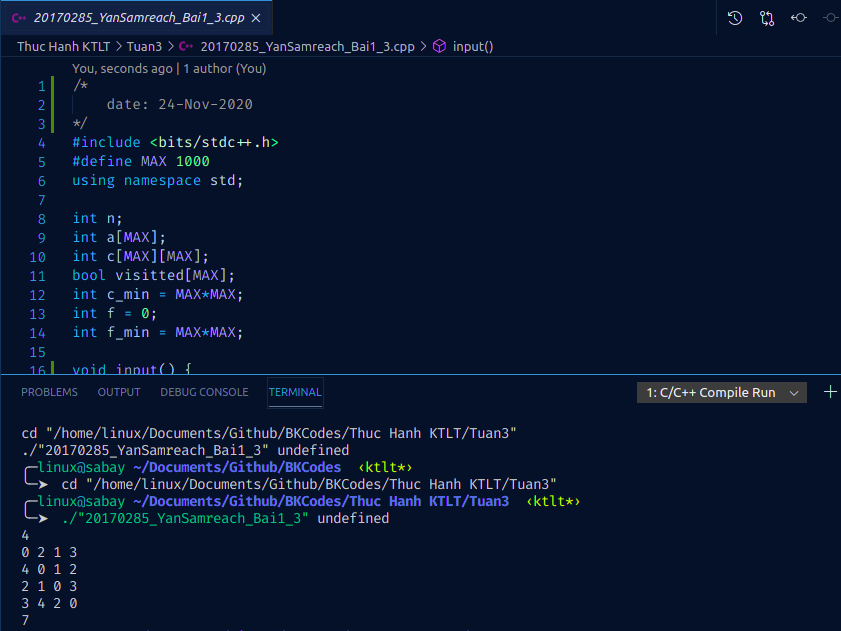
Try(2);

cout << f\_min << "\n";

return 0;

}

Kết quả:



Hình 3 Bài 1.3 Bài toán người du lịch

### **Bài tập 4: LIS**

*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*

#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

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 hoach dong,

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

int lis(int i) {

if(i == 1) {

if(a[1] > a[0]){

mem[0] = 1;

mem[1] = 2;

return 2;

}

mem[0] = 1;

mem[1] = 1;

return 1;

} else {

for(int j = i-1; j >= 0; j--){

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

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

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

return mem[i];

} return mem[j];

}

}

}

return 1;

}

//# 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(){

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;

}

}

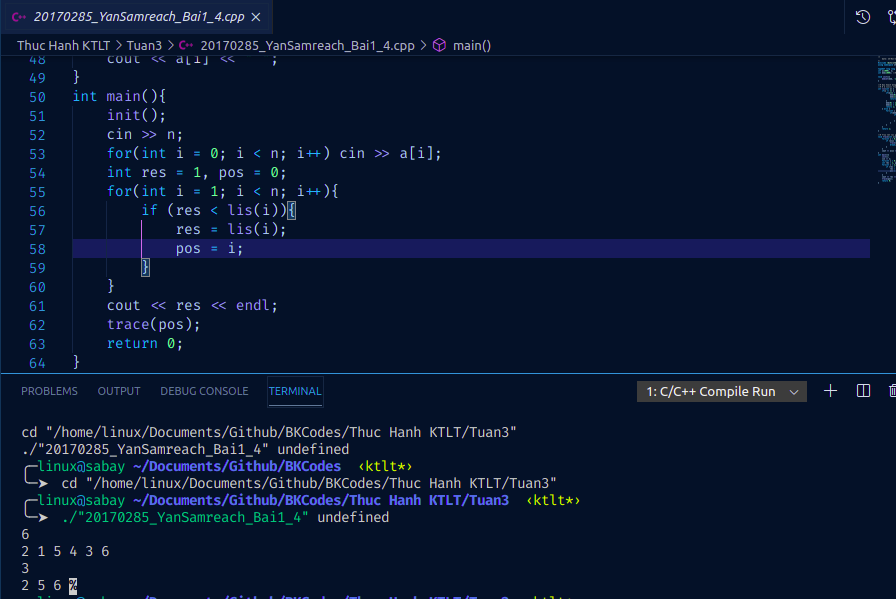
cout << res << endl;

trace(pos);

return 0;

}

Kết quả:



Hình 4 Bài 1.4 LIS

## **Phần 2.** **Thực hành về đệ quy**

### Bài tập 5: **Tính tổ hợp**

*Tính của C (k, n)*

#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){

long res = 1;

for ( int i = 1; i <= k ; i ++ , n-- )

res = res \* n/i;

return res;

}

int main() {

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));

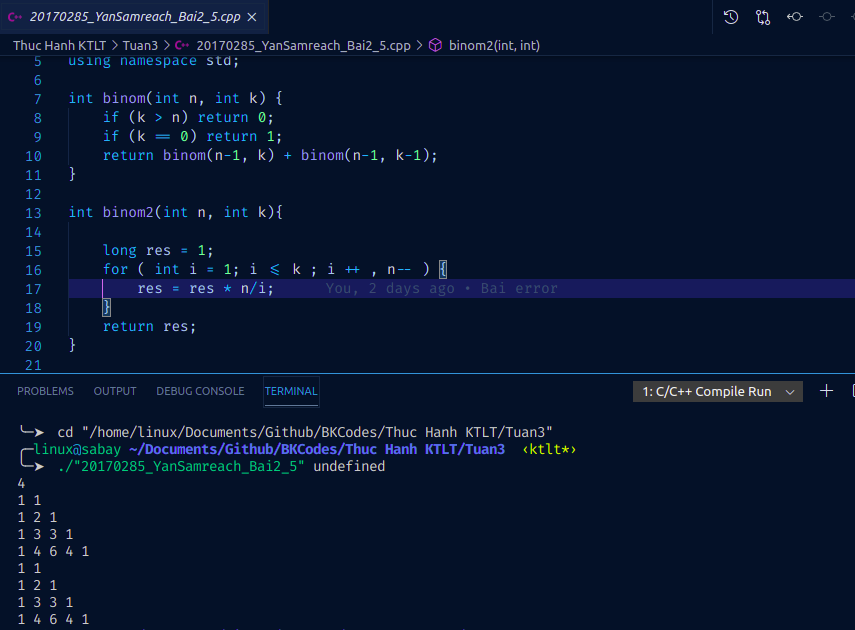
printf("\n");

}

return 0;

}

Kết quả:



Hình 5 Bài 1.5 Tính tổ hợp

### Bài tập 6: **Tính ước chung lớn nhất**

*Tính ước chung lớn nhất của hai số 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){

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

return a + b;

}

while (a != b){

if (a > b){

a -= b; // a = a - b

}else{

b -= a;

}

}

return a;

}

int main() {

int a, b;

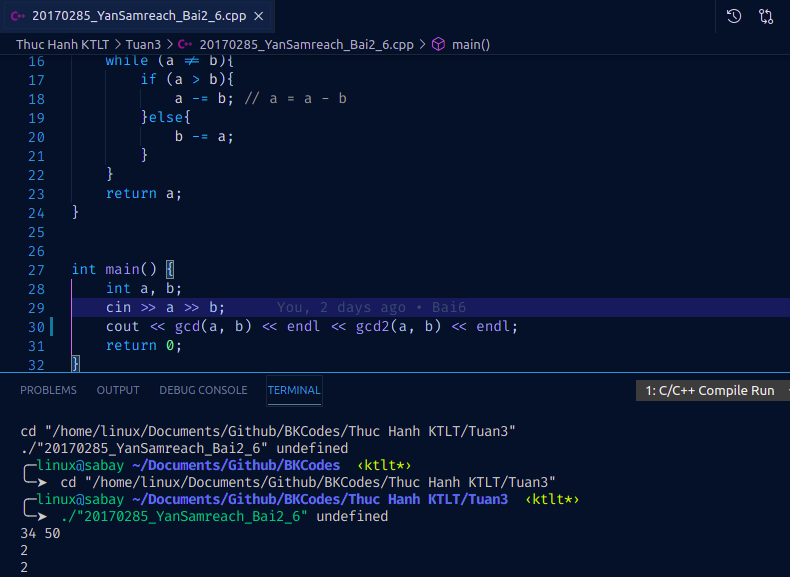
cin >> a >> b;

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

return 0;

}

Kết quả:



Hình 6 Bài 1.6 Tính 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 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() {

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;

}

//# Khu de quy

if(top.j){

L = top.old\_L;

}

// Neu bit thu i > 1 thi bo qua

if(top.j > 1){

s.pop();

continue;

}

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

// gan gia tri bit i cho x[i]

x[top.i] = top.j;

top.old\_L = L;

L = top.j ? L + 1 : 0;

// bit i+1 co gia tri 0

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

}

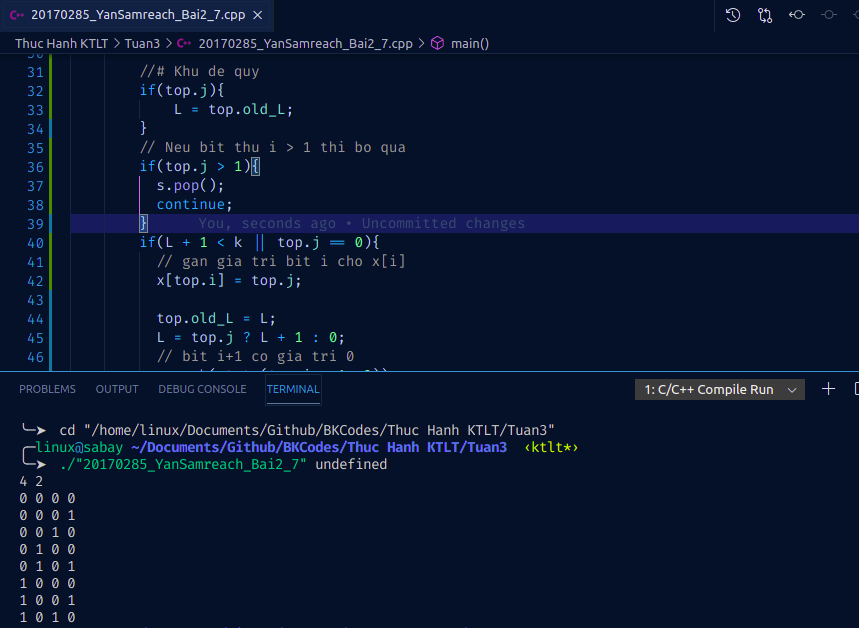
++top.j;

}

return 0;

}

Kết quả:



Hình 7 Bài 1.7 Liệt kê xâu nhị phân

### **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* *m**i. 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;

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

};

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));

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

if(top.j > -1){

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

}

if(top.j > 1){

s.pop();

continue;

}

x[top.i] = top.j;

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

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

top.j++;

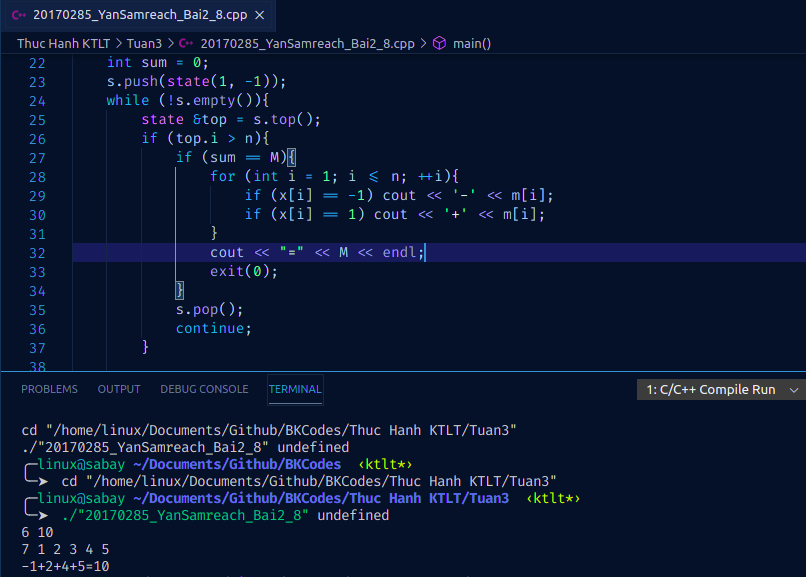
}

cout << -1;

return 0;

}

Kết quả:



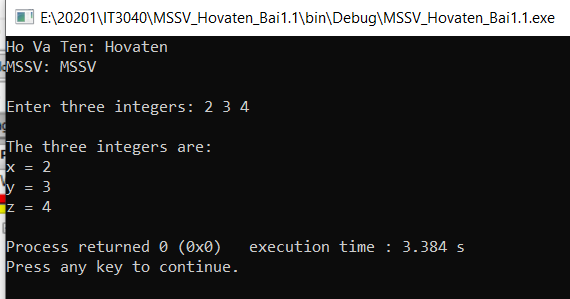
Hình 8 Bài 1.8 Cân đĩa

## Phần 3. **Bài tập về nhà**

### 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* *[**K**1**,**K**2**]. 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*

Kết quả:

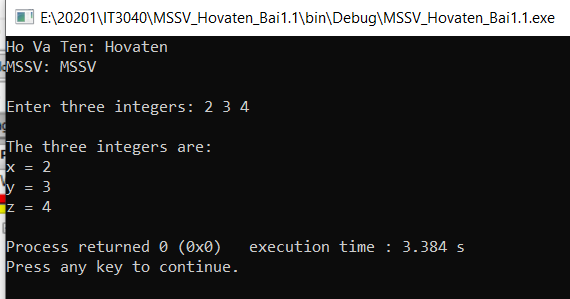


Hình 9 Bài 1.9 Lập lịch cho y tá

### 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* *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*

Kết quả:

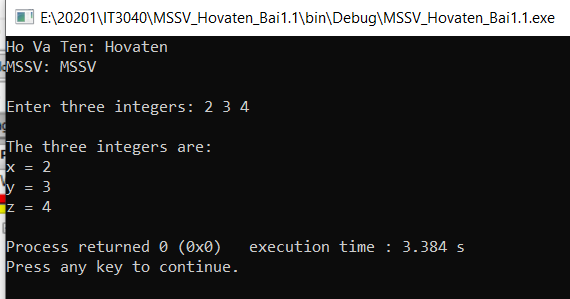


Hình 10 Bài 1.10 Khoảng cách hamming

### **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.

Kết quả:

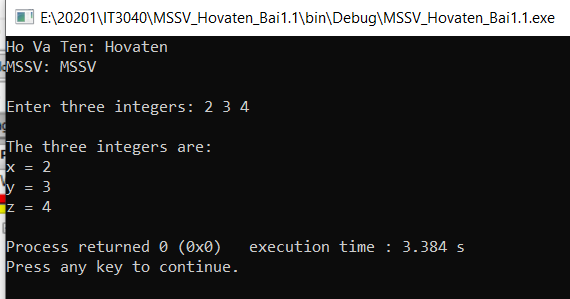


Hình 11 Bài 1.11 Lịch trình chụp ảnh

### **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 một lần.*

Kết quả:



Hình 12 Bài 1.12 Đếm đường đi