**I.Câu 12**

**1. Tên đề bài**

Tính diện tích của một hình thang cân biết độ dài hai cạnh đáy là a và b và góc nhọn thuộc đáy lớn là α. Viết chương trình sử dụng hàm để giải bài toán này.

**2. Thuật toán**

Bước 1: Nhập ba số nguyên a, b, và alpha

Bước 2: Đổi độ sang radian

Bước 3: Tính đường cao

Bước 4: Tính diện tích theo công thức (a+b)/2\*h

Bước 5: In ra kết quả, lấy 2 chữ số sau dấu phẩy

Bước 6: Kết thúc chương trình.

**3. Chương trình nguồn:**

#include <stdio.h>

#include <math.h>

double dt\_ht(double a, double b, double alpha) {

double alphaRad = alpha \* M\_PI / 180.0;

double h = abs(a - b) / 2 \* tan(alphaRad);

double S = ((a + b) / 2) \* h;

return S;

}

int main() {

double a, b, alpha;

scanf("%lf", &a);

scanf("%lf", &b);

scanf("%lf", &alpha);

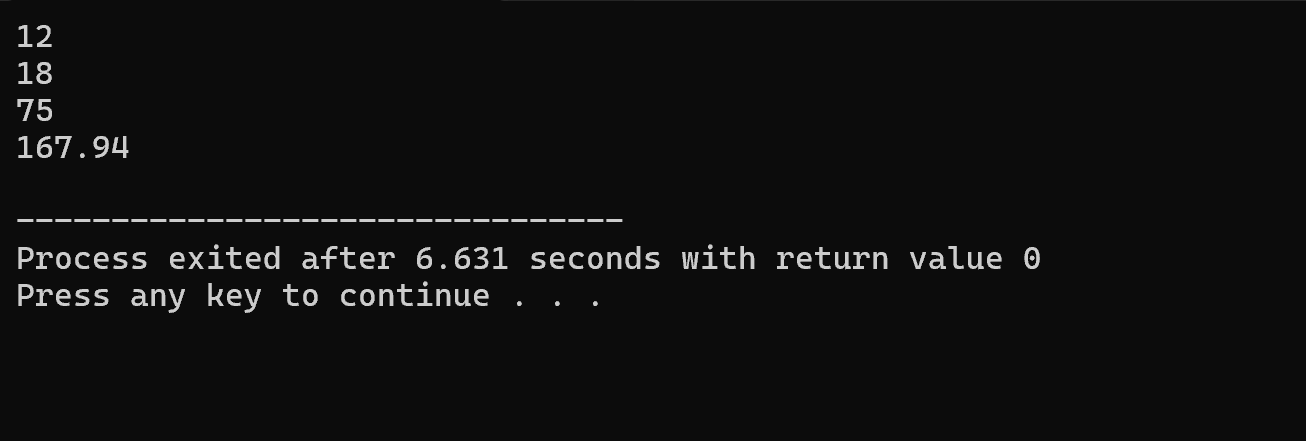
double dt = dt\_ht(a, b, alpha);

printf("%.2lf\n", dt);

return 0;

}

**4. Kết quả chương trình**



**II. Bài 27**

**1. Tên đề bài**

Giả sử n là số tự nhiên và n!! được định ngĩa như sau:  
N!!=1.3.5…n với n lẻ, N!!=2.4.6…n với n chẵn  
Với n (n<=2000) . Hãy viết chương trình sử dụng hàm để tính:  
a. n!!  
b. (-1)n+1n!!

**2. Thuật toán**

Bước 1: Khởi tạo và kiểm tra điều kiện đầu vào, n< 2000

Bước 2: Tính n!!

Bước 3: Tính (-1)^n+1 \* n!!**:**

Bước 4: In kết quả

**3. Chương trình nguồn**

#include <stdio.h>

#include <math.h>

// Hàm tính n!!

long long tinh\_n\_fact(int n) {

long long ket\_qua = 1;

if (n % 2 == 0) {

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

ket\_qua \*= i;

}

} else {

for (int i = 1; i <= n; i += 2) {

ket\_qua \*= i;

}

}

return ket\_qua;

}

// Hàm tính (-1)^n+1 \* n!!

long long tinh\_bieu\_thuc(int n) {

long long n\_fact = tinh\_n\_fact(n);

int he\_so = pow(-1, n + 1);

return he\_so \* n\_fact;

}

int main() {

int n;

printf(" nhap n (n <= 2000): ");

scanf("%d", &n);

if (n > 2000) {

printf("Gia tri cua n < 2000!\n");

} else {

// Tính n!!

long long n\_fact = tinh\_n\_fact(n);

printf(" Gia tri cua %d!! la %lld\n", n, n\_fact);

// Tính (-1)^n+1 \* n!!

long long ket\_qua\_bieu\_thuc = tinh\_bieu\_thuc(n);

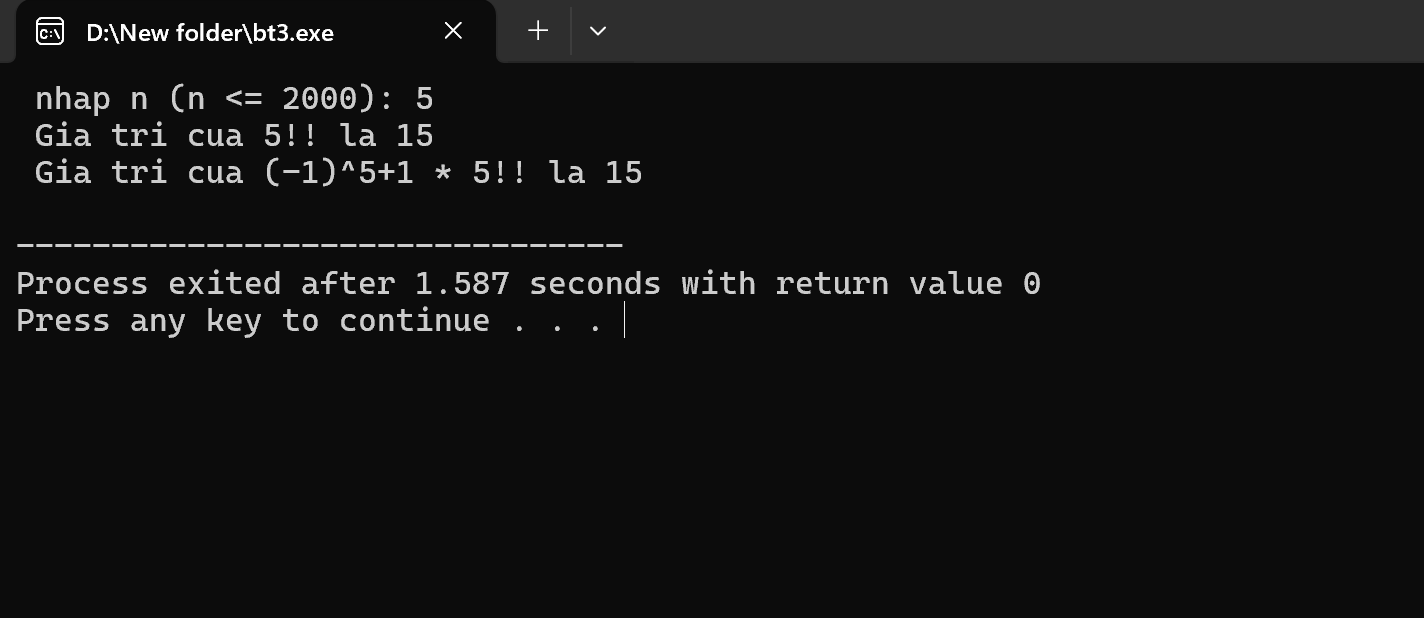
printf(" Gia tri cua (-1)^%d+1 \* %d!! la %lld\n", n, n, ket\_qua\_bieu\_thuc);

}

return 0;

}

**4. Kết quả chương trình**

****

**III. Bài 47**

**1. Tên đề bài**

Viết chương trình tính *sin(x)* triển khai theo chuỗi sau:  
S=sin(x) = x/1! – x3/3! + ... + (-1)n.x2n+1/(2n+1)!  
số phần tử được chọn cho tới khi đạt độ chính xác: |x2n+1/(2n+1)!| < EPS=10-6

**2. Thuật toán**

Bước 1: Nhập x đơn vị độ

Bước 2: Tạo hàm đổi độ sang radian

Bước 3: Tạo hàm tính giai thừa

Bước 4: Tính giá trị sin(x) theo chuỗi Taylor với độ chính xác 10^(-6)

Bước 5: In ra kết quả

**3. Chương trình nguồn**

#include <stdio.h>

#include <math.h>

double giai\_thua(int n) {

double kq = 1;

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

kq \*= i;

}

return kq;

}

double do\_sang\_radian(double x\_do) {

return x\_do \* M\_PI / 180.0;

}

double tinh\_sin(double x) {

double EPS = 1e-6;

double term = x;

double sin\_x = term;

int n = 1;

while (fabs(term) >= EPS) {

term = -term \* x \* x / ((2 \* n) \* (2 \* n + 1));

sin\_x += term;

n++;

}

return sin\_x;

}

int main() {

double x\_do;

printf(" Nhap x (do): ");

scanf("%lf", &x\_do);

double x\_radian = do\_sang\_radian(x\_do);

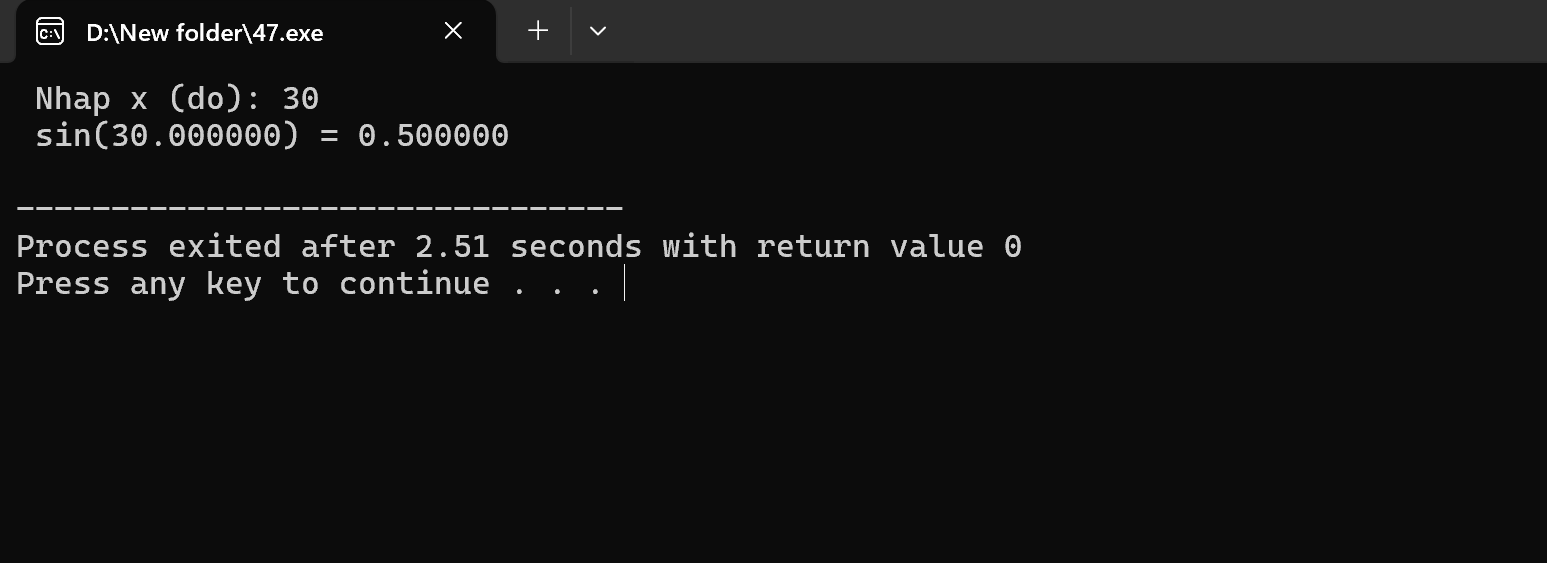
double ket\_qua = tinh\_sin(x\_radian);

printf(" sin(%.6f) = %.6f\n", x\_do, ket\_qua);

return 0;

}

**4. Kết quả chương trình**



**IV. Bài 59**

**1. Tên đề bài**

Viết chương trình nhập mảng 1 chiều A nguyên có n phần tử có sử dụng hàm  
(với n nhập từ bàn phím). :  
a. Xuất các phần tử đã nhập ra màn hình.  
b. Tìm ước số chung lớn nhất của các số trên  
c. Tìm bội số chung nhỏ nhất của các số trên

**2. Thuật toán**

Bước 1: Nhập các phần tử cho mảng và xuất ra

Bước 2: Tạo hàm tính USCLL,BSCNN

Bước 3: Đưa các phần tử trong mảng vào để tính

Bước 4: In kết quả chương trình

**3. Chương trình nguồn**

#include <stdio.h>

int uscln(int a, int b) {

while (b != 0) {

int t = b;

b = a % b;

a = t;

}

return a;

}

int bscnn(int a, int b) {

return (a \* b) / uscln(a, b);

}

void nhapMang(int A[], int n) {

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

printf("Nhập phần tử A[%d]: ", i);

scanf("%d", &A[i]);

}

}

void xuatMang(int A[], int n) {

printf("Các phần tử trong mảng: ");

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

printf("%d ", A[i]);

}

printf("\n");

}

int usclnMang(int A[], int n) {

int result = A[0];

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

result = uscln(result, A[i]);

}

return result;

}

int bscnnMang(int A[], int n) {

int result = A[0];

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

result = bscnn(result, A[i]);

}

return result;

}

int main() {

int n;

printf("Nhập số lượng phần tử của mảng: ");

scanf("%d", &n);

int A[n];

nhapMang(A, n);

xuatMang(A, n);

int usclnCuaMang = usclnMang(A, n);

printf("USCLN = %d\n", usclnCuaMang);

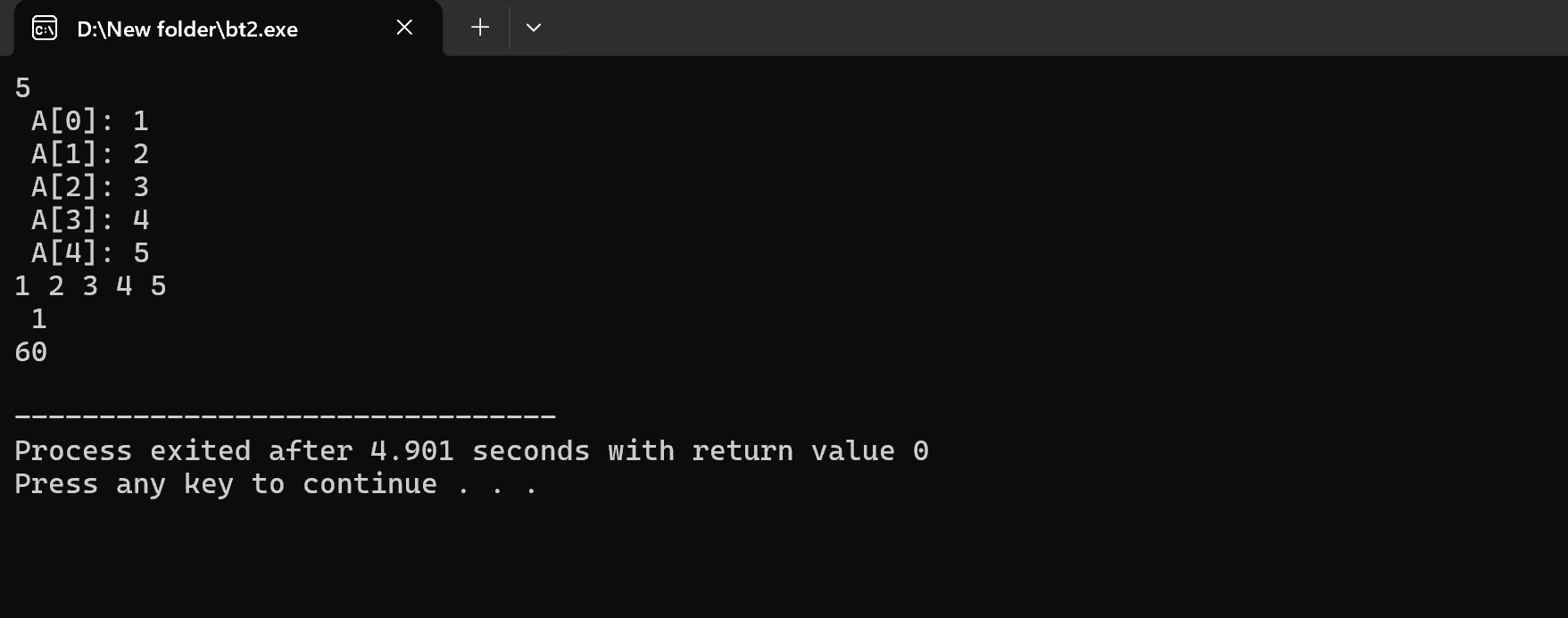
int bscnnCuaMang = bscnnMang(A, n);

printf("BSCNN = %d\n", bscnnCuaMang);

return 0;

}

**4. Kết quả chương trình**



**V. Bài 70**

**1. Tên đề bài**

Viết chương trình có sử dụng hàm  
a. Viết chương trình nhập hai mảng một chiều A và B.Xuất mảng A và B ra  
màn hình .  
b. Nối mảng A vào B (không dùng mảng phụ và không đươc sắp xếp trước và  
sau) đảm bảo thứ tự tăng dần. Xuất mảng sau khi nối ra màn hình.

**2. Thuật toán**

Bước 1: Dùng hàm nhập và cho nhập vào 2 mảng A,B

Bước 2: Dùng vòng lặp xuất ra các phần tử mảng

Bước 3: Nối và sắp xếp:

+Sao chép các phần tử của mảng A vào mảng mới từ đầu đến giữa và sao chép các phần tử của mảng B vào mảng mới từ giữa đến cuối.

+Sắp xếp từ theo thứ tự tăng dần

Bước 4: Xuất mảng ra sau khi nối.

**3. Chương trình nguồn**

#include <stdio.h>

void nhap\_mang(int mang[], int so\_phan\_tu, char ten\_mang) {

for (int i = 0; i < so\_phan\_tu; i++) {

printf("Nhập phần tử %d của mảng %c: ", i + 1, ten\_mang);

scanf("%d", &mang[i]);

}

}

void xuat\_mang(int mang[], int so\_phan\_tu, char ten\_mang) {

printf("Mảng %c: ", ten\_mang);

for (int i = 0; i < so\_phan\_tu; i++) {

printf("%d ", mang[i]);

}

printf("\n");

}

void noi\_mang(int a[], int size\_a, int b[], int size\_b, int mang\_noi[]) {

int i = 0, j = 0, k = 0;

while (i < size\_a && j < size\_b) {

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

mang\_noi[k++] = a[i++];

} else {

mang\_noi[k++] = b[j++];

}

}

while (i < size\_a) {

mang\_noi[k++] = a[i++];

}

while (j < size\_b) {

mang\_noi[k++] = b[j++];

}

}

int main() {

int size\_a, size\_b;

// Nhập kích thước của mảng A và B

printf("Nhập số phần tử của mảng A: ");

scanf("%d", &size\_a);

printf("Nhập số phần tử của mảng B: ");

scanf("%d", &size\_b);

int a[size\_a], b[size\_b], mang\_noi[size\_a + size\_b];

nhap\_mang(a, size\_a, 'A');

nhap\_mang(b, size\_b, 'B');

xuat\_mang(a, size\_a, 'A');

xuat\_mang(b, size\_b, 'B');

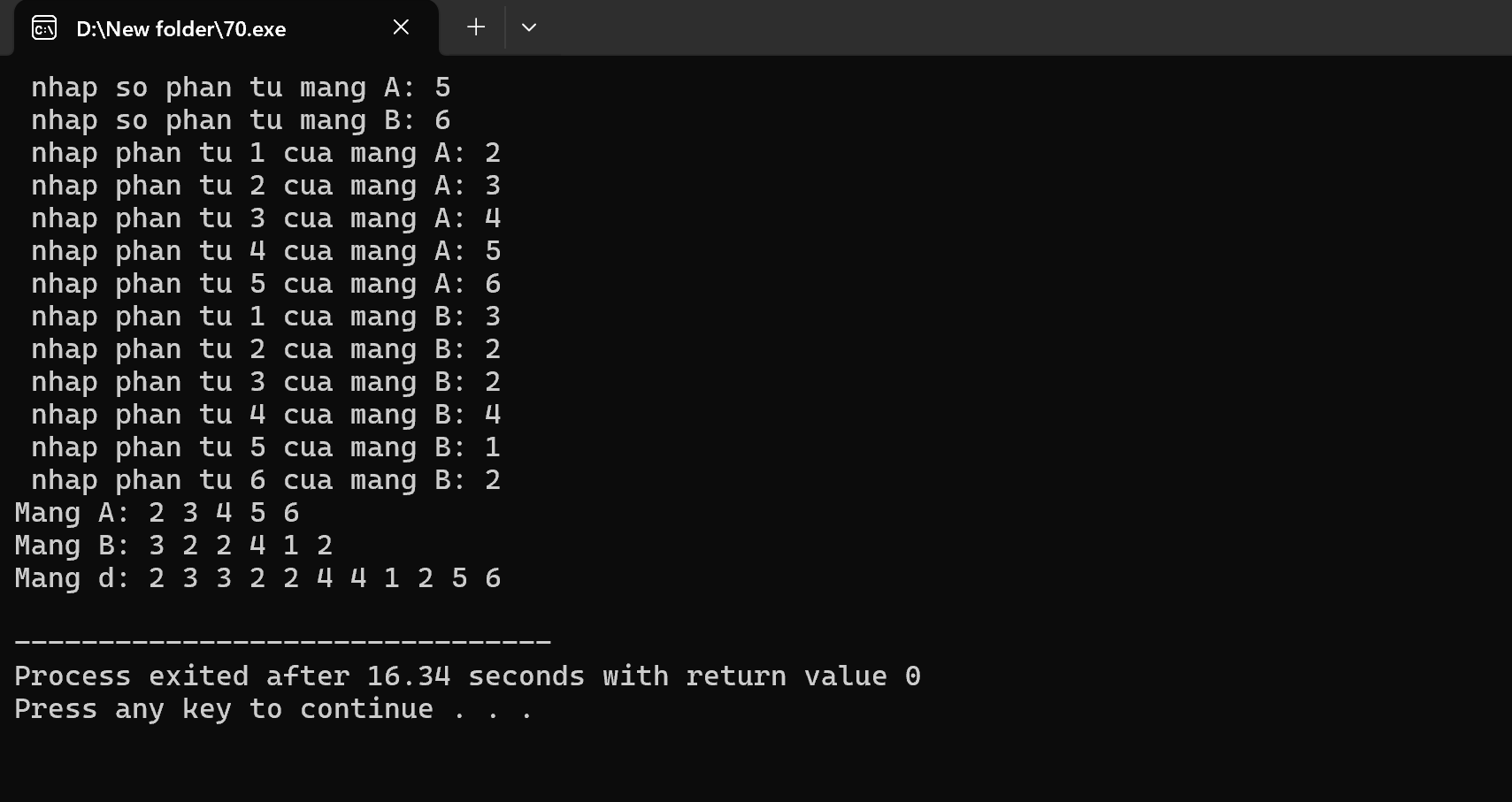
noi\_mang(a, size\_a, b, size\_b, mang\_noi);

xuat\_mang(mang\_noi, size\_a + size\_b, "A nối với B");

return 0;

}

**4. In kết quả chương trình**

****