Mục lục nội dung

[Bài thực hành số 01 – Tuần 32 3](#_Toc161238431)

[BÀI TẬP TRÊN LAP 3](#_Toc161238432)

[**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](#_Toc161238433)

[**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}. 5](#_Toc161238434)

[**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. 7](#_Toc161238435)

[**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. 7](#_Toc161238436)

[**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. 7](#_Toc161238437)

[**Bài tập 1.6.** Viết hàm đảo ngược một mảng các số nguyên theo hai cách: dùng chỉ số và dùng con trỏ. 7](#_Toc161238438)

[**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. 8](#_Toc161238439)

[**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. 8](#_Toc161238440)

[**BÀI TẬP VỀ NHÀ** 8](#_Toc161238441)

[**Bài tập 1.9.** Viết chương trình in ra tất cả các dãy con của một dãy cho trước. 8](#_Toc161238442)

[**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. 8](#_Toc161238443)

Mục lục hình ảnh

[Hình 1 Bài 1.1 In ra giá trị tham chiếu ngược của con trỏ 4](#_Toc161238452)

[Hình 2 Bài 1.2 In ra địa chỉ của phần tử trong mảng 6](#_Toc161238453)

[Hình 3 Bài 1.3 Dùng con trỏ cộng giá trị mỗi biến thêm 100 7](#_Toc161238454)

[Hình 4 Bài 1.4 Hàm trả về số phần tử chẵn trong mảng 7](#_Toc161238455)

[Hình 5 Bài 1.5 Dùng con trỏ tìm phần tử lớn nhất trong mảng 7](#_Toc161238456)

[Hình 6 Bài 1.6 Hàm đảo ngược các phần tử trong mảng dùng chỉ số và con trỏ 7](#_Toc161238457)

[Hình 7 Bài 1.7 Dùng con trỏ sắp xếp mảng theo thứ tự tăng dần 8](#_Toc161238458)

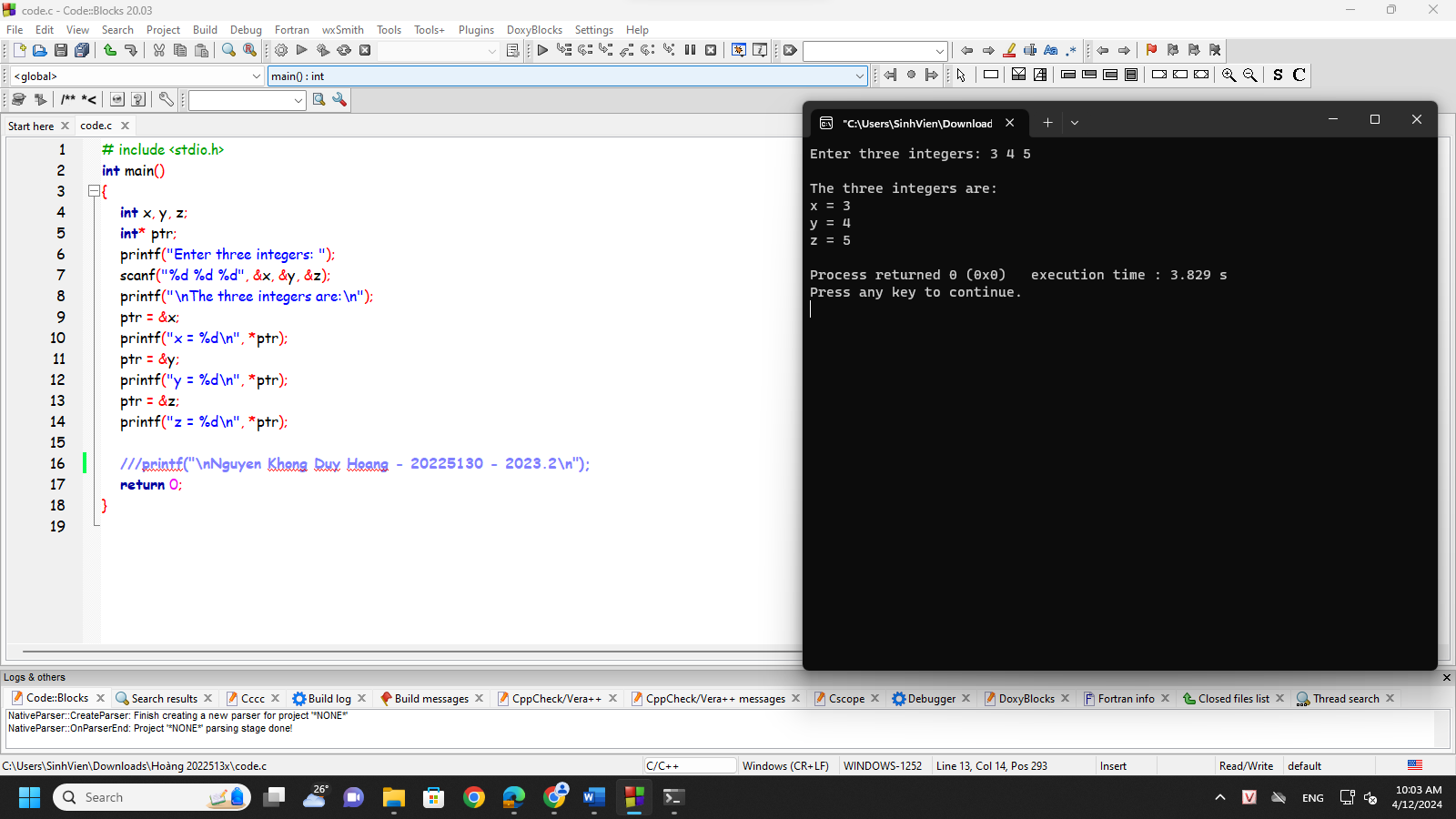
[Hình 8 Bài 1.8 Nhập mảng 2 chiều rồi tính tổng các phần tử chẵn 8](#_Toc161238459)

# Bài thực hành số 01 – Tuần 32

# 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**.

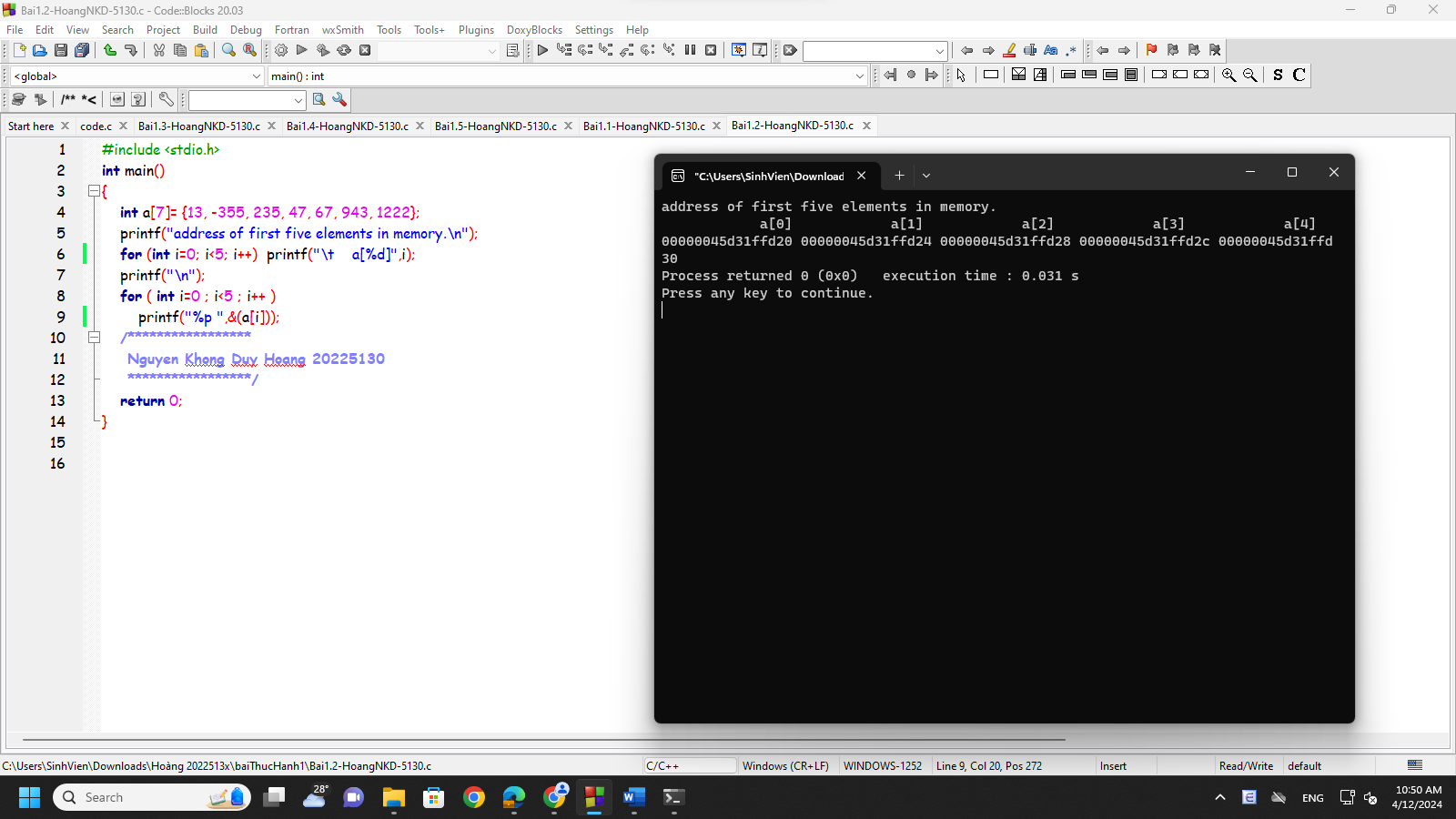


Hình 1 Bài 1.1 In ra giá trị tham chiếu ngược của con trỏ

|  |
| --- |
| # 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);  ptr = &y;  printf("y = %d\n", \*ptr);  ptr = &z;  printf("z = %d\n", \*ptr);  printf("\nNguyen Khong Duy Hoang - 20225130 - 2023.2\n");  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}.

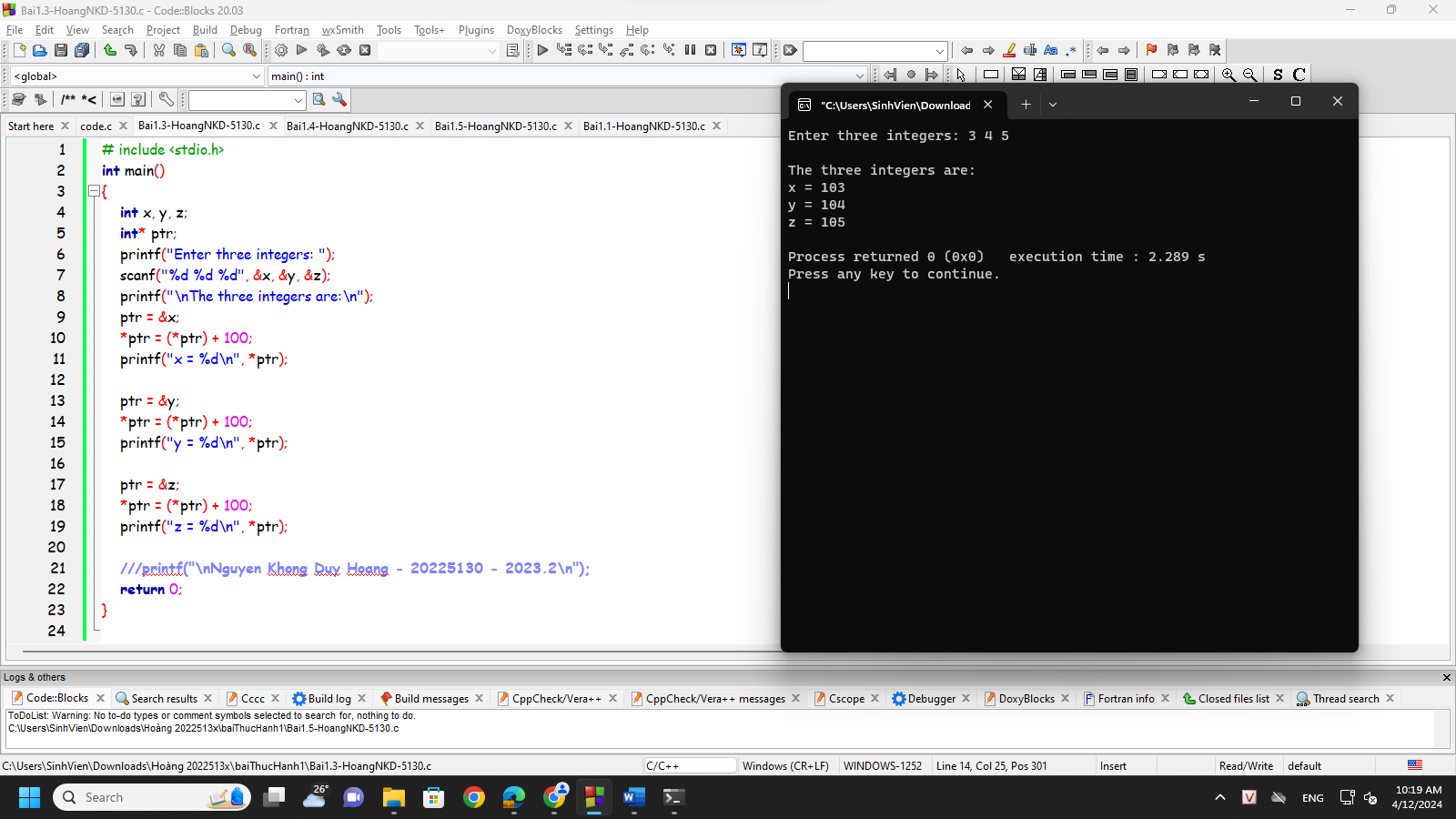
Lưu ý:  
Để in địa chỉ con trỏ các bạn sử dụng ký tự định dạng %p  
Để lấy địa chỉ của một biến ta có thể dùng phép toán &



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

|  |
| --- |
| #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("\t a[%d]",i);  printf("\n");  for ( int i=0 ; i<5 ; i++ )  printf("%p ",&(a[i]));  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Nguyen Khong Duy Hoang 20225130  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  return 0;  } |

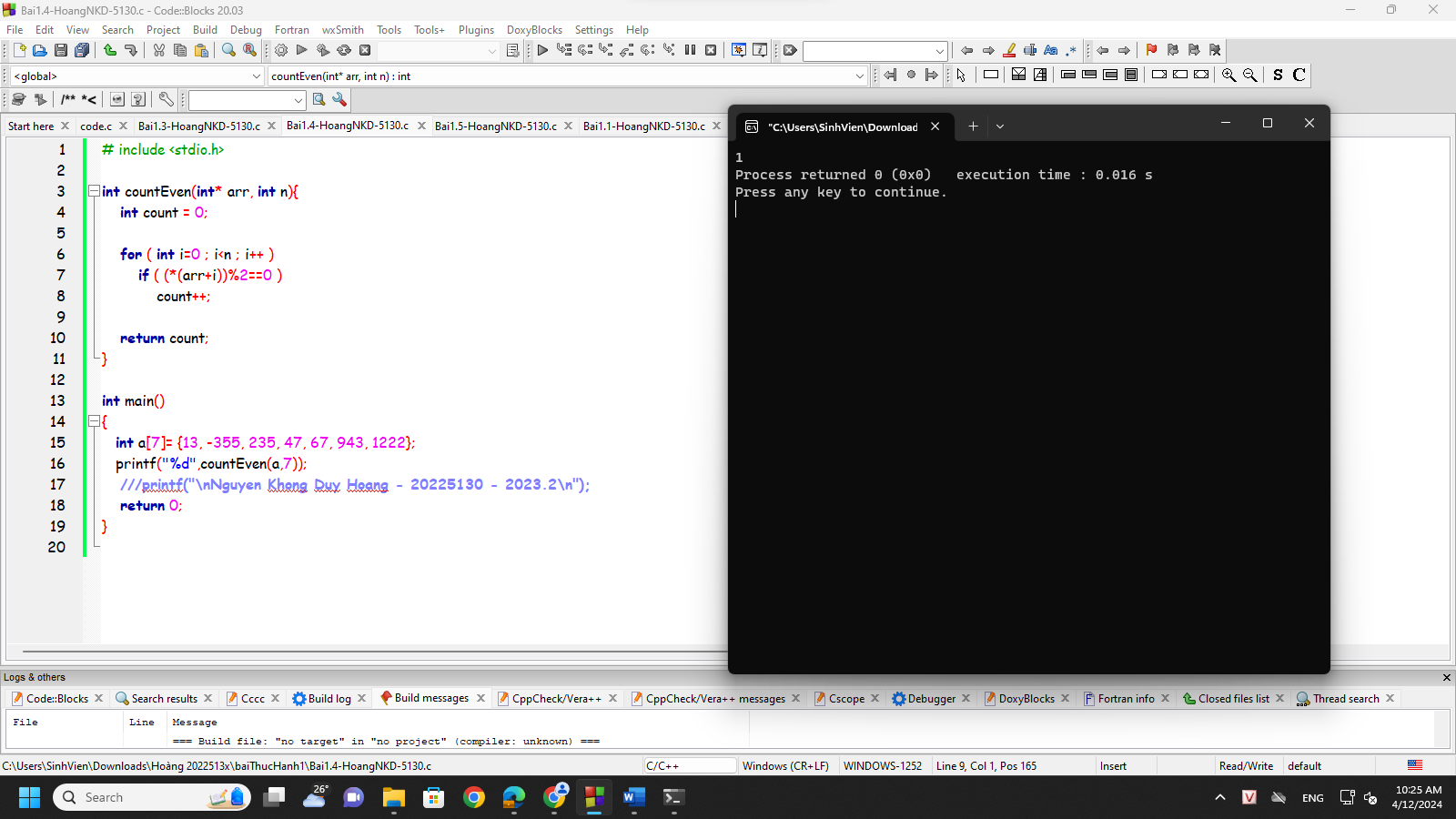
## **Bài tập 1.3.** Viết chương trình yêu cầu nhập giá trị cho 3 biến số nguyên x, y, z kiểu int. Sau đó sử dụng duy nhất một con trỏ để cộng giá trị của mỗi biến thêm 100.



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

|  |
| --- |
| # 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;  \*ptr = (\*ptr) + 100;  printf("x = %d\n", \*ptr);  ptr = &y;  \*ptr = (\*ptr) + 100;  printf("y = %d\n", \*ptr);  ptr = &z;  \*ptr = (\*ptr) + 100;  printf("z = %d\n", \*ptr);  ///printf("\nNguyen Khong Duy Hoang - 20225130 - 2023.2\n");  return 0;  } |

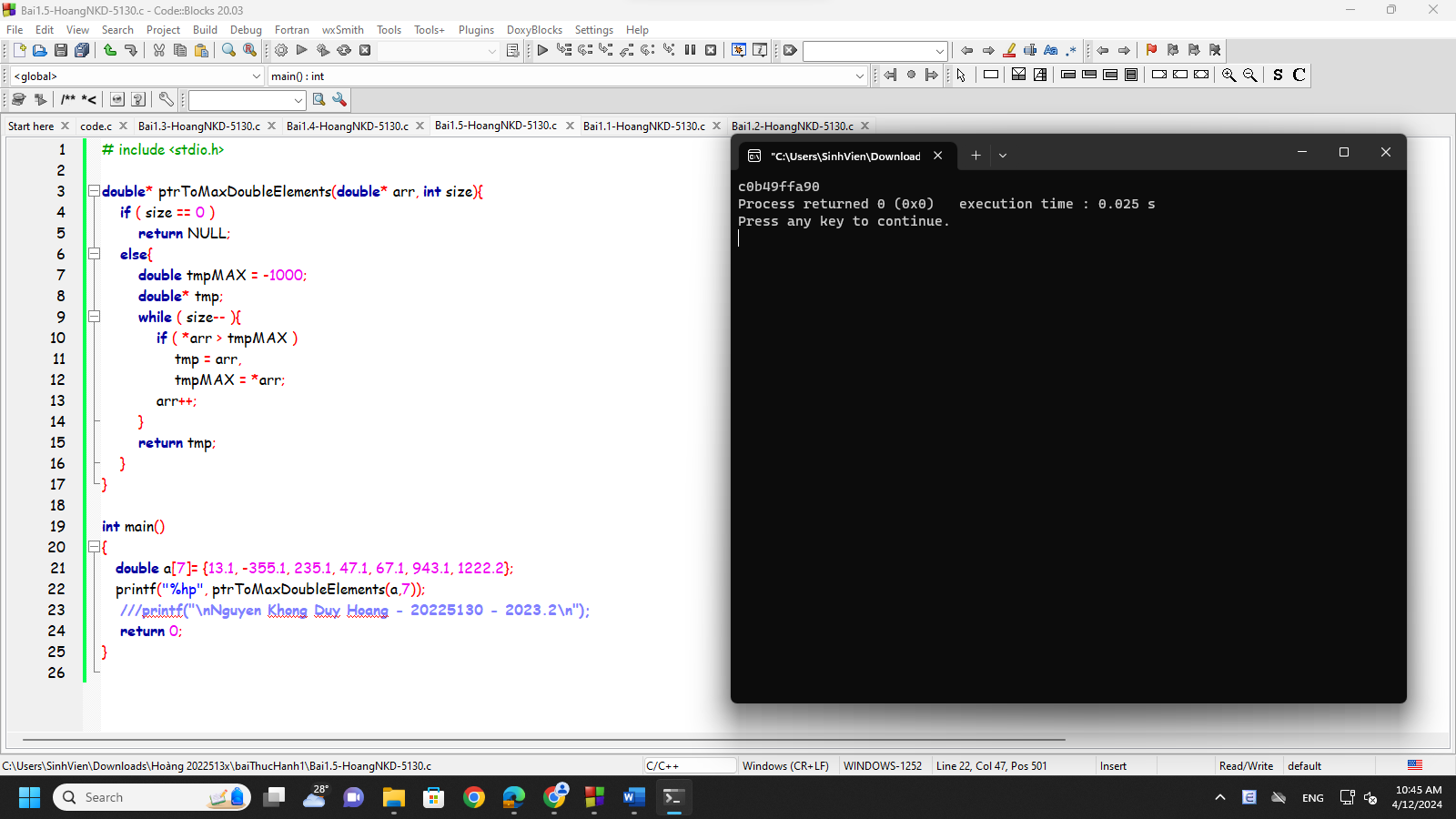
## **Bài tập 1.4.** Viết hàm countEven(int\*, int) nhận một mảng số nguyên và kích thước của mảng, trả về số lượng số chẵn trong mảng.



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

|  |
| --- |
| # include <stdio.h>  int countEven(int\* arr, int n){  int count = 0;  for ( int i=0 ; i<n ; i++ )  if ( (\*(arr+i))%2==0 )  count++;  return count;  }  int main()  {  int a[7]= {13, -355, 235, 47, 67, 943, 1222};  printf("%d",countEven(a,7));  ///printf("\nNguyen Khong Duy Hoang - 20225130 - 2023.2\n");  return 0;  } |

## **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.

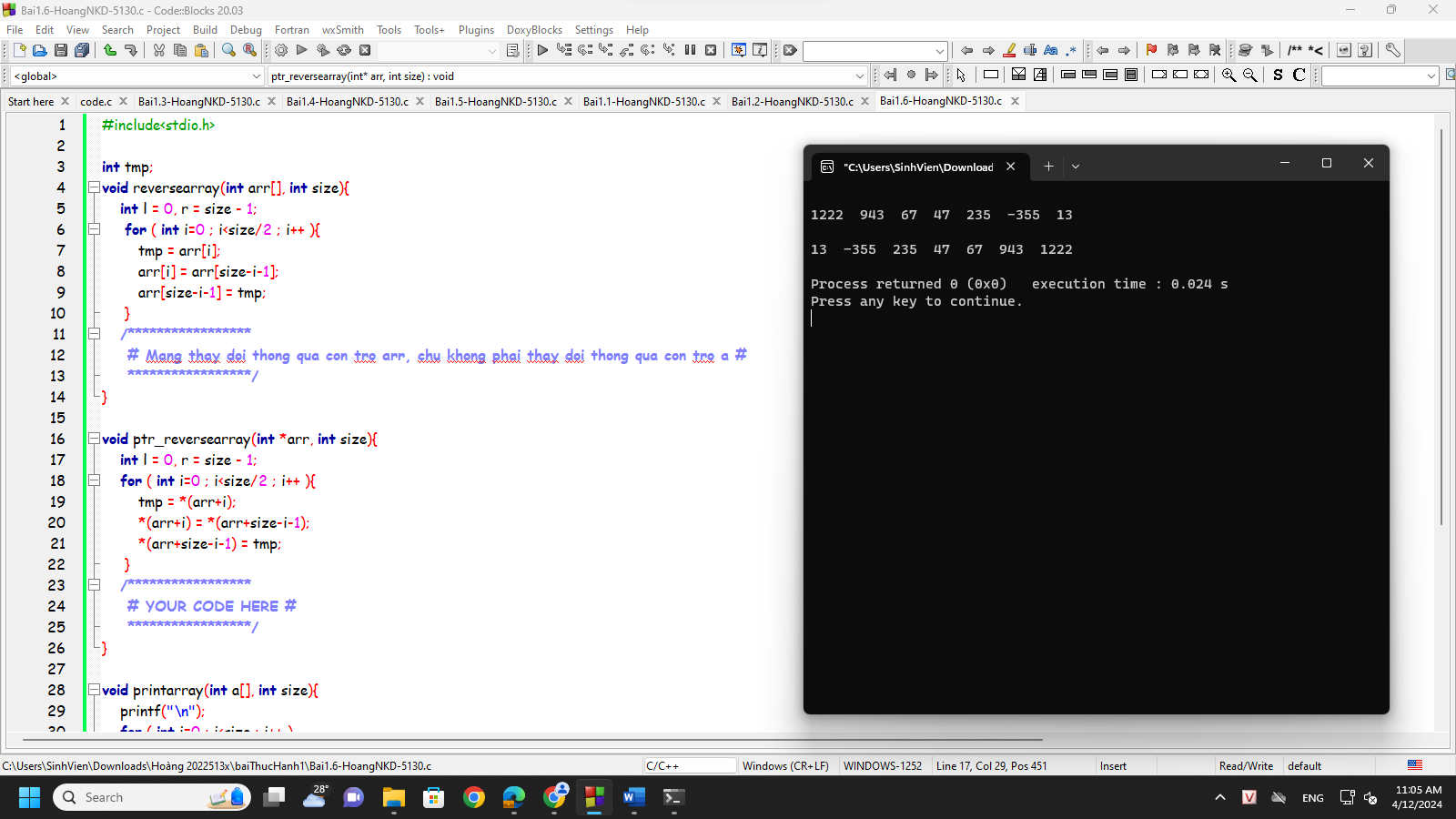


Hình 5 Bài 1.5 Dùng con trỏ tìm phần tử lớn nhất trong mảng

|  |
| --- |
| # include <stdio.h>  double\* ptrToMaxDoubleElements(double\* arr, int size){  if ( size == 0 )  return NULL;  else{  double tmpMAX = -1000;  double\* tmp;  while ( size-- ){  if ( \*arr > tmpMAX )  tmp = arr,  tmpMAX = \*arr;  arr++;  }  return tmp;  }  }  int main()  {  double a[7]= {13.1, -355.1, 235.1, 47.1, 67.1, 943.1, 1222.2};  printf("%hp", ptrToMaxDoubleElements(a,7));  ///printf("\nNguyen Khong Duy Hoang - 20225130 - 2023.2\n");  return 0;  } |

## **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

|  |
| --- |
| #include<stdio.h>  int tmp;  void reversearray(int arr[], int size){  int l = 0, r = size - 1;  for ( int i=0 ; i<size/2 ; i++ ){  tmp = arr[i];  arr[i] = arr[size-i-1];  arr[size-i-1] = tmp;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Mang thay doi thong qua con tro arr, chu khong phai thay doi thong qua con tro a #  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  }  void ptr\_reversearray(int \*arr, int size){  int l = 0, r = size - 1;  for ( int i=0 ; i<size/2 ; i++ ){  tmp = \*(arr+i);  \*(arr+i) = \*(arr+size-i-1);  \*(arr+size-i-1) = tmp;  }  }  void printarray(int a[], int size){  printf("\n");  for ( int i=0 ; i<size ; i++ )  printf("%d ", a[i]);  printf("\n");  }  int main(){  int a[7]= {13, -355, 235, 47, 67, 943, 1222};  reversearray(a,7);  printarray(a,7);  ptr\_reversearray(a,7);  printarray(a,7);  return 0;  } |

## **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.

A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  int \*a;  int n, tmp;  void inoutfile(){  #define taskname "hello123"  if ( fopen(taskname".inp","r")){  freopen(taskname".inp","r",stdin);  freopen(taskname".out","w",stdout);  }  }  int main(){  ///inoutfile();  printf("Enter the number of elements: ");  scanf("%d", &n);  //#Allocate memory  a = (int \*)malloc(n\*sizeof(int));  ///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //# YOUR CODE HERE #  ///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  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  int tmp;  for ( int i=0 ; i<n ; i++ )  for ( int j=0 ; j<i ; j++ ){  if ( \*(a+i) > \*(a+j) ){  tmp = \*(a+i);  \*(a+i) = \*(a+j);  \*(a+j) = tmp;  }  }  printf("The sorted array is: \n");  for(int i = 0; i < n; i++)  printf("%d ", \*(a + i));  printf("\n");  free(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.

A computer screen shot of a computer screen

Description automatically generated

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

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  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++ )  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("mt[%d][%d] = ",i,j);  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 = 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", &m);  scanf("%d", &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.

A computer screen shot of a computer screen

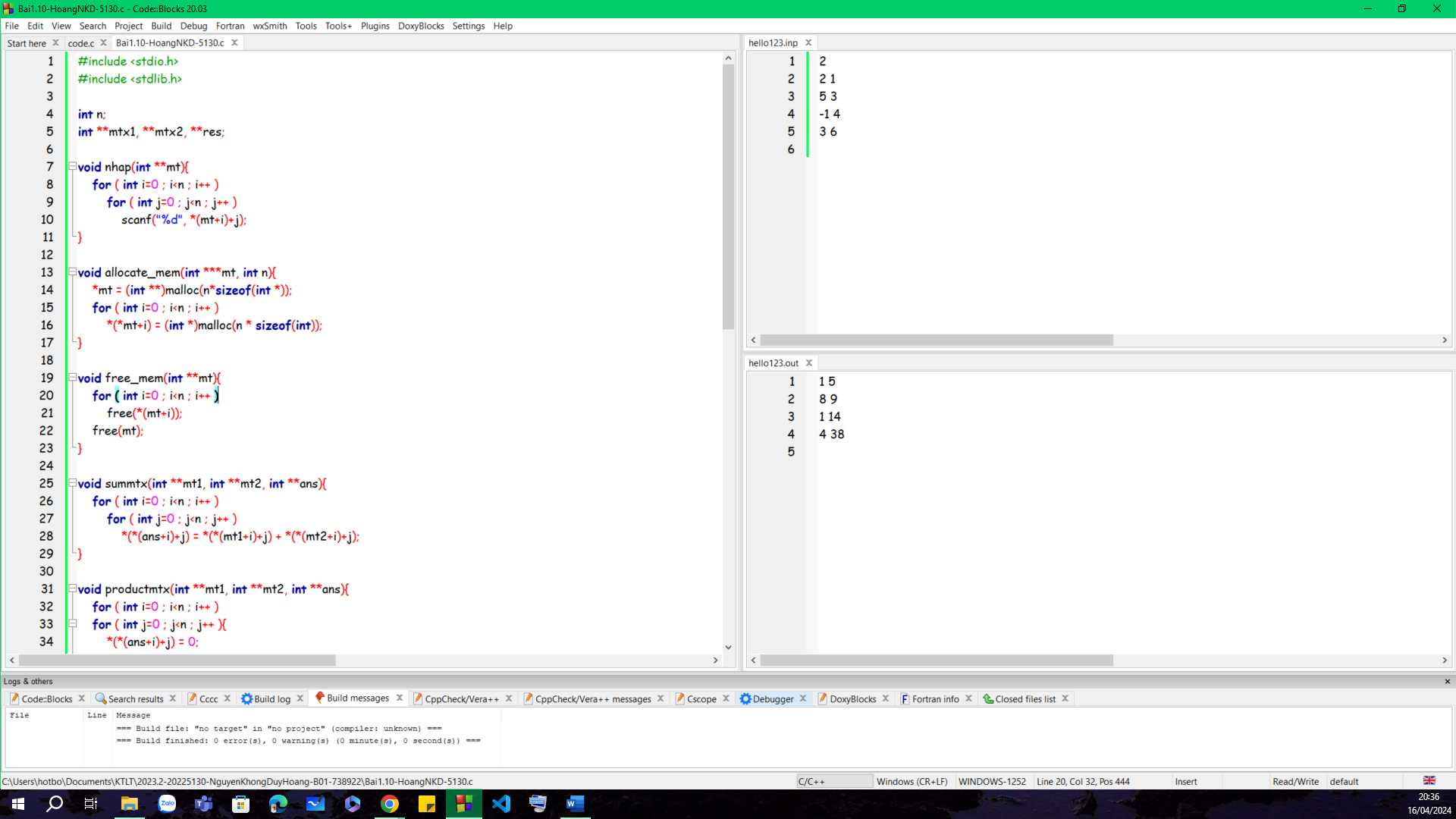
Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  int main(){  int n=0;  int arr[1000];  scanf("%d",&n);  for ( int i=0 ; i<n ; i++ )  scanf("%d",&arr[i]);  for ( int i=0 ; i<n ; i++ )  for ( int j=i ; j<n ; j++ ){  for ( int tmp = i ; tmp<=j ; tmp++ )  printf("%d ",arr[tmp]);  printf("\n");  }  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.



A screenshot of a computer

Description automatically generated

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  int n;  int \*\*mtx1, \*\*mtx2, \*\*res;  void nhap(int \*\*mt){  for ( int i=0 ; i<n ; i++ )  for ( int j=0 ; j<n ; j++ )  scanf("%d", \*(mt+i)+j);  }  void allocate\_mem(int \*\*\*mt, int n){  \*mt = (int \*\*)malloc(n\*sizeof(int \*));  for ( int i=0 ; i<n ; i++ )  \*(\*mt+i) = (int \*)malloc(n \* sizeof(int));  }  void free\_mem(int \*\*mt){  for ( int i=0 ; i<n ; i++ )  free(\*(mt+i));  free(mt);  }  void summtx(int \*\*mt1, int \*\*mt2, int \*\*ans){  for ( int i=0 ; i<n ; i++ )  for ( int j=0 ; j<n ; j++ )  \*(\*(ans+i)+j) = \*(\*(mt1+i)+j) + \*(\*(mt2+i)+j);  }  void productmtx(int \*\*mt1, int \*\*mt2, int \*\*ans){  for ( int i=0 ; i<n ; i++ )  for ( int j=0 ; j<n ; j++ ){  \*(\*(ans+i)+j) = 0;  for ( int k=0 ; k<n ; k++ )  \*(\*(ans+i)+j) = \*(\*(ans+i)+j) + \*(\*(mt1+i)+k) \* \*(\*(mt2+k)+j);  }  }  int main(){  #define taskname "hello123"  if ( fopen(taskname".inp","r"))  freopen(taskname".inp","r",stdin),  freopen(taskname".out","w",stdout);  scanf("%d",&n);  allocate\_mem(&mtx1, n);  allocate\_mem(&mtx2, n);  allocate\_mem(&res, n);  nhap(mtx1);  nhap(mtx2);  summtx(mtx1,mtx2,res);  for ( int i=0 ; i<n ; i++ ){  for ( int j=0 ; j<n ; j++ )  printf("%d ", \*(\*(res+i)+j));  printf("\n");  }  productmtx(mtx1,mtx2,res);  for ( int i=0 ; i<n ; i++ ){  for ( int j=0 ; j<n ; j++ )  printf("%d ",\*(\*(res+i)+j));  printf("\n");  }  free\_mem(mtx1);  free\_mem(mtx2);  free\_mem(res);  return 0;  } |