Mục lục nội dung

[Mục lục hình ảnh 2](#_Toc163221689)

[BÀI TẬP TRÊN LAP 3](#_Toc163221691)

[**Bài tập 1.1.**Viết một chương trình C nhập vào 3 số nguyên. Thiết lập một con trỏ để lần lượt trỏ tới từng số nguyên và hiển thị kết quả giá trị tham chiếu ngược của con trỏ. 3](#_Toc163221692)

[**Bài tập 1.2.** Viết chương trình in ra địa chỉ của 5 phần tử đầu tiên trong mảng được định nghĩa sau đây: int a[7]= {13, -355, 235, 47, 67, 943, 1222}. 4](#_Toc163221693)

[**Bài tập 1.3.** Viết chương trình yêu cầu nhập giá trị cho 3 biến số nguyên x, y, z kiểu int. Sau đó sử dụng duy nhất một con trỏ để cộng giá trị của mỗi biến thêm 100. 6](#_Toc163221694)

[**Bài tập 1.4.** Viết hàm countEven(int\*, int) nhận một mảng số nguyên và kích thước của mảng, trả về số lượng số chẵn trong mảng. 8](#_Toc163221695)

[**Bài tập 1.5.** Viết hàm trả về con trỏ trỏ tới giá trị lớn nhất của một mảng các số double. Nếu mảng rỗng hãy trả về NULL. 9](#_Toc163221696)

[**Bài tập 1.6.** Viết hàm đảo ngược một mảng các số nguyên theo hai cách: dùng chỉ số và dùng con trỏ. 11](#_Toc163221697)

[**Bài tập 1.7.** Viết chương trình nhập vào một mảng các số nguyên với số lượng các phần tử nhập từ bàn phím. Sau đó sắp xếp mảng theo thứ tự tăng dần. Hiển thị danh sách mảng trước và sau khi sắp xếp. 14](#_Toc163221698)

[**Bài tập 1.8.** Viết chương trình nhập vào một ma trận 2 chiều kích thước m\*n với m và n nhập từ bàn phím. Sau đó đưa ra tổng các phần tử chẵn của ma trận đó. 16](#_Toc163221699)

[**BÀI TẬP VỀ NHÀ** 19](#_Toc163221700)

[**Bài tập 1.9.** Viết chương trình in ra tất cả các dãy con của một dãy cho trước. 19](#_Toc163221701)

[**Bài tập 1.10.** Viết chương trình nhập vào 2 ma trận vuông cùng kích thước n\*n, trong đó n nhập từ bàn phím. Sau đó tính tổng và tích của hai ma trận đó và đưa kết quả ra màn hình. 21](#_Toc163221702)

# Mục lục hình ảnh

[Hình 1 Bài 1.1 In ra giá trị tham chiếu ngược của con trỏ 3](#_Toc163221618)

[Hình 2 Bài 1.2 In ra địa chỉ của phần tử trong mảng 5](#_Toc163221619)

[Hình 3 Bài 1.3 Dùng con trỏ cộng giá trị mỗi biến thêm 100 6](#_Toc163221620)

[Hình 4 Bài 1.4 Hàm trả về số phần tử chẵn trong mảng 8](#_Toc163221622)

[Hình 5 Bài 1.5 Dùng con trỏ tìm phần tử lớn nhất trong mảng 9](#_Toc163221623)

[Hình 6 Bài 1.6 Hàm đảo ngược các phần tử trong mảng dùng chỉ số và con trỏ 12](#_Toc163221624)

[Hình 7 Bài 1.7 Dùng con trỏ sắp xếp mảng theo thứ tự tăng dần 14](#_Toc163221625)

[Hình 8 Bài 1.8 Nhập mảng 2 chiều rồi tính tổng các phần tử chẵn 16](#_Toc163221626)

[Hình 9 Bài 1.9. In ra tất cả các dãy con liên tiếp 19](#_Toc163221627)

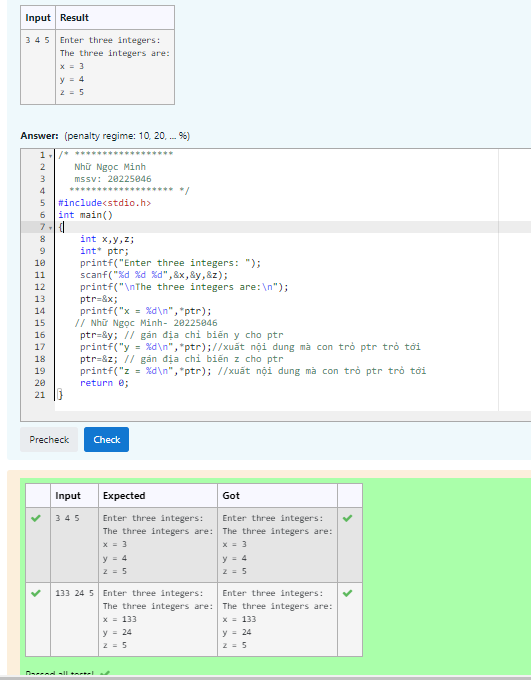
[Hình 10 Bài 1.10. Tổng và tích hai ma trận vuông cấp n 21](#_Toc163221628)

# Bài thực hành số XX – Tuần XX

# BÀI TẬP TRÊN LAP

## **Bài tập 1.1.**Viết một chương trình C nhập vào 3 số nguyên. Thiết lập một con trỏ để lần lượt trỏ tới từng số nguyên và hiển thị kết quả giá trị tham chiếu ngược của con trỏ.

**Lưu ý: Phép toán & trả về địa chỉ của biến**.



Hình 1 Bài 1.1 In ra giá trị tham chiếu ngược của con trỏ

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

Nhữ Ngọc Minh

mssv: 20225046

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

#include<stdio.h>

int main()

{

int x,y,z;

int\* ptr;

printf("Enter three integers: ");

scanf("%d %d %d",&x,&y,&z);

printf("\nThe three integers are:\n");

ptr=&x;

printf("x = %d\n",\*ptr);

// Nhữ Ngọc Minh- 20225046

ptr=&y; // gán địa chỉ biến y cho ptr

printf("y = %d\n",\*ptr);//xuất nội dung mà con trỏ ptr trỏ tới

ptr=&z; // gán địa chỉ biến z cho ptr

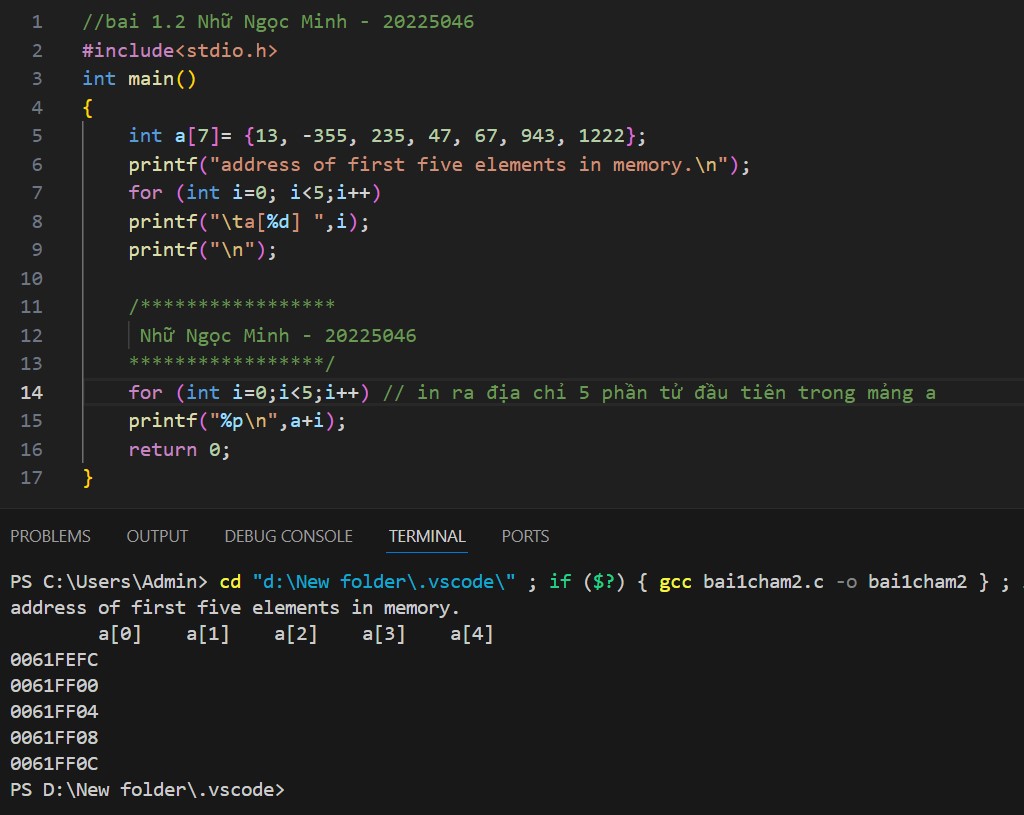
printf("z = %d\n",\*ptr); //xuất nội dung mà con trỏ ptr trỏ tới

return 0;

}

## **Bài tập 1.2.** Viết chương trình in ra địa chỉ của 5 phần tử đầu tiên trong mảng được định nghĩa sau đây: int a[7]= {13, -355, 235, 47, 67, 943, 1222}.

Lưu ý:  
Để in địa chỉ con trỏ các bạn sử dụng ký tự định dạng %p  
Để lấy địa chỉ của một biến ta có thể dùng phép toán &



Hình 2 Bài 1.2 In ra địa chỉ của phần tử trong mảng

//bai 1.2 Nhữ Ngọc Minh - 20225046

#include<stdio.h>

int main()

{

int a[7]= {13, -355, 235, 47, 67, 943, 1222};

printf("address of first five elements in memory.\n");

for (int i=0; i<5;i++)

printf("\ta[%d] ",i);

printf("\n");

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

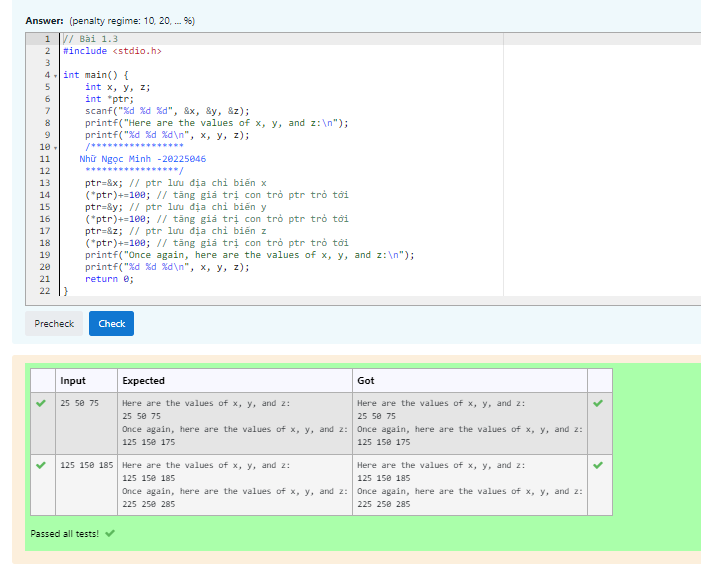
Nhữ Ngọc Minh - 20225046

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ for (int i=0;i<5;i++) // in ra địa chỉ 5 phần tử đầu tiên trong mảng a printf("%p\n",a+i);

return 0;

}

## **Bài tập 1.3.** Viết chương trình yêu cầu nhập giá trị cho 3 biến số nguyên x, y, z kiểu int. Sau đó sử dụng duy nhất một con trỏ để cộng giá trị của mỗi biến thêm 100.



Hình 3 Bài 1.3 Dùng con trỏ cộng giá trị mỗi biến thêm 100

// Bài 1.3

#include <stdio.h>

int main() {

int x, y, z;

int \*ptr;

scanf("%d %d %d", &x, &y, &z);

printf("Here are the values of x, y, and z:\n");

printf("%d %d %d\n", x, y, z);

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

Nhữ Ngọc Minh -20225046

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

ptr=&x; // ptr lưu địa chỉ biến x

(\*ptr)+=100; // tăng giá trị con trỏ ptr trỏ tới

ptr=&y; // ptr lưu địa chỉ biến y

(\*ptr)+=100; // tăng giá trị con trỏ ptr trỏ tới

ptr=&z; // ptr lưu địa chỉ biến z

(\*ptr)+=100; // tăng giá trị con trỏ ptr trỏ tới

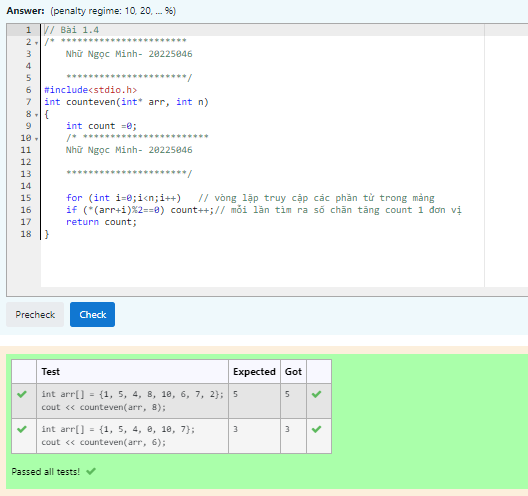
printf("Once again, here are the values of x, y, and z:\n");

printf("%d %d %d\n", x, y, z);

return 0;

}

## **Bài tập 1.4.** Viết hàm countEven(int\*, int) nhận một mảng số nguyên và kích thước của mảng, trả về số lượng số chẵn trong mảng.



Hình 4 Bài 1.4 Hàm trả về số phần tử chẵn trong mảng

// Bài 1.4

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

Nhữ Ngọc Minh- 20225046

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

#include<stdio.h>

int counteven(int\* arr, int n)

{

int count =0;

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

Nhữ Ngọc Minh- 20225046

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

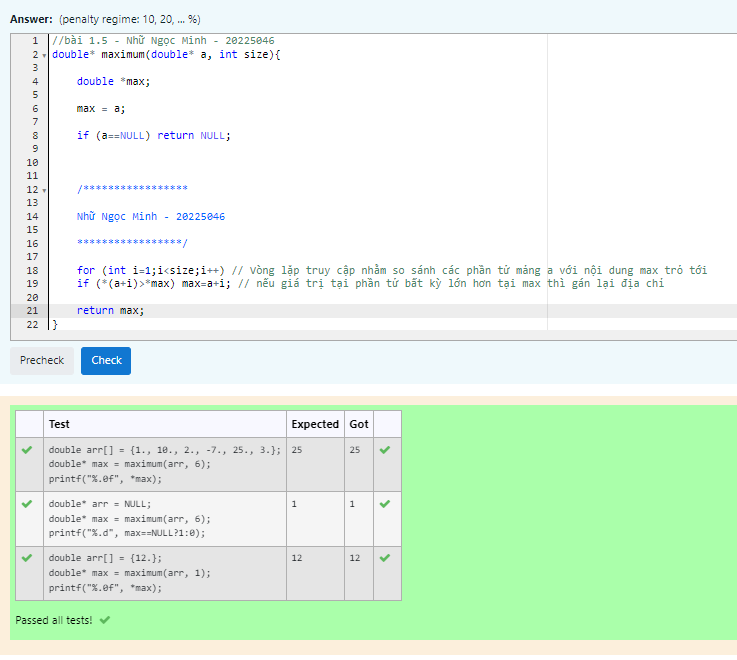
for (int i=0;i<n;i++) // vòng lặp truy cập các phần tử trong mảng

if (\*(arr+i)%2==0) count++;// mỗi lần tìm ra số chãn tăng count 1 đơn vị

return count;

}

## **Bài tập 1.5.** Viết hàm trả về con trỏ trỏ tới giá trị lớn nhất của một mảng các số double. Nếu mảng rỗng hãy trả về NULL.



Hình 5 Bài 1.5 Dùng con trỏ tìm phần tử lớn nhất trong mảng

//bài 1.5 - Nhữ Ngọc Minh - 20225046

double\* maximum(double\* a, int size){

double \*max;

max = a;

if (a==NULL) return NULL;

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

Nhữ Ngọc Minh - 20225046

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

for (int i=1;i<size;i++) // Vòng lặp truy cập nhằm so sánh các phần tử mảng a với nội dung max trỏ tới

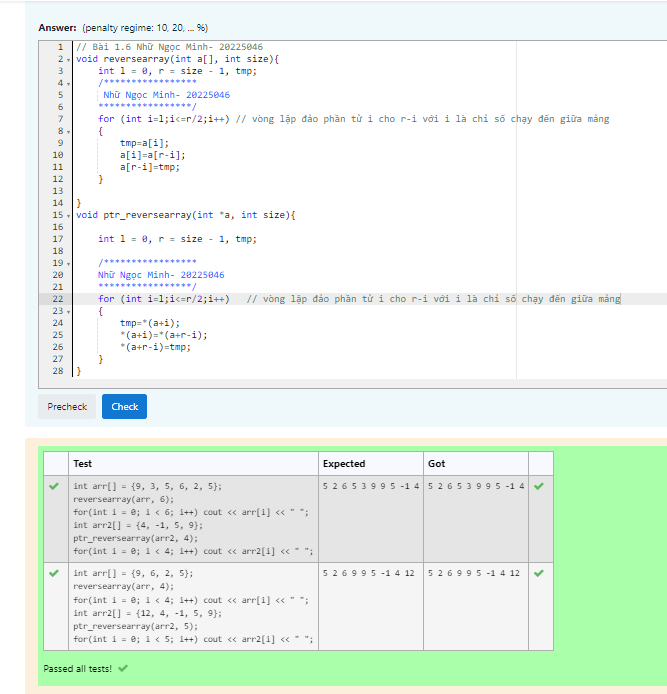
if (\*(a+i)>\*max) max=a+i; // nếu giá trị tại phần tử bất kỳ lớn hơn tại max thì gán lại địa chỉ

return max;

}

## **Bài tập 1.6.** Viết hàm đảo ngược một mảng các số nguyên theo hai cách: dùng chỉ số và dùng con trỏ.

**Ví dụ mảng đầu vào là [9, -1, 4, 5, 7] thì kết quả là [7, 5, 4, -1, 9].**



Hình 6 Bài 1.6 Hàm đảo ngược các phần tử trong mảng dùng chỉ số và con trỏ

// Bài 1.6 Nhữ Ngọc Minh- 20225046

void reversearray(int a[], int size){

int l = 0, r = size - 1, tmp;

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

Nhữ Ngọc Minh- 20225046

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

for (int i=l;i<=r/2;i++) // vòng lặp đảo phần tử i cho r-i với i là chỉ số chạy đến giữa mảng

{

tmp=a[i];

a[i]=a[r-i];

a[r-i]=tmp;

}

}

void ptr\_reversearray(int \*a, int size){

int l = 0, r = size - 1, tmp;

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

Nhữ Ngọc Minh- 20225046

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

for (int i=l;i<=r/2;i++) // vòng lặp đảo phần tử i cho r-i với i là chỉ số chạy đến giữa mảng

{

tmp=\*(a+i);

\*(a+i)=\*(a+r-i);

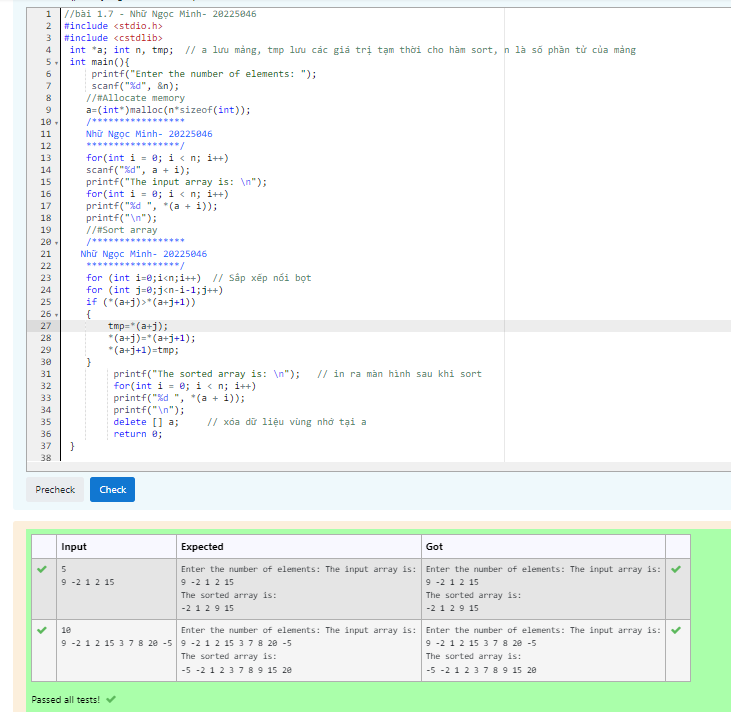
\*(a+r-i)=tmp;

}

}

## **Bài tập 1.7.** Viết chương trình nhập vào một mảng các số nguyên với số lượng các phần tử nhập từ bàn phím. Sau đó sắp xếp mảng theo thứ tự tăng dần. Hiển thị danh sách mảng trước và sau khi sắp xếp.

Yêu cầu chỉ sử dụng con trỏ để truy cập mảng, không truy cập theo index mảng.



Hình 7 Bài 1.7 Dùng con trỏ sắp xếp mảng theo thứ tự tăng dần

//bài 1.7 - Nhữ Ngọc Minh- 20225046

#include <stdio.h>

#include <cstdlib>

int \*a; int n, tmp; // a lưu mảng, tmp lưu các giá trị tạm thời cho hàm sort, n là số phần tử của mảng

int main(){

printf("Enter the number of elements: ");

scanf("%d", &n);

//#Allocate memory

a=(int\*)malloc(n\*sizeof(int));

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

Nhữ Ngọc Minh- 20225046

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

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

scanf("%d", a + i);

printf("The input array is: \n");

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

printf("%d ", \*(a + i));

printf("\n");

//#Sort array

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

Nhữ Ngọc Minh- 20225046

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

for (int i=0;i<n;i++) // Sắp xếp nổi bọt

for (int j=0;j<n-i-1;j++)

if (\*(a+j)>\*(a+j+1))

{

tmp=\*(a+j);

\*(a+j)=\*(a+j+1);

\*(a+j+1)=tmp;

}

printf("The sorted array is: \n"); // in ra màn hình sau khi sort

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

printf("%d ", \*(a + i));

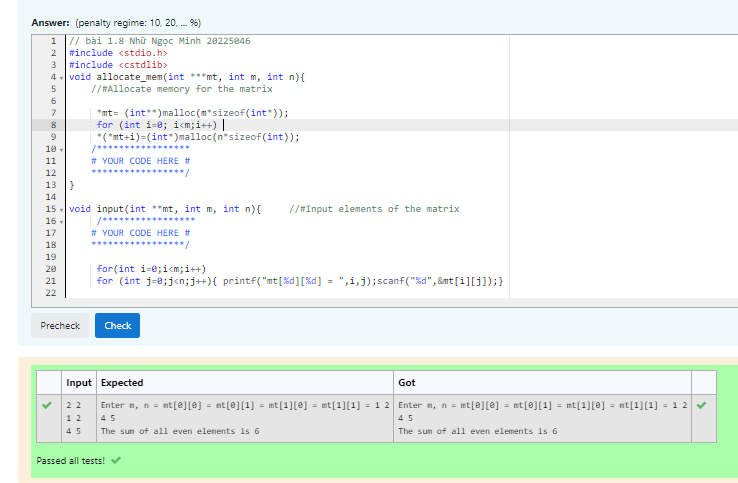
printf("\n");

delete [] a; // xóa dữ liệu vùng nhớ tại a

return 0;

}

## **Bài tập 1.8.** Viết chương trình nhập vào một ma trận 2 chiều kích thước m\*n với m và n nhập từ bàn phím. Sau đó đưa ra tổng các phần tử chẵn của ma trận đó.



Hình 8 Bài 1.8 Nhập mảng 2 chiều rồi tính tổng các phần tử chẵn

// bài 1.8 Nhữ Ngọc Minh 20225046

#include <stdio.h>

#include <cstdlib>

void allocate\_mem(int \*\*\*mt, int m, int n){

//#Allocate memory for the matrix

\*mt= (int\*\*)malloc(m\*sizeof(int\*));

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

\*(\*mt+i)=(int\*)malloc(n\*sizeof(int));

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

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

}

void input(int \*\*mt, int m, int n){ //#Input elements of the matrix

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

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

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

for (int j=0;j<n;j++){ printf("mt[%d][%d] = ",i,j);scanf("%d",&mt[i][j]);}

} void output(int \*\*mt, int m, int n){ //# Print all elements of the matrix

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

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

printf("%d ",mt[i][j]);

printf("\n");

}

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

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

} int process(int \*\*mt, int m, int n){ int tong = 0;

//# Calculate the sum of all even elements in the matrix

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

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

if (\*(\*(mt+i)+j) %2==0) tong+=\*(\*(mt+i)+j);

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

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

return tong;

} void free\_mem(int \*\*mt, int m, int n){

//# Free memory

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

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

for (int i=0;i<m;i++) free(\*(mt+i));

free(mt);

} int main(){

int m, n, \*\*mt;

printf("Enter m, n = ");

scanf("%d%d", &m, &n);

allocate\_mem(&mt, m, n);

input(mt, m, n);

output(mt, m, n);

printf("The sum of all even elements is %d", process(mt, m, n)); free\_mem(mt, m, n);

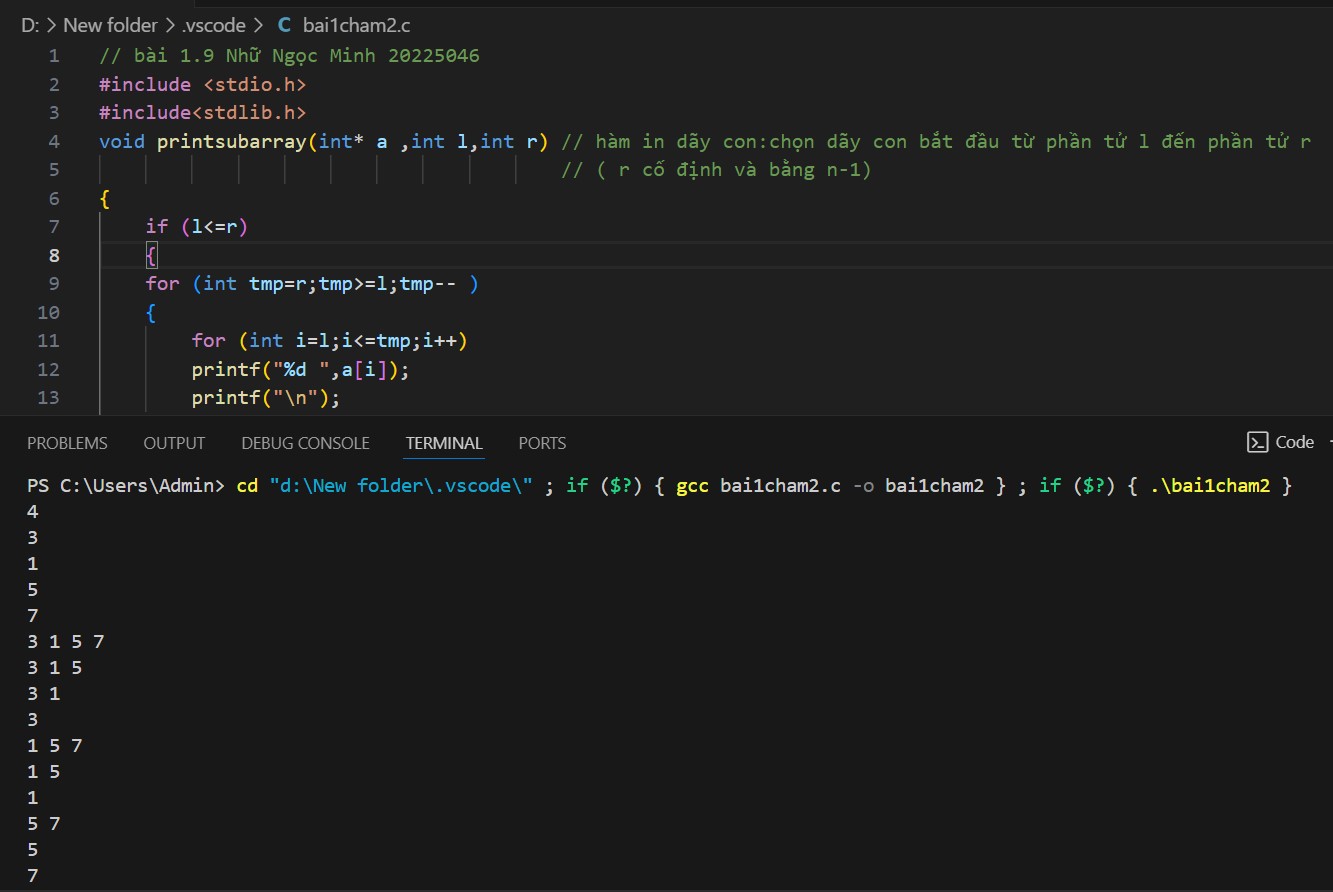
return 0;

}

# **BÀI TẬP VỀ NHÀ**

Chụp ảnh kết quả của tất cả các test.

## **Bài tập 1.9.** Viết chương trình in ra tất cả các dãy con của một dãy cho trước.



Hình 9 Bài 1.9 In ra tất cả các dãy con liên tiếp

// bài 1.9 Nhữ Ngọc Minh 20225046

#include <stdio.h>

#include<stdlib.h>

void printsubarray(int\* a ,int l,int r)

// hàm in dãy con:chọn dãy con bắt đầu từ phần tử l đến phần tử r

// ( r cố định và bằng n-1)

{

if (l<=r)

{

for (int tmp=r;tmp>=l;tmp-- )

{ for (int i=l;i<=tmp;i++)

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

printf("\n");

}

printsubarray(a,l+1,r);

}

}

int main()

{

int n;

scanf("%d",&n);

int\* a=(int\*)malloc(n\*sizeof(int));

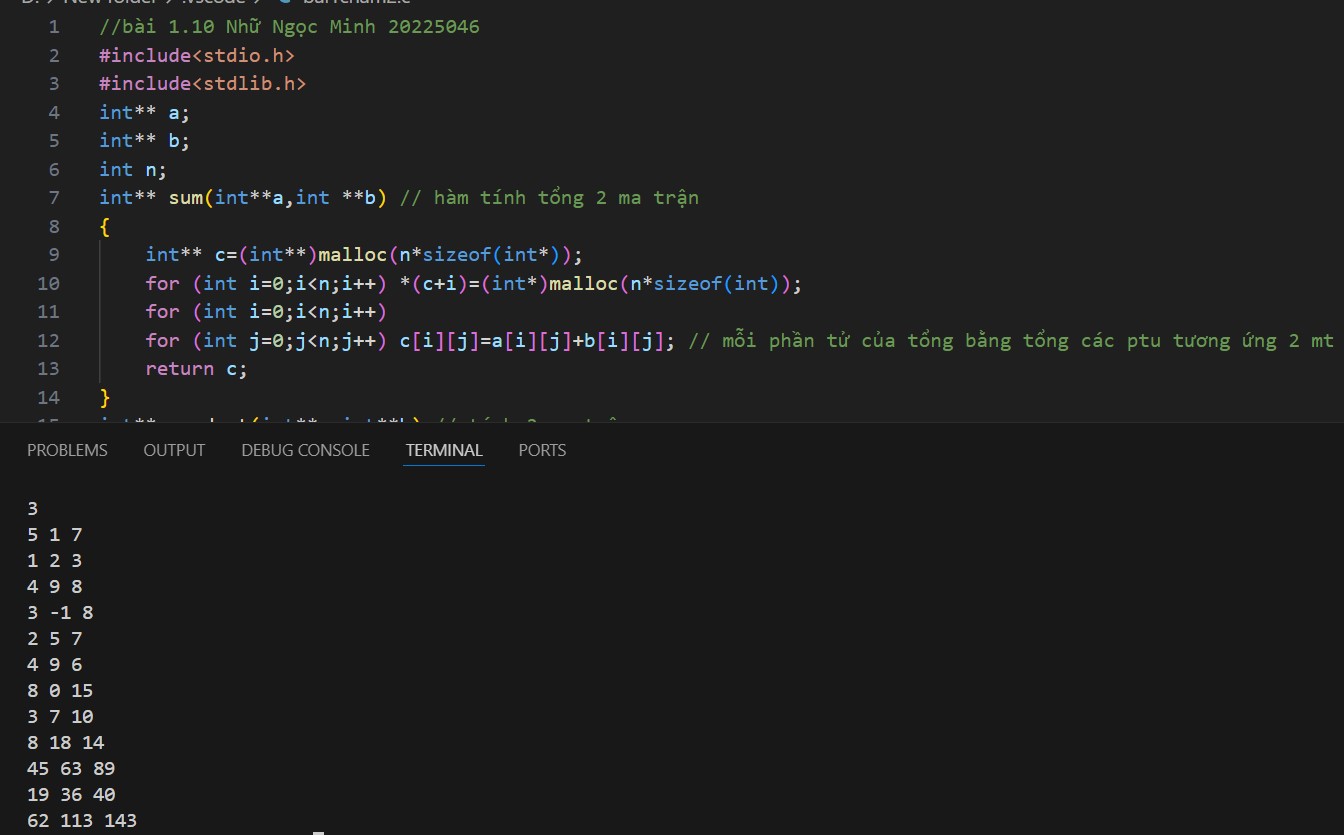
for (int i=0;i<n;i++) scanf("%d",&a[i]);

printsubarray(a,0,n-1);

return 0;

}

## **Bài tập 1.10.** Viết chương trình nhập vào 2 ma trận vuông cùng kích thước n\*n, trong đó n nhập từ bàn phím. Sau đó tính tổng và tích của hai ma trận đó và đưa kết quả ra màn hình.



Hình 10 Bài 1.10 Tổng và tích hai ma trận vuông cấp n

//bài 1.10 Nhữ Ngọc Minh 20225046

#include<stdio.h>

#include<stdlib.h>

int\*\* a;

int\*\* b;

int n;

int\*\* sum(int\*\*a,int \*\*b) // hàm tính tổng 2 ma trận

{

int\*\* c=(int\*\*)malloc(n\*sizeof(int\*));

for (int i=0;i<n;i++) \*(c+i)=(int\*)malloc(n\*sizeof(int));

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

for (int j=0;j<n;j++) c[i][j]=a[i][j]+b[i][j]; // mỗi phần tử của tổng bằng tổng các ptu tương ứng 2 mt

return c;

}

int\*\* product(int\*\*a,int\*\*b) // tích 2 ma trận

{

int\*\* c=(int\*\*)malloc(n\*sizeof(int\*));

for (int i=0;i<n;i++) \*(c+i)=(int\*)malloc(n\*sizeof(int));

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

for (int j=0;j<n;j++) //mỗi phần tử c[i][j]=tổng sigma a[i][k]\*b[k][j] với k từ 1 đến n-1

{ c[i][j]=0;

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

{

c[i][j]+=a[i][k]\*b[k][j];

}

}

return c;

}

void print(int\*\* a) // hàm in ra ma trận

{

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

{

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

printf("%d ",a[i][j]);

printf("\n");

}

}

int main()

{

scanf("%d",&n);// đọc n

a=(int\*\*)malloc(n\*sizeof(int\*));

for (int i=0;i<n;i++) \*(a+i)=(int\*)malloc(n\*sizeof(int)); // cấp phát mảng a

b=(int\*\*)malloc(n\*sizeof(int\*));

for (int i=0;i<n;i++) \*(b+i)=(int\*)malloc(n\*sizeof(int)); // cấp phát mảng b

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

for (int j=0;j<n;j++) scanf("%d",&a[i][j]);// đọc mảng a

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

for (int j=0;j<n;j++) scanf("%d",&b[i][j]);// đọc mảng b

print(sum(a,b)); // in ma trận tổng

print(product(a,b));// in ma trận tích

return 0;

}