Contents

[Bài thực hành số 1 4](#_Toc149748829)

[**Bài 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. 4](#_Toc149748830)

[**Bài 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}; 6](#_Toc149748831)

[**Bài 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. 8](#_Toc149748832)

[**Bài 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??? 10](#_Toc149748833)

[**Bài 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. 12](#_Toc149748834)

[**Bài 1.6.** Viết hàm đảo ngược một mảng các số nguyên theo hai cách: dùng chỉ số và dùng con trỏ. 14](#_Toc149748835)

[Ví dụ mảng đầu vào là [9, -1, 4, 5, 7] thì kết quả là [7, 5, 4, -1, 9]. 14](#_Toc149748836)

[**Bài 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. 17](#_Toc149748837)

[**Bài 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. 19](#_Toc149748838)

[**Bài 1.9.** Viết chương trình in ra tất cả các dãy con của một dãy cho trước. 27](#_Toc149748839)

[**Bài 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. Yêu cầu sử dụng cấp phát động để cấp phát bộ nhớ cho các ma trận. 37](#_Toc149748840)

[Hình 1. Code bài 1.1 ảnh 1 4](#_Toc149748779)

[Hình 2. Code bài 1.1 ảnh 2 4](file:///C:\Users\Admin\Desktop\Báo%20cáo_IT3040-20215434-Phan%20Hoàng%20Nam-Buổi%201-732826.docx#_Toc149748780)

[Hình 3. Test check bài 1.1 5](#_Toc149748781)

[Hình 4. Code và output bài 1.2 7](#_Toc149748782)

[Hình 5. Code bài 1.3 ảnh 1 8](#_Toc149748783)

[Hình 6. Code bài 1.3 ảnh 2 8](file:///C:\Users\Admin\Desktop\Báo%20cáo_IT3040-20215434-Phan%20Hoàng%20Nam-Buổi%201-732826.docx#_Toc149748784)

[Hình 7. Code ảnh 3 và Test check bài 1.3 9](#_Toc149748785)

[Hình 8. Code bài 1.4 ảnh 1 10](#_Toc149748786)

[Hình 9. Code ảnh 2 và Test check bài 1.4 11](#_Toc149748787)

[Hình 10. Code bài 1.5 ảnh 1 12](#_Toc149748788)

[Hình 11. Code bài 1.5 ảnh 2 12](#_Toc149748789)

[Hình 12. Test check bài 1.5 13](#_Toc149748790)

[Hình 13. Code bài 1.6 ảnh 1 14](#_Toc149748791)

[Hình 14. Code bài 1.6 ảnh 2 15](#_Toc149748792)

[Hình 15. Code ảnh 3 và Test check bài 1.6 15](#_Toc149748793)

[Hình 16. Code bài 1.7 ảnh 1 17](#_Toc149748794)

[Hình 17. Code bài 1.7 ảnh 2 17](#_Toc149748795)

[Hình 18. Code ảnh 3 và Test check bài 1.7 18](#_Toc149748796)

[Hình 19. Code bài 1.8 ảnh 1 20](#_Toc149748797)

[Hình 20. Code bài 1.8 ảnh 2 20](#_Toc149748798)

[Hình 21. Code bài 1.8 ảnh 3 21](#_Toc149748799)

[Hình 22. Code bài 1.8 ảnh 4 21](#_Toc149748800)

[Hình 23. Code bài 1.8 ảnh 5 22](#_Toc149748801)

[Hình 24. Code bài 1.8 ảnh 6 22](#_Toc149748802)

[Hình 25. Code ảnh 7 và Test check bài 1.8 23](#_Toc149748803)

[Hình 26. Code bài 1.9 27](#_Toc149748804)

[Hình 27. Test case 1 bài 1.9 28](#_Toc149748805)

[Hình 28. Test case 2 bài 1.9 28](#_Toc149748806)

[Hình 29. Test case 3 bài 1.9 ảnh 1 29](#_Toc149748807)

[Hình 30. Test case 3 bài 1.9 ảnh 2 29](#_Toc149748808)

[Hình 31. Test case 3 bài 1.9 ảnh 3 30](#_Toc149748809)

[Hình 32. Test case 4 bài 1.9 ảnh 1 31](#_Toc149748810)

[Hình 33. Test case 4 bài 1.9 ảnh 2 31](#_Toc149748811)

[Hình 34. Test case 4 bài 1.9 ảnh 3 32](#_Toc149748812)

[Hình 35. Test case 4 bài 1.9 ảnh 4 32](#_Toc149748813)

[Hình 36. Test case 4 bài 1.9 ảnh 5 33](#_Toc149748814)

[Hình 37. Test case 4 bài 1.9 ảnh 6 33](file:///C:\Users\Admin\Desktop\Báo%20cáo_IT3040-20215434-Phan%20Hoàng%20Nam-Buổi%201-732826.docx#_Toc149748815)

[Hình 38. Test case 4 bài 1.9 ảnh 7 34](file:///C:\Users\Admin\Desktop\Báo%20cáo_IT3040-20215434-Phan%20Hoàng%20Nam-Buổi%201-732826.docx#_Toc149748816)

[Hình 39. Test case 4 bài 1.9 ảnh 8 34](#_Toc149748817)

[Hình 40. Test case 4 bài 1.9 ảnh 9 35](#_Toc149748818)

[Hình 41. Test case 4 bài 1.9 ảnh 10 35](#_Toc149748819)

[Hình 42. Code bài 1.10 ảnh 1 37](#_Toc149748820)

[Hình 43. Code bài 1.10 ảnh 2 38](#_Toc149748821)

[Hình 44. Code bài 1.10 ảnh 3 38](#_Toc149748822)

[Hình 45. Code bài 1.10 ảnh 4 39](#_Toc149748823)

[Hình 46. Test case 1 bài 1.10 39](#_Toc149748824)

[Hình 47. Test case 2 bài 1.10 40](#_Toc149748825)

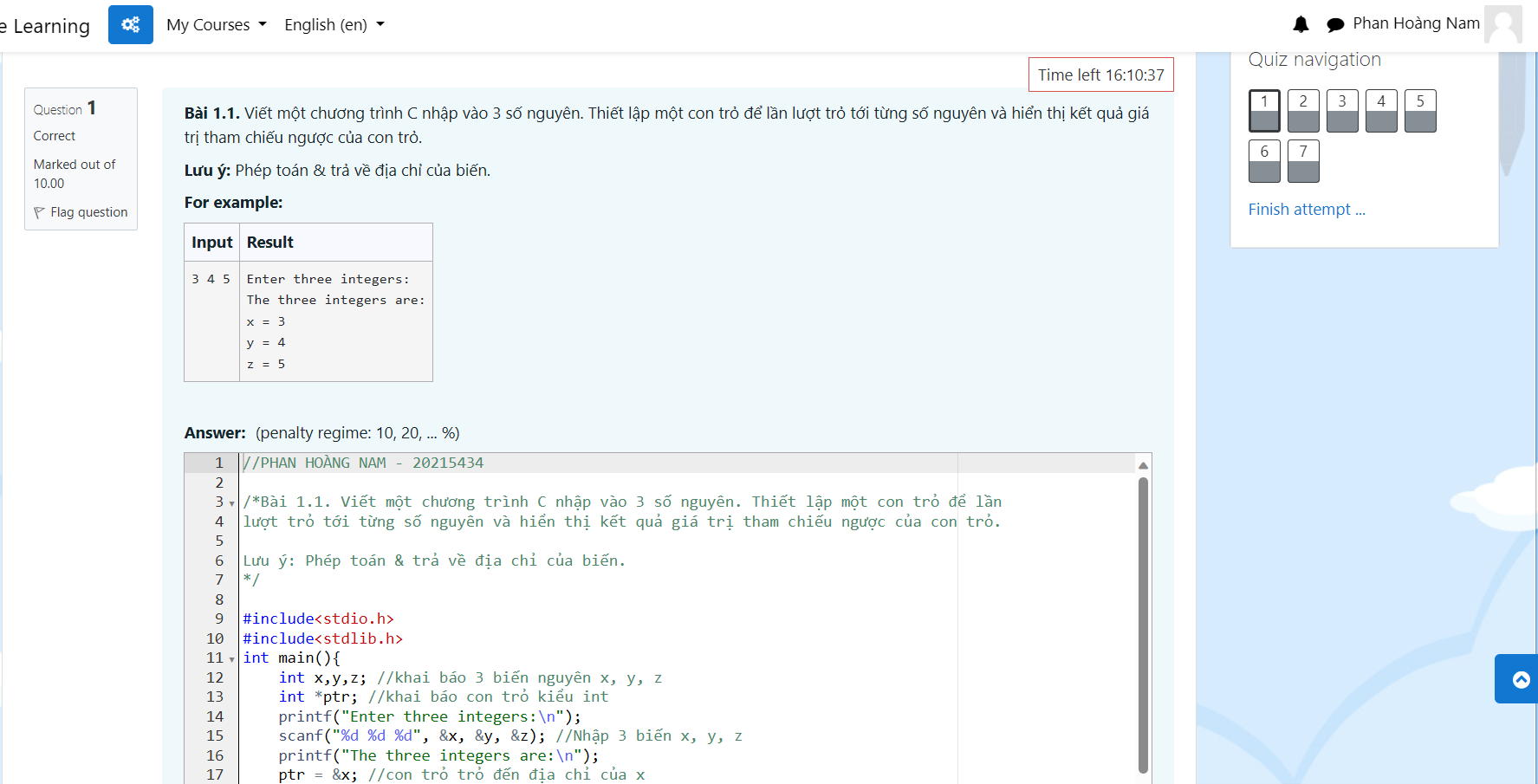
[Hình 48. Test case 3 bài 1.10 40](#_Toc149748826)

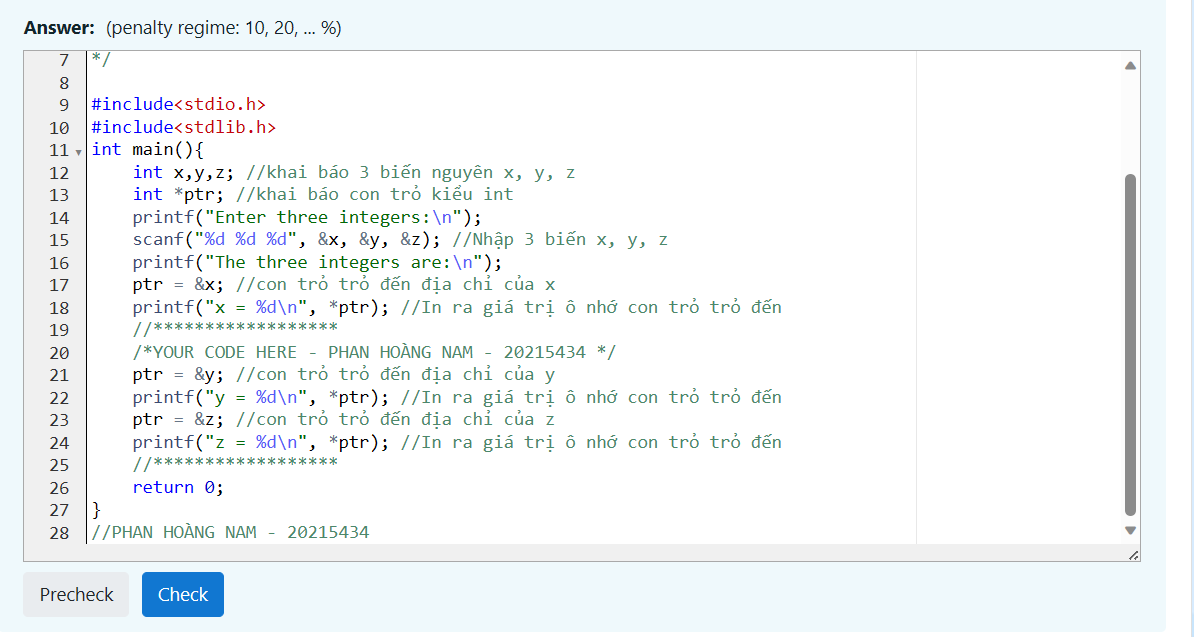
[Hình 49. Test case 4 bài 1.10 ảnh 1 41](#_Toc149748827)

[Hình 50. Test case 4 bài 1.10 ảnh 2 41](#_Toc149748828)

# Bài thực hành số 1

## **Bài 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.



Hình 1. Code bài 1.1 ảnh 1

Hình . Code bài 1.1 ảnh 2



Hình 3. Test check bài 1.1

//PHAN HOÀNG NAM - 20215434

/\*Bài 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.

\*/

#include<stdio.h>

#include<stdlib.h>

int main(){

int x,y,z; //khai báo 3 biến nguyên x, y, z

int \*ptr; //khai báo con trỏ kiểu int

printf("Enter three integers:\n");

scanf("%d %d %d", &x, &y, &z); //Nhập 3 biến x, y, z

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

ptr = &x; //con trỏ trỏ đến địa chỉ của x

printf("x = %d\n", \*ptr); //In ra giá trị ô nhớ con trỏ trỏ đến

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

/\*YOUR CODE HERE - PHAN HOÀNG NAM - 20215434 \*/

ptr = &y; //con trỏ trỏ đến địa chỉ của y

printf("y = %d\n", \*ptr); //In ra giá trị ô nhớ con trỏ trỏ đến

ptr = &z; //con trỏ trỏ đến địa chỉ của z

printf("z = %d\n", \*ptr); //In ra giá trị ô nhớ con trỏ trỏ đến

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

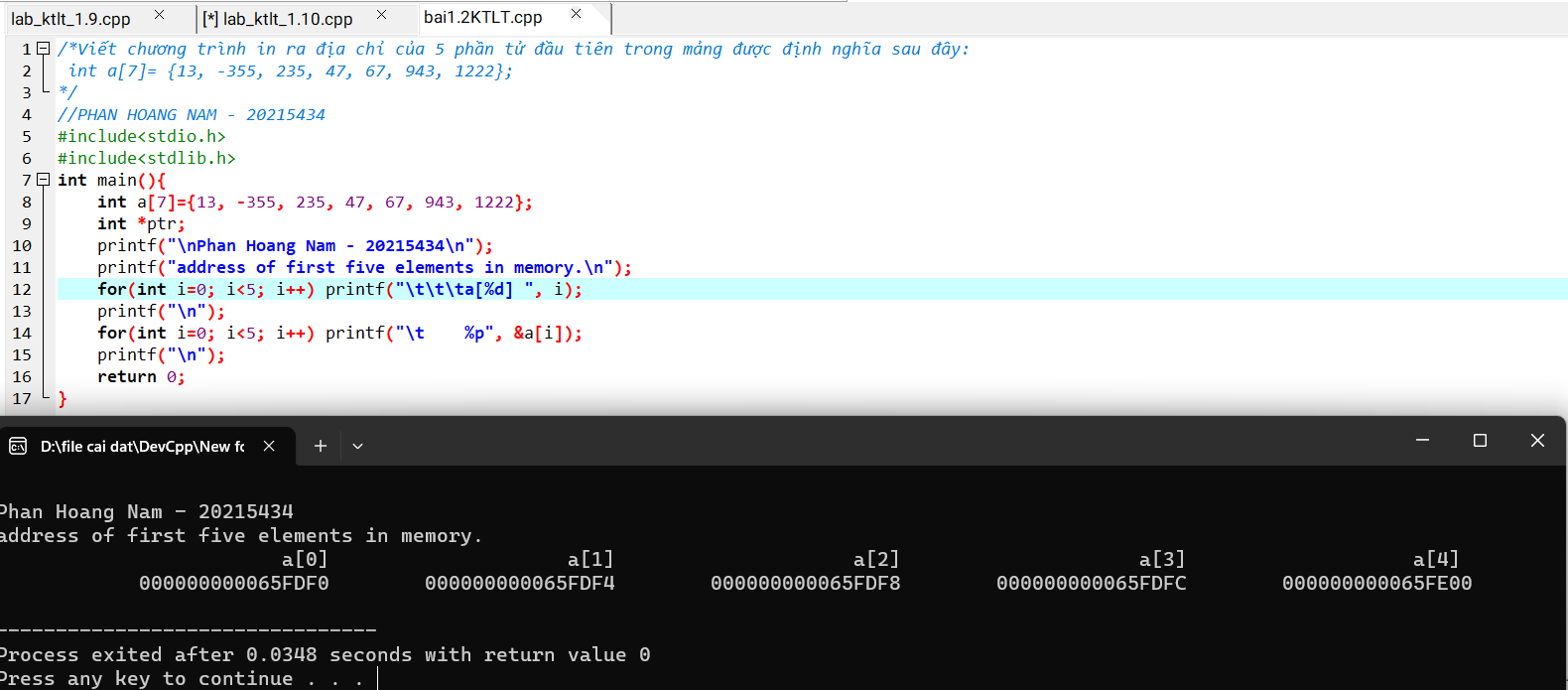
return 0;

}

//PHAN HOÀNG NAM – 20215434

## **Bài 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 4. Code và output bài 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};

\*/

//PHAN HOANG NAM - 20215434

#include<stdio.h>

#include<stdlib.h>

int main(){

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

int \*ptr;

printf("\nPhan Hoang Nam - 20215434\n");

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

for(int i=0; i<5; i++) printf("\t\t\ta[%d] ", i);

printf("\n");

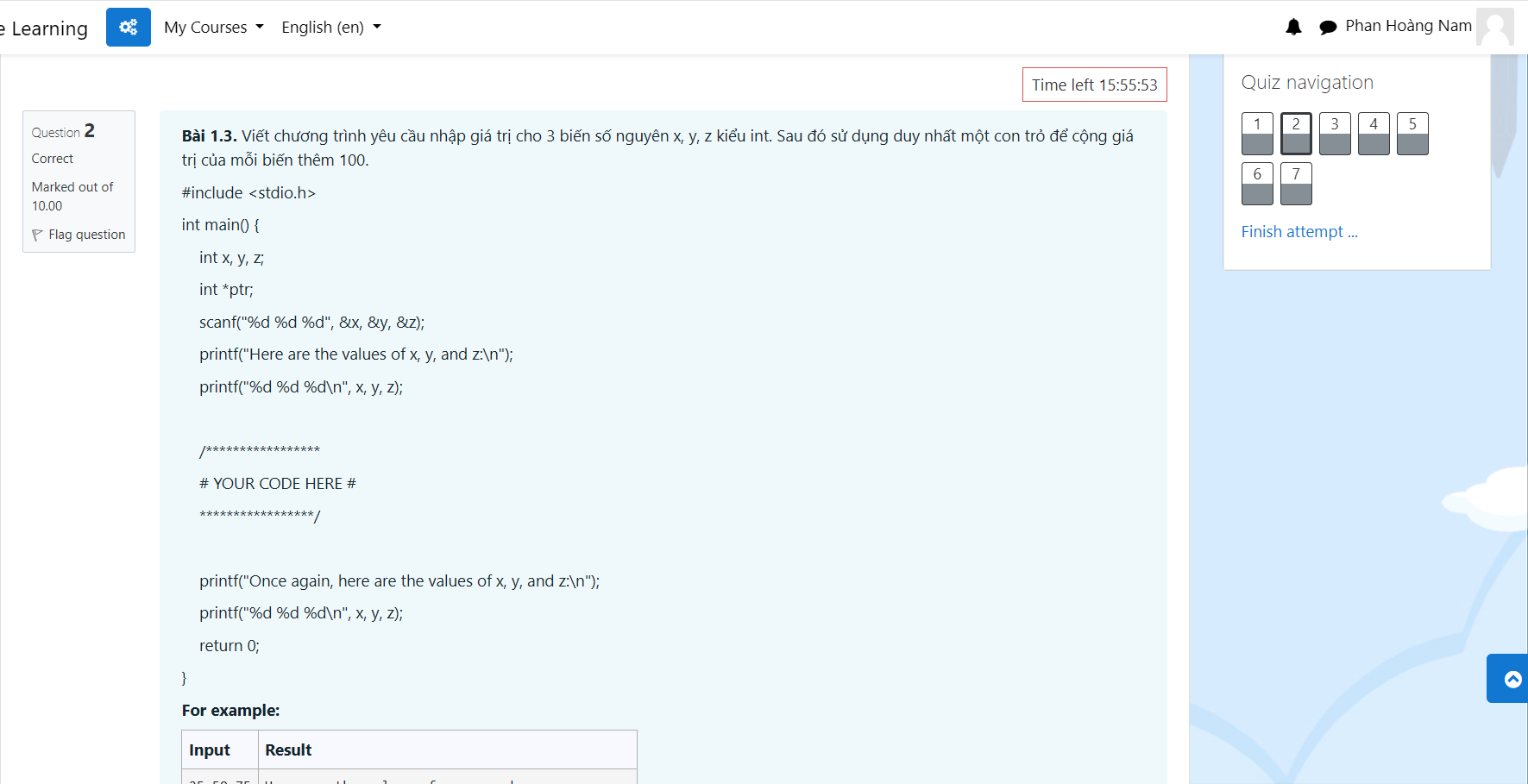
for(int i=0; i<5; i++) printf("\t %p", &a[i]);

printf("\n");

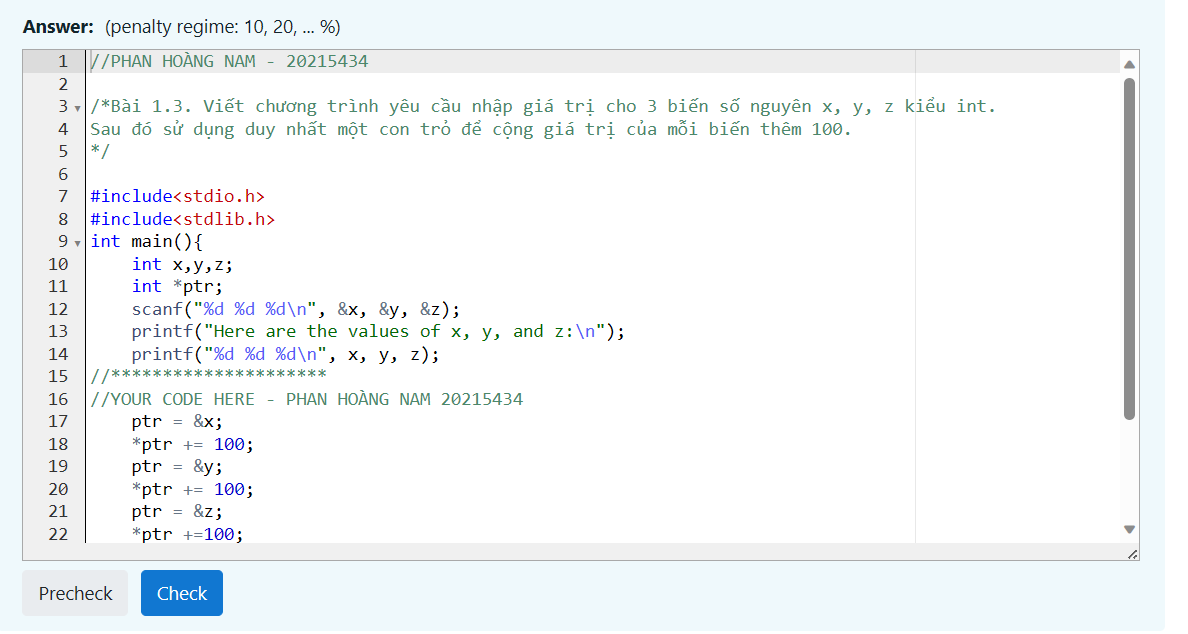
return 0;

}

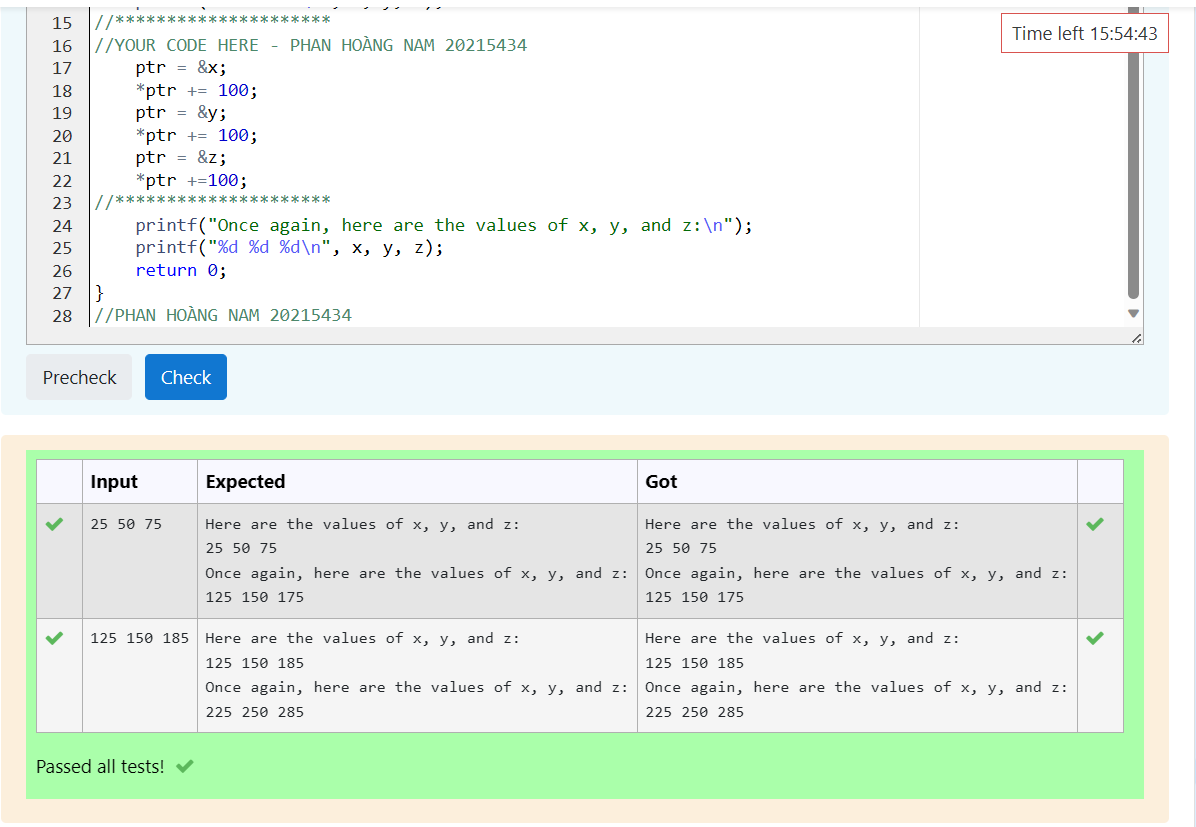
## **Bài 1.3.** Viết chương trình yêu cầu nhập giá trị cho 3 biến số nguyên x, y, z kiểu int. Sau đó sử dụng duy nhất một con trỏ để cộng giá trị của mỗi biến thêm 100.



Hình 5. Code bài 1.3 ảnh 1



Hình . Code bài 1.3 ảnh 2



Hình 7. Code ảnh 3 và Test check bài 1.3

//PHAN HOÀNG NAM - 20215434

/\*Bài 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.

\*/

#include<stdio.h>

#include<stdlib.h>

int main(){

int x,y,z;

int \*ptr;

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

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

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

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

//YOUR CODE HERE - PHAN HOÀNG NAM 20215434

ptr = &x;

\*ptr += 100;

ptr = &y;

\*ptr += 100;

ptr = &z;

\*ptr +=100;

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

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

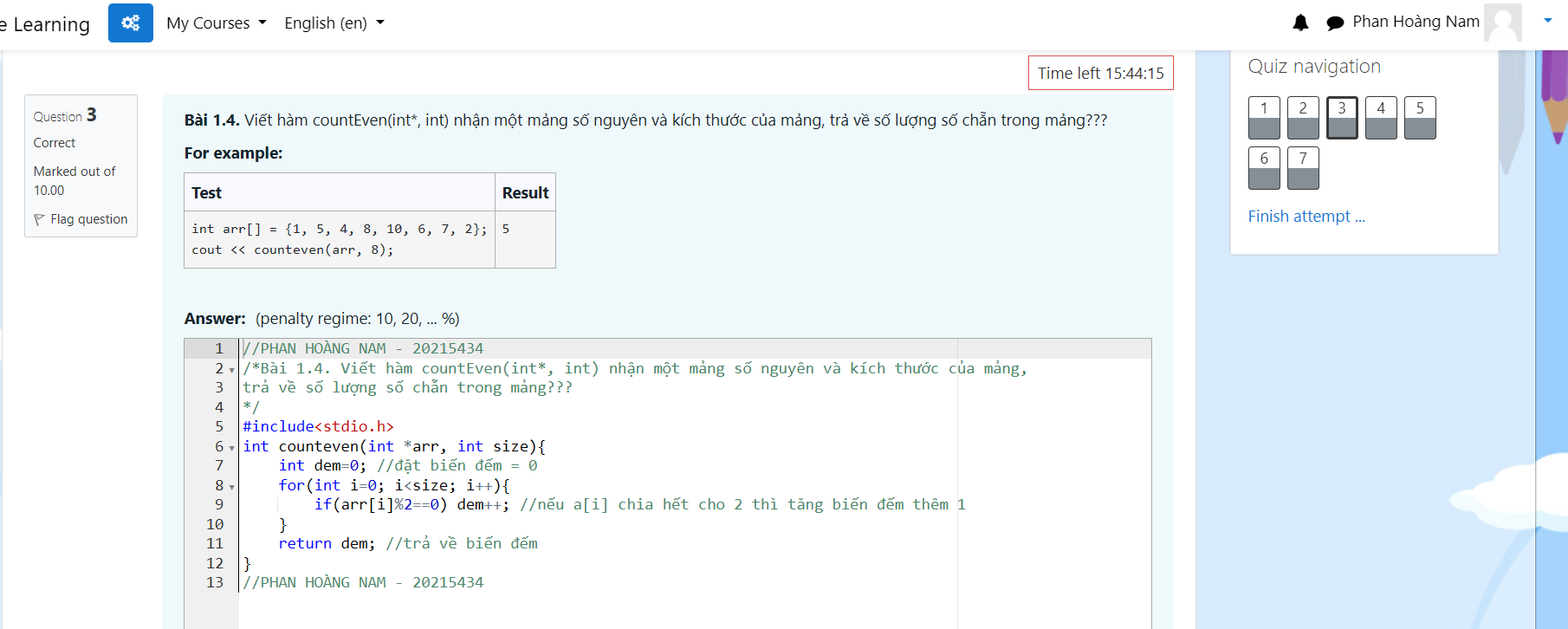
printf("%d %d %d\n", x, y, z);

return 0;

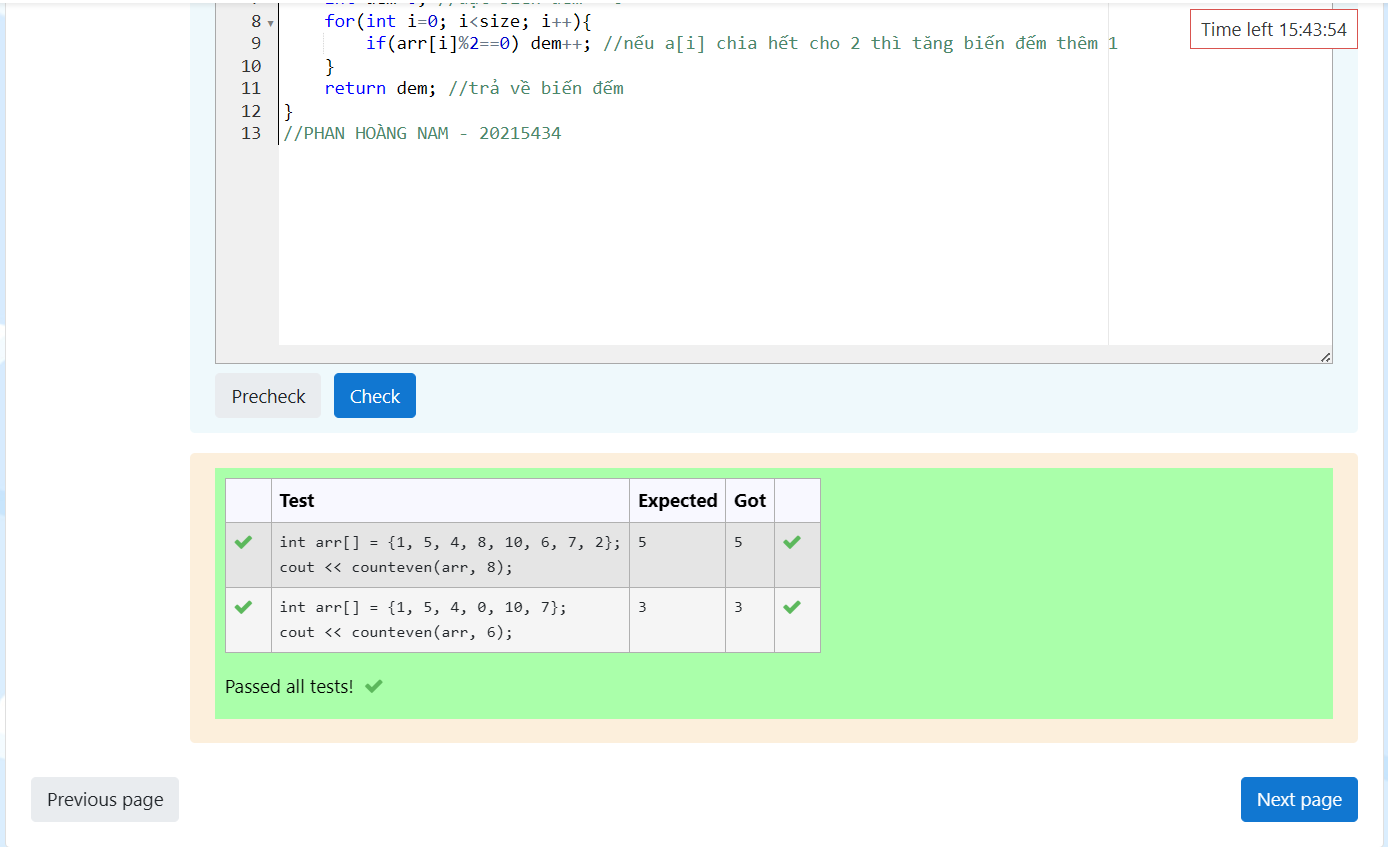
}

//PHAN HOÀNG NAM 20215434

## **Bài 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 8. Code bài 1.4 ảnh 1



Hình 9. Code ảnh 2 và Test check bài 1.4

//PHAN HOÀNG NAM - 20215434

/\*Bài 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???

\*/

#include<stdio.h>

int counteven(int \*arr, int size){

int dem=0; //đặt biến đếm = 0

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

if(arr[i]%2==0) dem++; //nếu a[i] chia hết cho 2 thì tăng biến đếm thêm 1

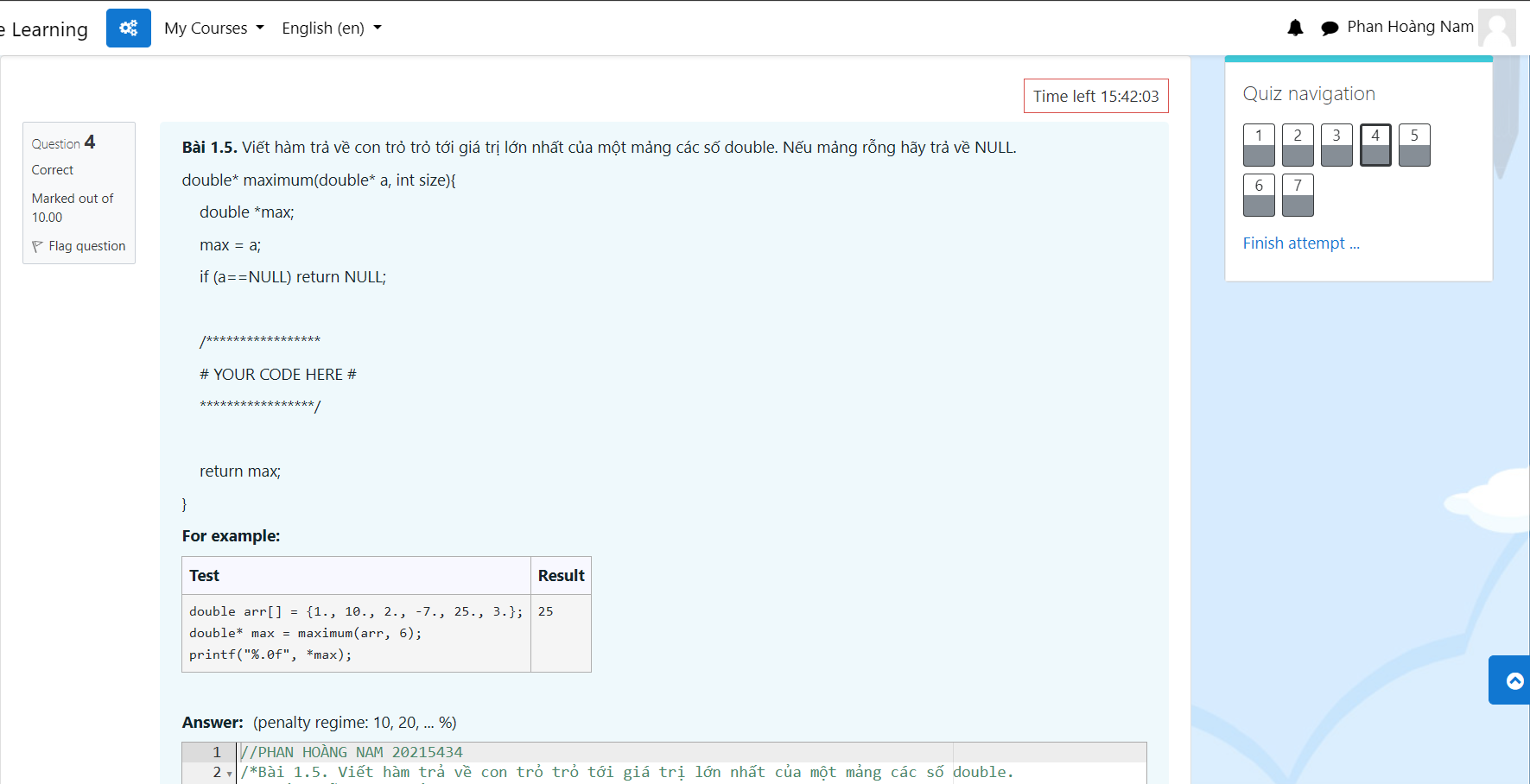
}

return dem; //trả về biến đếm

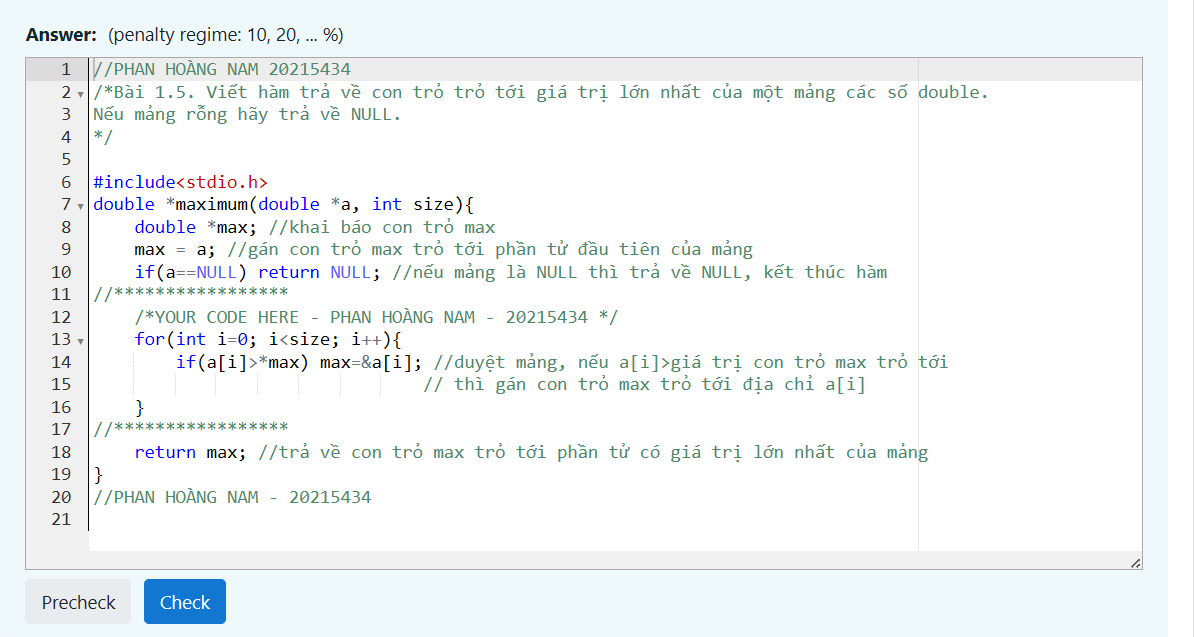
}

//PHAN HOÀNG NAM – 20215434

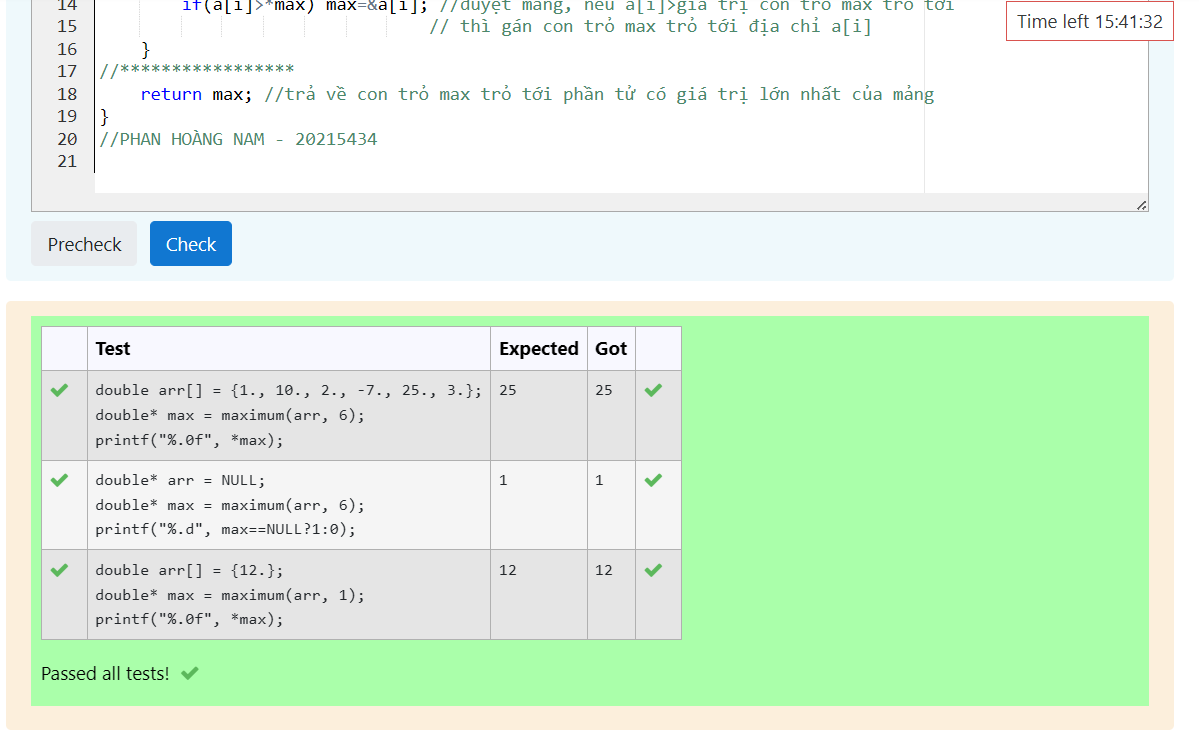
## **Bài 1.5.** Viết hàm trả về con trỏ trỏ tới giá trị lớn nhất của một mảng các số double. Nếu mảng rỗng hãy trả về NULL.



Hình 10. Code bài 1.5 ảnh 1



Hình 11. Code bài 1.5 ảnh 2



Hình 12. Test check bài 1.5

//PHAN HOÀNG NAM 20215434

/\*Bài 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.

\*/

#include<stdio.h>

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

double \*max; //khai báo con trỏ max

max = a; //gán con trỏ max trỏ tới phần tử đầu tiên của mảng

if(a==NULL) return NULL; //nếu mảng là NULL thì trả về NULL, kết thúc hàm

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

/\*YOUR CODE HERE - PHAN HOÀNG NAM - 20215434 \*/

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

if(a[i]>\*max) max=&a[i]; //duyệt mảng, nếu a[i]>giá trị con trỏ max trỏ tới

// thì gán con trỏ max trỏ tới địa chỉ a[i]

}

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

return max; //trả về con trỏ max trỏ tới phần tử có giá trị lớn nhất của mảng

}

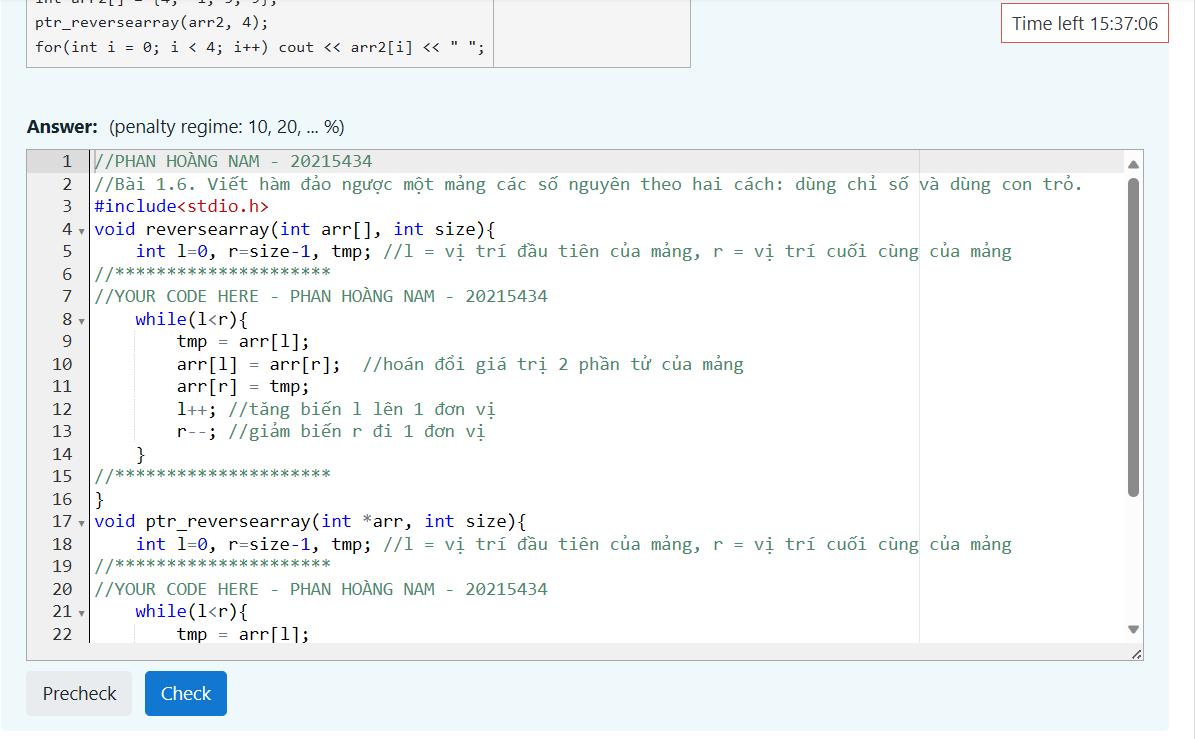
//PHAN HOÀNG NAM – 20215434

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

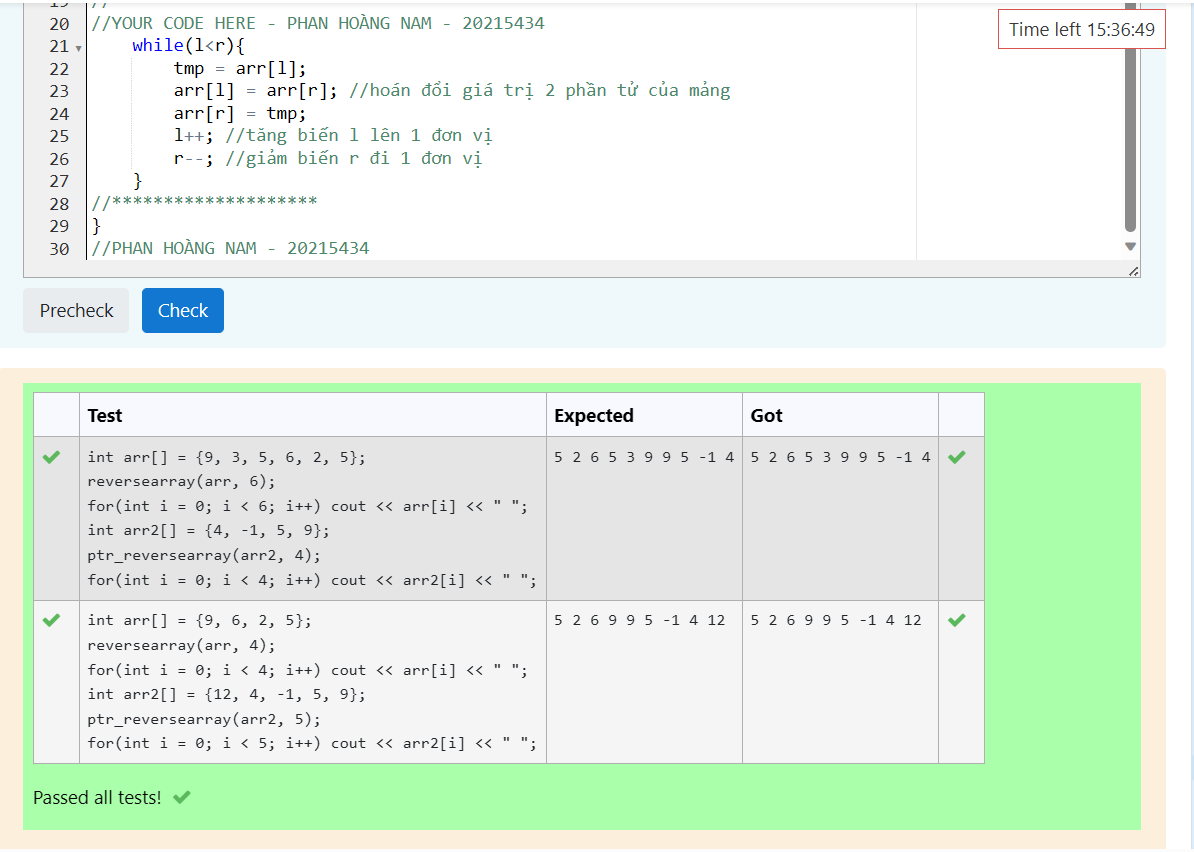
## Ví dụ mảng đầu vào là [9, -1, 4, 5, 7] thì kết quả là [7, 5, 4, -1, 9].



Hình 13. Code bài 1.6 ảnh 1



Hình 14. Code bài 1.6 ảnh 2



Hình 15. Code ảnh 3 và Test check bài 1.6

//PHAN HOÀNG NAM - 20215434

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

#include<stdio.h>

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

int l=0, r=size-1, tmp; //l = vị trí đầu tiên của mảng, r = vị trí cuối cùng của mảng

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

while(l<r){

tmp = arr[l];

arr[l] = arr[r]; //hoán đổi giá trị 2 phần tử của mảng

arr[r] = tmp;

l++; //tăng biến l lên 1 đơn vị

r--; //giảm biến r đi 1 đơn vị

}

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

}

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

int l=0, r=size-1, tmp; //l = vị trí đầu tiên của mảng, r = vị trí cuối cùng của mảng

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

while(l<r){

tmp = arr[l];

arr[l] = arr[r]; //hoán đổi giá trị 2 phần tử của mảng

arr[r] = tmp;

l++; //tăng biến l lên 1 đơn vị

r--; //giảm biến r đi 1 đơn vị

}

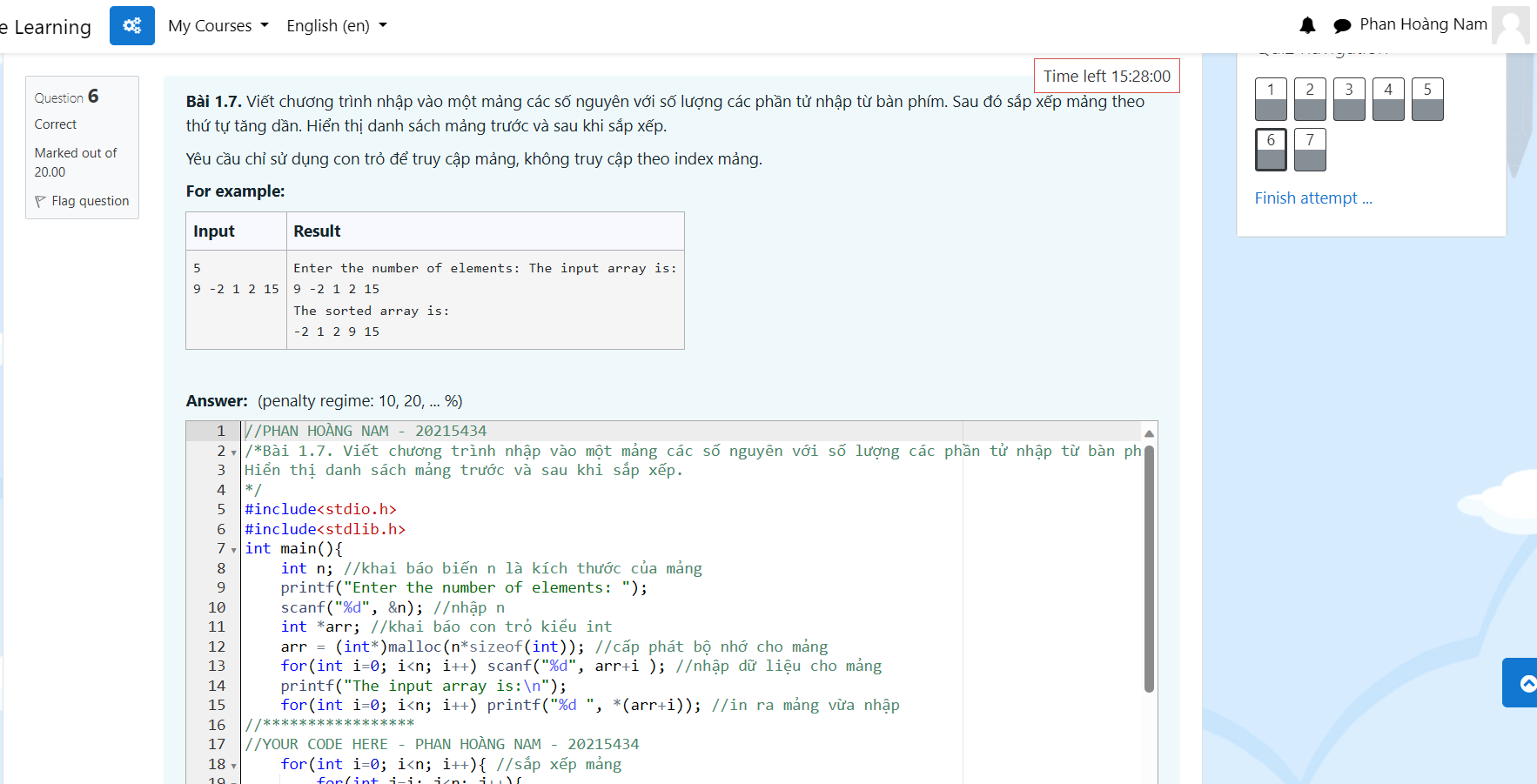
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

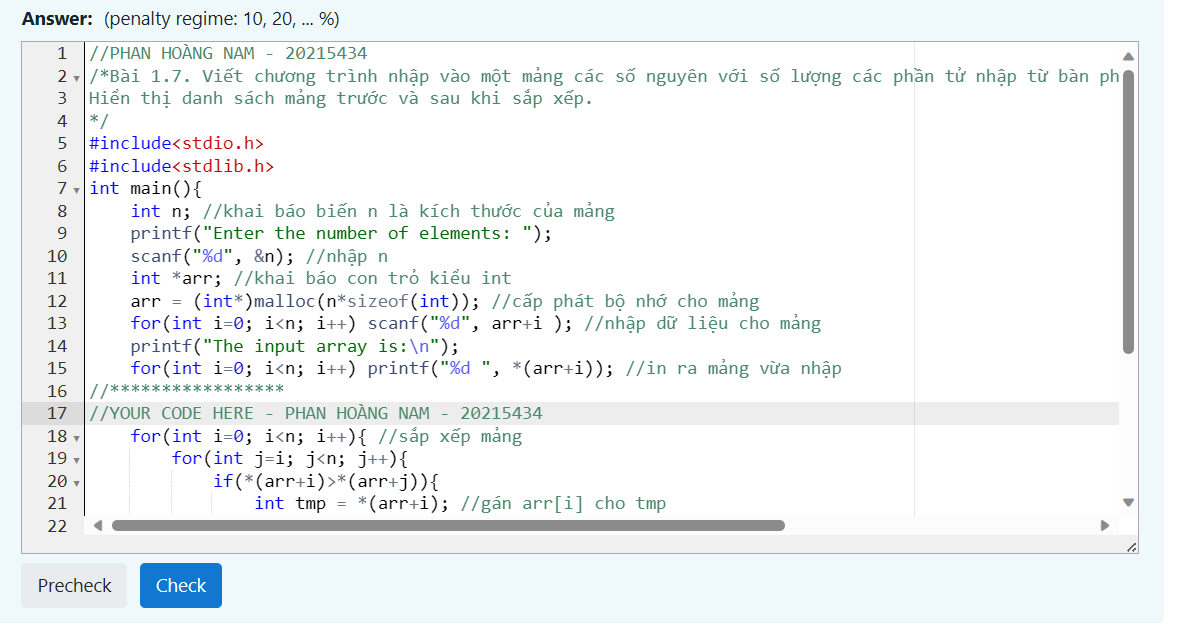
//PHAN HOÀNG NAM – 20215434

## **Bài 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.

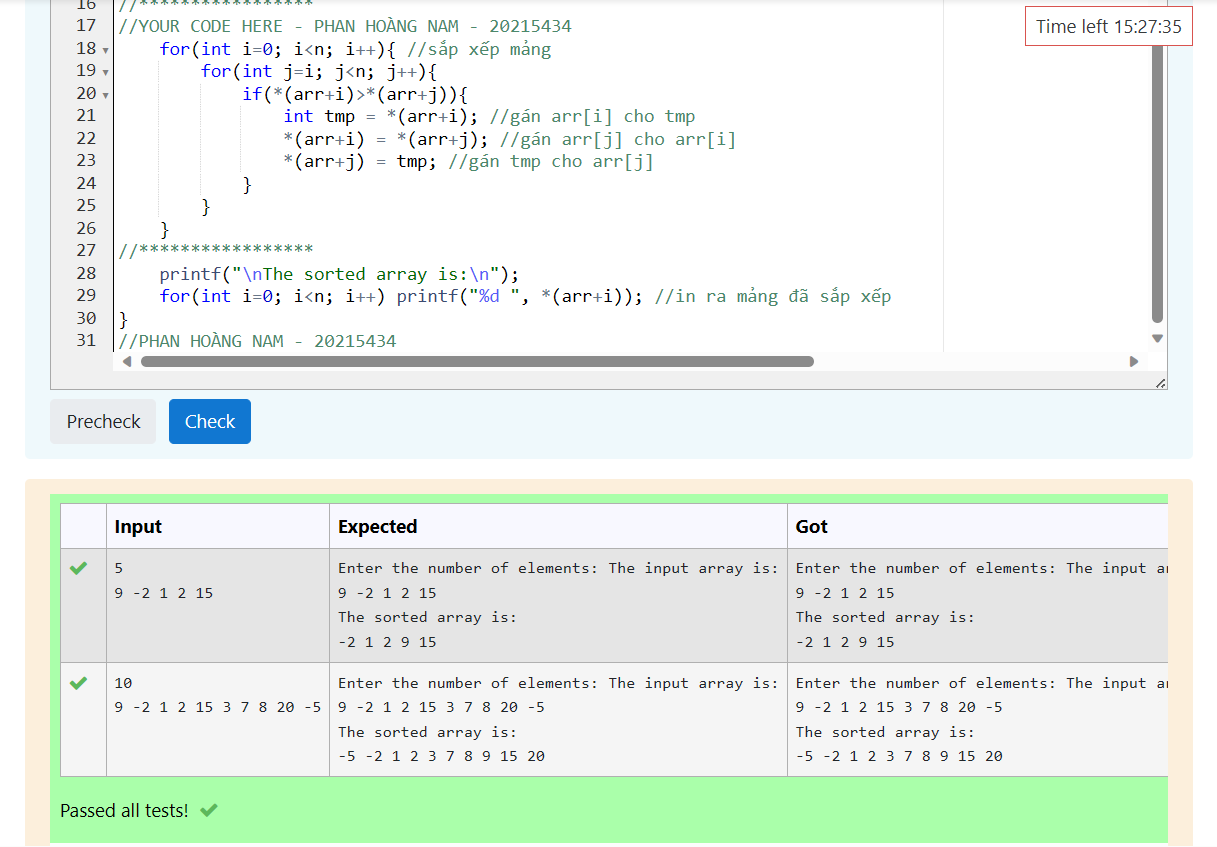
Yêu cầu chỉ sử dụng con trỏ để truy cập mảng, không truy cập theo index mảng.



Hình 16. Code bài 1.7 ảnh 1



Hình 17. Code bài 1.7 ảnh 2



Hình 18. Code ảnh 3 và Test check bài 1.7

//PHAN HOÀNG NAM - 20215434

/\*Bài 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.

\*/

#include<stdio.h>

#include<stdlib.h>

int main(){

int n; //khai báo biến n là kích thước của mảng

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

scanf("%d", &n); //nhập n

int \*arr; //khai báo con trỏ kiểu int

arr = (int\*)malloc(n\*sizeof(int)); //cấp phát bộ nhớ cho mảng

for(int i=0; i<n; i++) scanf("%d", arr+i ); //nhập dữ liệu cho mảng

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

for(int i=0; i<n; i++) printf("%d ", \*(arr+i)); //in ra mảng vừa nhập

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

for(int i=0; i<n; i++){ //sắp xếp mảng

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

if(\*(arr+i)>\*(arr+j)){

int tmp = \*(arr+i); //gán arr[i] cho tmp

\*(arr+i) = \*(arr+j); //gán arr[j] cho arr[i]

\*(arr+j) = tmp; //gán tmp cho arr[j]

}

}

}

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

printf("\nThe sorted array is:\n");

for(int i=0; i<n; i++) printf("%d ", \*(arr+i)); //in ra mảng đã sắp xếp

}

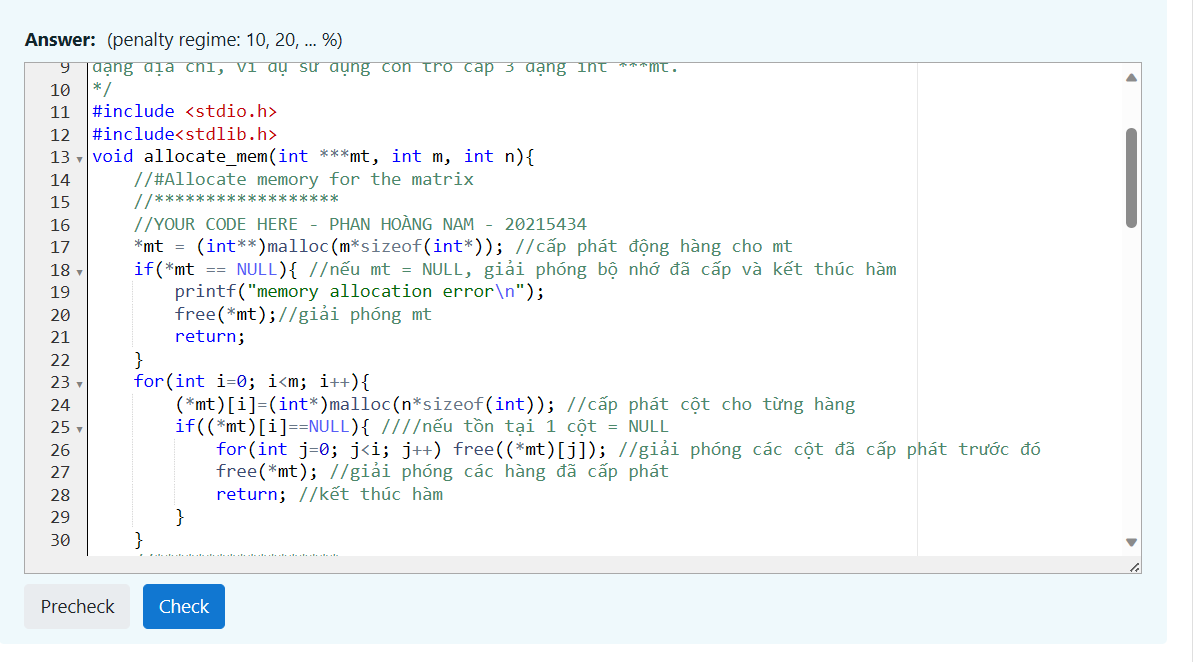
//PHAN HOÀNG NAM – 20215434

## **Bài 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.

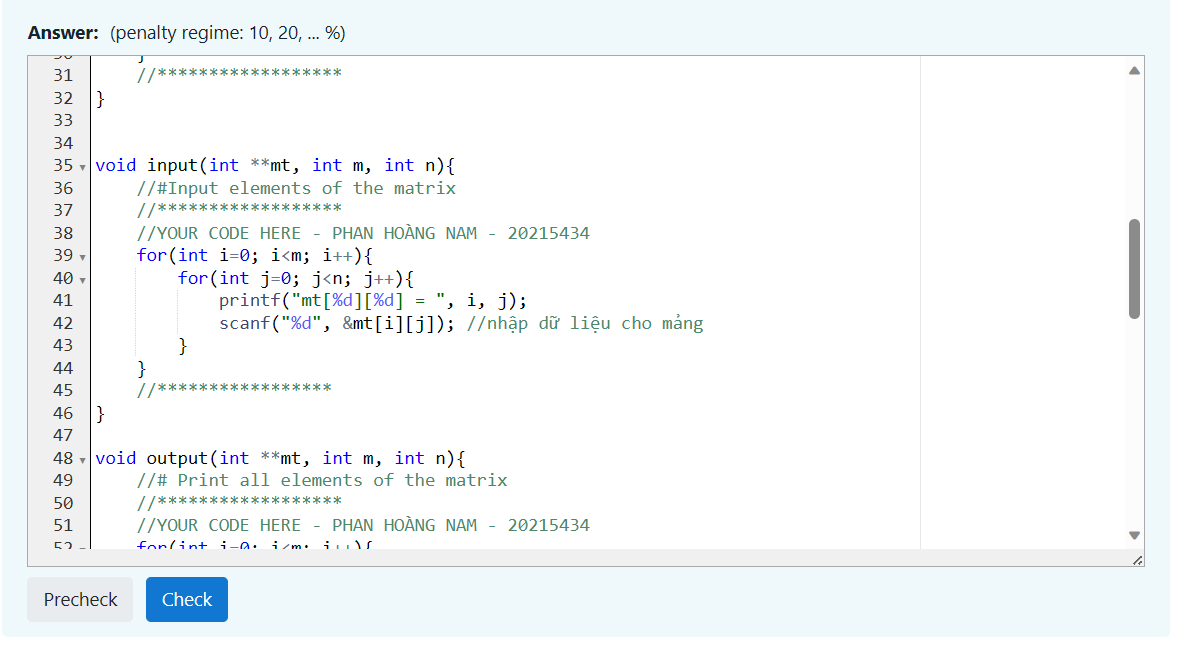
**Lưu ý:** Khi viết hàm cấp phát bộ nhớ cho một ma trận hai chiều biểu diễn bởi con trỏ int \*\*mt, nếu ta truyền con trỏ theo kiểu địa chỉ void allocate\_mem(int \*\*mt, int m, int n) sẽ dẫn tới việc cấp phát bộ nhớ cho một bản sao của con trỏ \*\*mt. Do đó, sau khi gọi hàm thì con trỏ \*\*mt gốc vẫn không được cấp phát bộ nhớ. Để cấp phát thành công cần truyền con trỏ theo dạng địa chỉ, ví dụ sử dụng con trỏ cấp 3 dạng int \*\*\*mt.



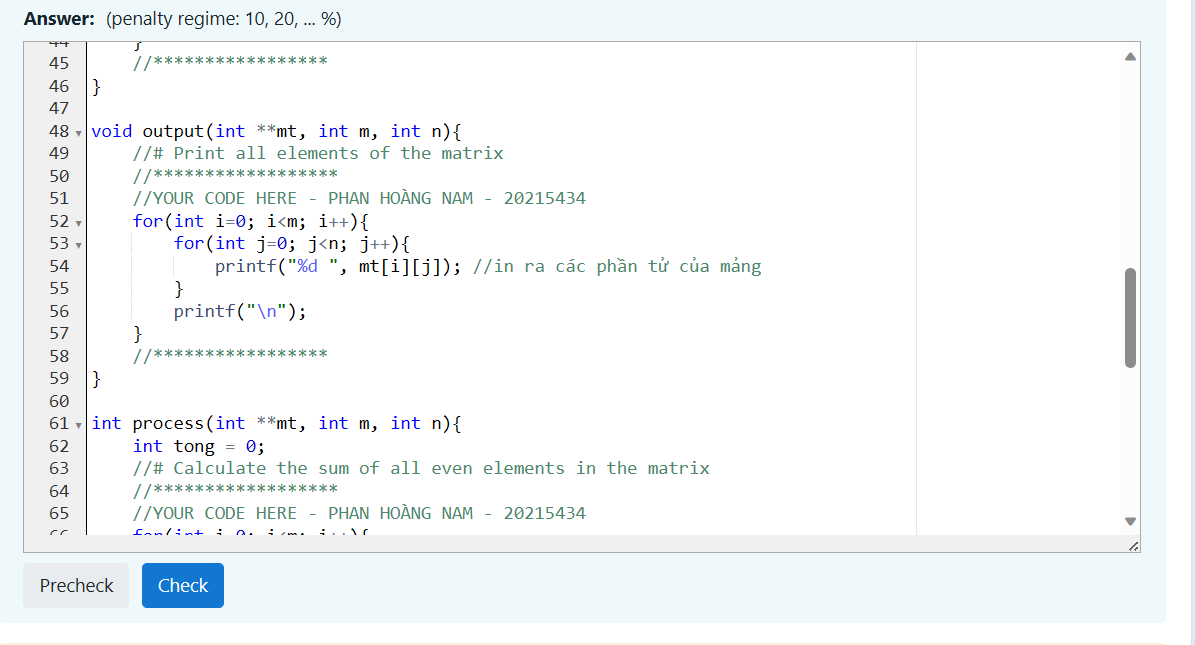
Hình 19. Code bài 1.8 ảnh 1



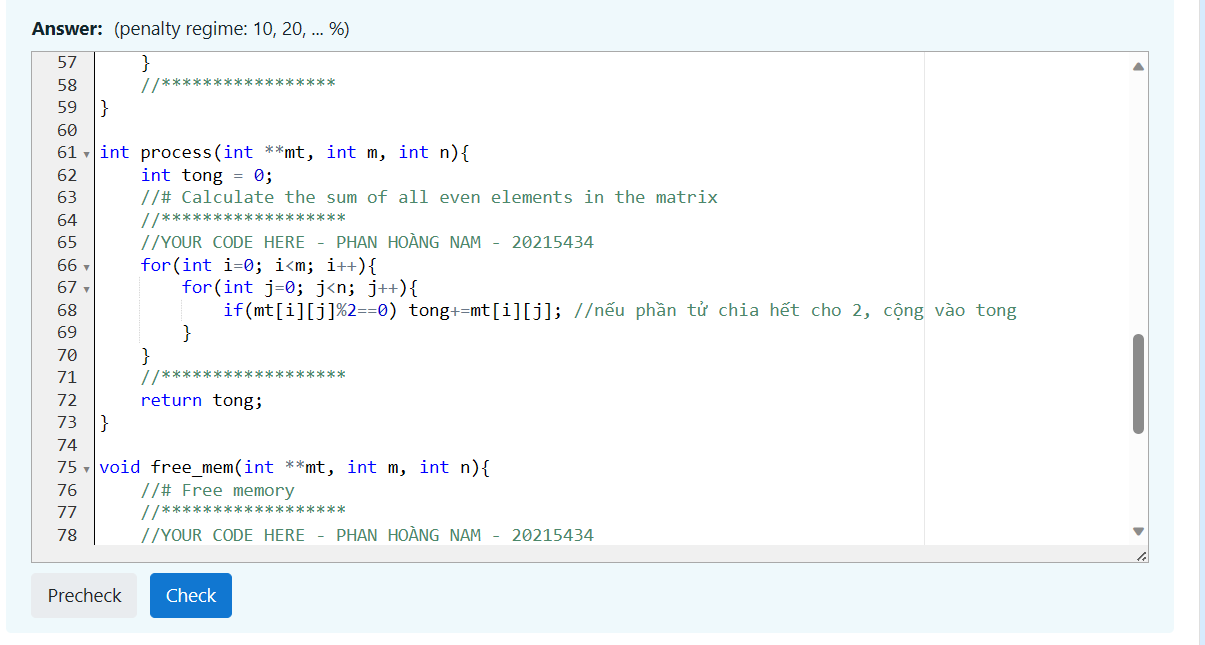
Hình 20. Code bài 1.8 ảnh 2



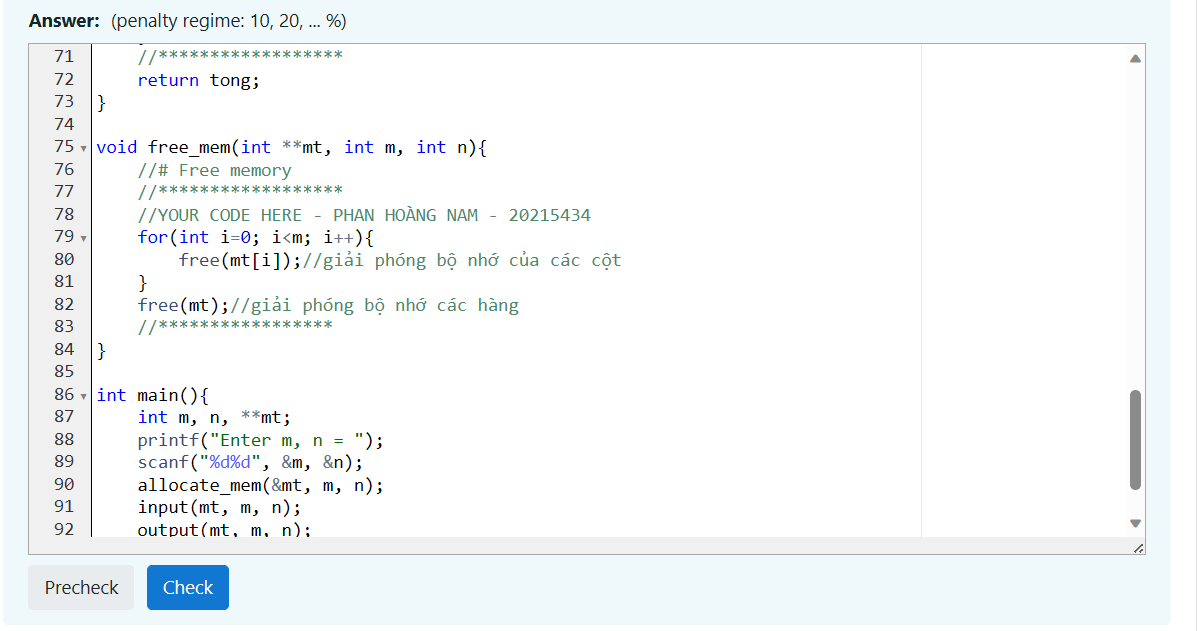
Hình 21. Code bài 1.8 ảnh 3



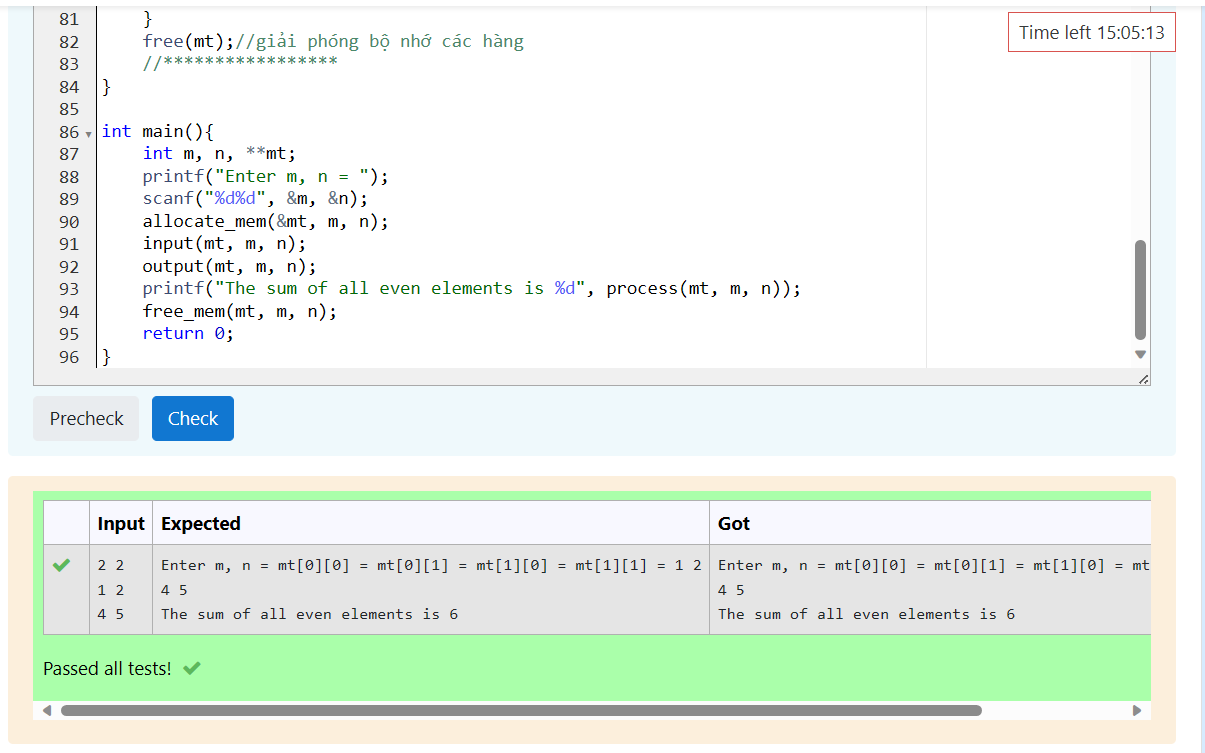
Hình 22. Code bài 1.8 ảnh 4



Hình 23. Code bài 1.8 ảnh 5



Hình 24. Code bài 1.8 ảnh 6



Hình 25. Code ảnh 7 và Test check bài 1.8

//PHAN HOÀNG NAM - 20215434

/\*Bài 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.

Lưu ý: Khi viết hàm cấp phát bộ nhớ cho một ma trận hai chiều biểu diễn bởi con trỏ int \*\*mt,

nếu ta truyền con trỏ theo kiểu địa chỉ void allocate\_mem(int \*\*mt, int m, int n)

sẽ dẫn tới việc cấp phát bộ nhớ cho một bản sao của con trỏ \*\*mt. Do đó, sau khi gọi hàm thì

con trỏ \*\*mt gốc vẫn không được cấp phát bộ nhớ. Để cấp phát thành công cần truyền con trỏ theo

dạng địa chỉ, ví dụ sử dụng con trỏ cấp 3 dạng int \*\*\*mt.

\*/

#include <stdio.h>

#include<stdlib.h>

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

//#Allocate memory for the matrix

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

\*mt = (int\*\*)malloc(m\*sizeof(int\*)); //cấp phát động hàng cho mt

if(\*mt == NULL){ //nếu mt = NULL, giải phóng bộ nhớ đã cấp và kết thúc hàm

printf("memory allocation error\n");

free(\*mt);//giải phóng mt

return;

}

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

(\*mt)[i]=(int\*)malloc(n\*sizeof(int)); //cấp phát cột cho từng hàng

if((\*mt)[i]==NULL){ ////nếu tồn tại 1 cột = NULL

for(int j=0; j<i; j++) free((\*mt)[j]); //giải phóng các cột đã cấp phát trước đó

free(\*mt); //giải phóng các hàng đã cấp phát

return; //kết thúc hàm

}

}

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

}

void input(int \*\*mt, int m, int n){

//#Input elements of the matrix

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

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]); //nhập dữ liệu cho mảng

}

}

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

}

void output(int \*\*mt, int m, int n){

//# Print all elements of the matrix

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

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

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

printf("%d ", mt[i][j]); //in ra các phần tử của mảng

}

printf("\n");

}

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

}

int process(int \*\*mt, int m, int n){

int tong = 0;

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

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

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

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

if(mt[i][j]%2==0) tong+=mt[i][j]; //nếu phần tử chia hết cho 2, cộng vào tong

}

}

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

return tong;

}

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

//# Free memory

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

//YOUR CODE HERE - PHAN HOÀNG NAM - 20215434

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

free(mt[i]);//giải phóng bộ nhớ của các cột

}

free(mt);//giải phóng bộ nhớ các hàng

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

}

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);

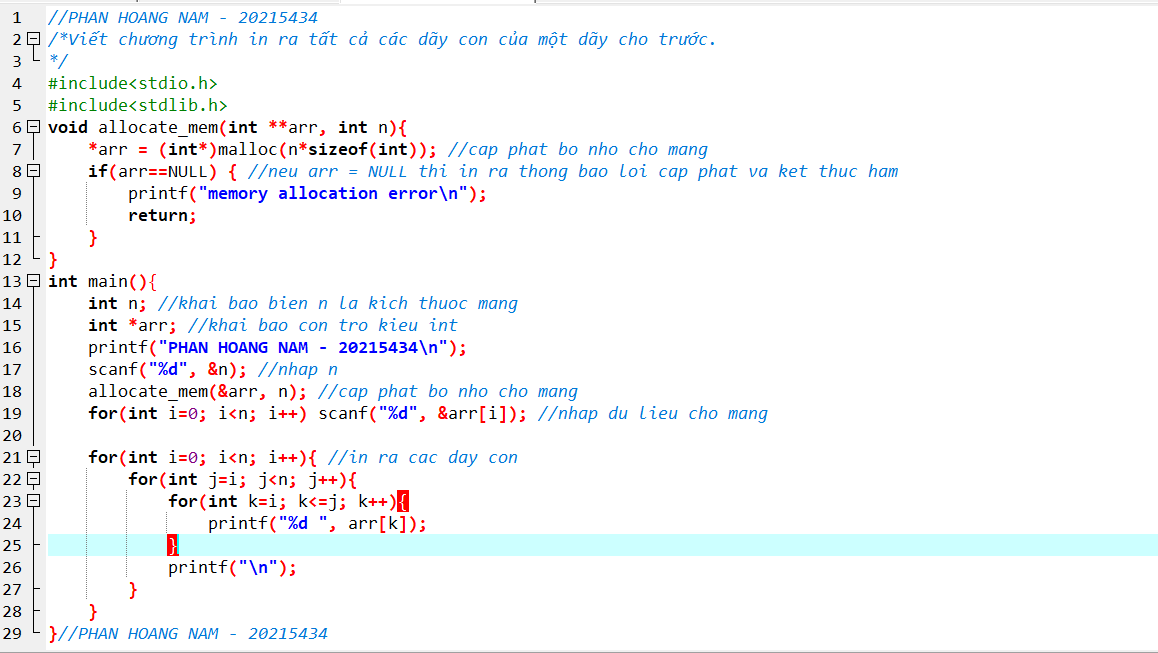
printf("The sum of all even elements is %d", process(mt, m, n));

free\_mem(mt, m, n);

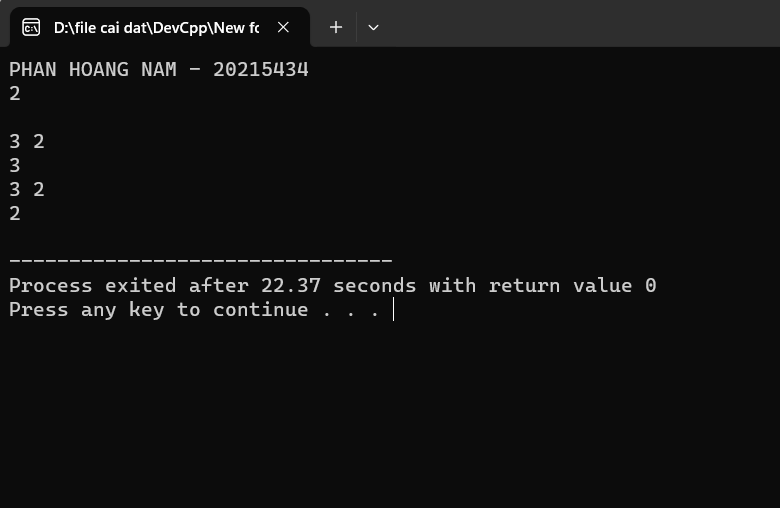
return 0;

}

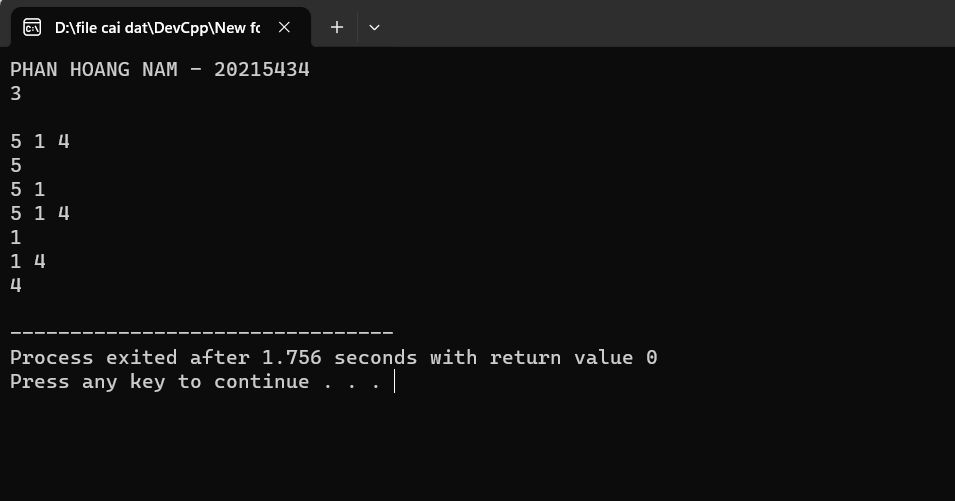
## **Bài 1.9.** Viết chương trình in ra tất cả các dãy con của một dãy cho trước.



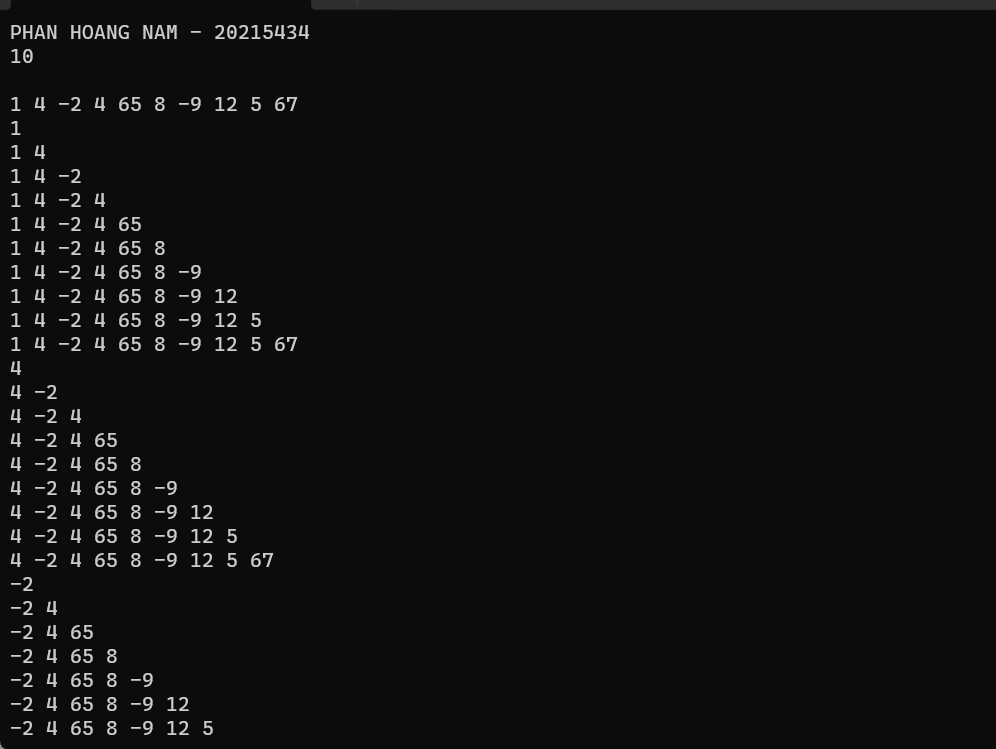
Hình 26. Code bài 1.9



Hình 27. Test case 1 bài 1.9



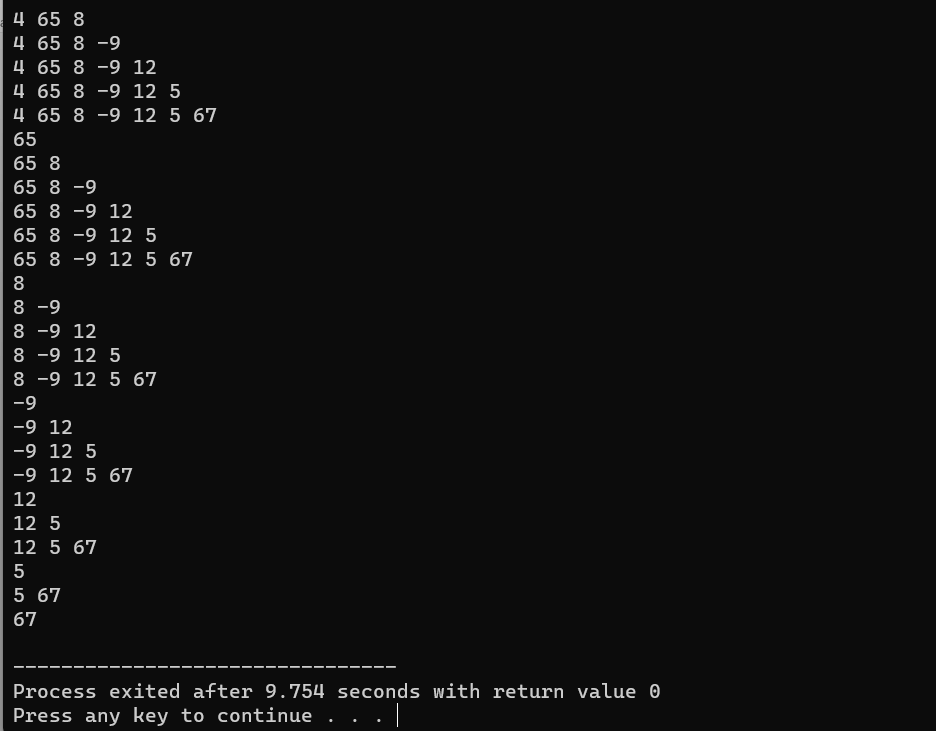
Hình 28. Test case 2 bài 1.9



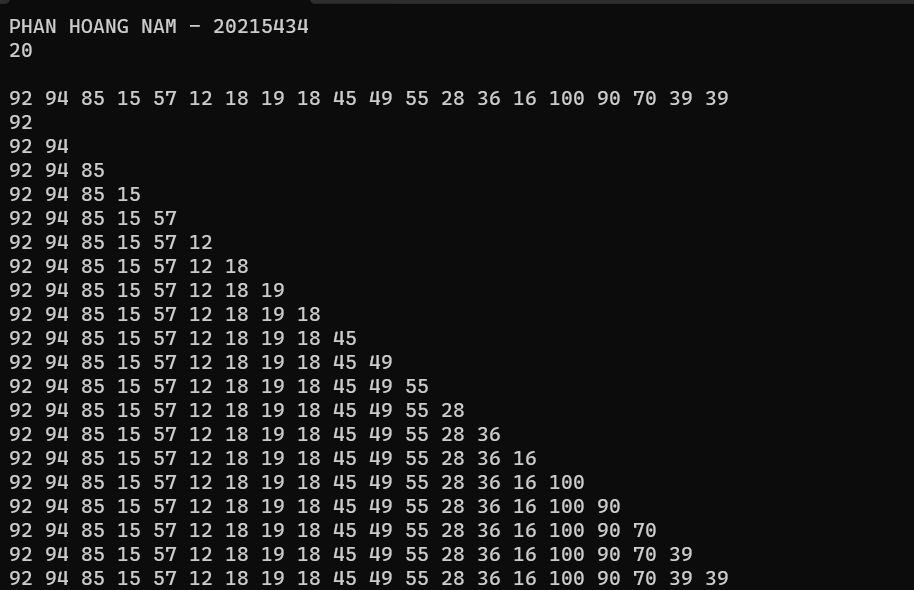
Hình 29. Test case 3 bài 1.9 ảnh 1



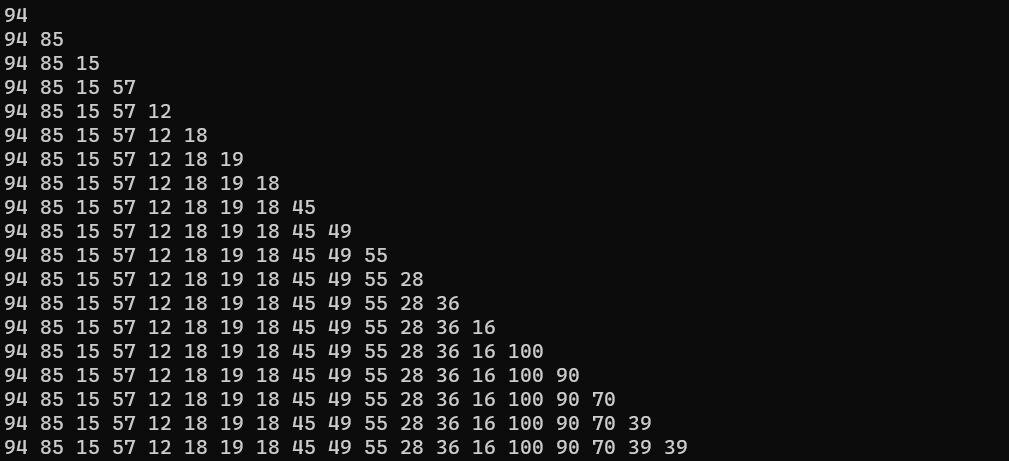
Hình 30. Test case 3 bài 1.9 ảnh 2



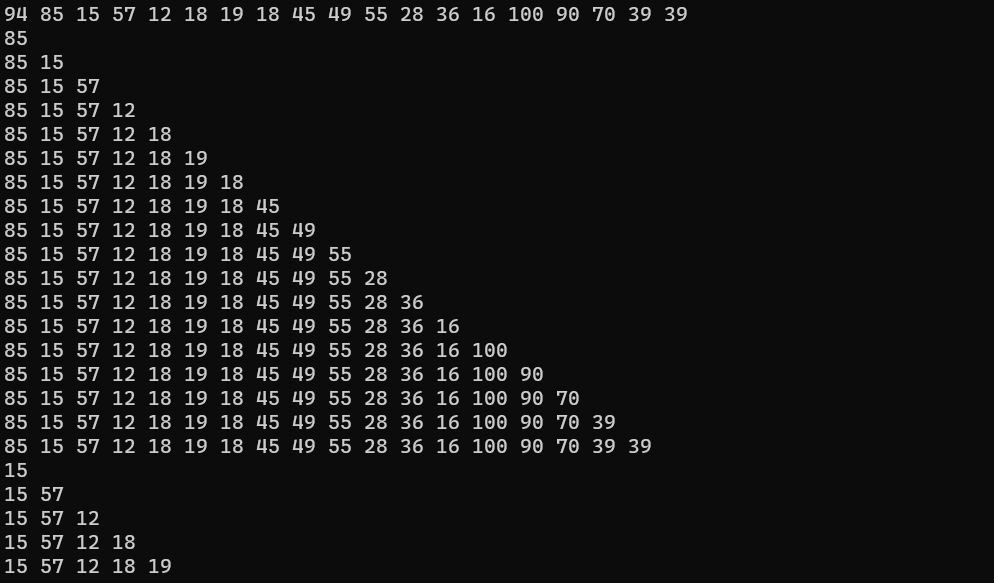
Hình 31. Test case 3 bài 1.9 ảnh 3



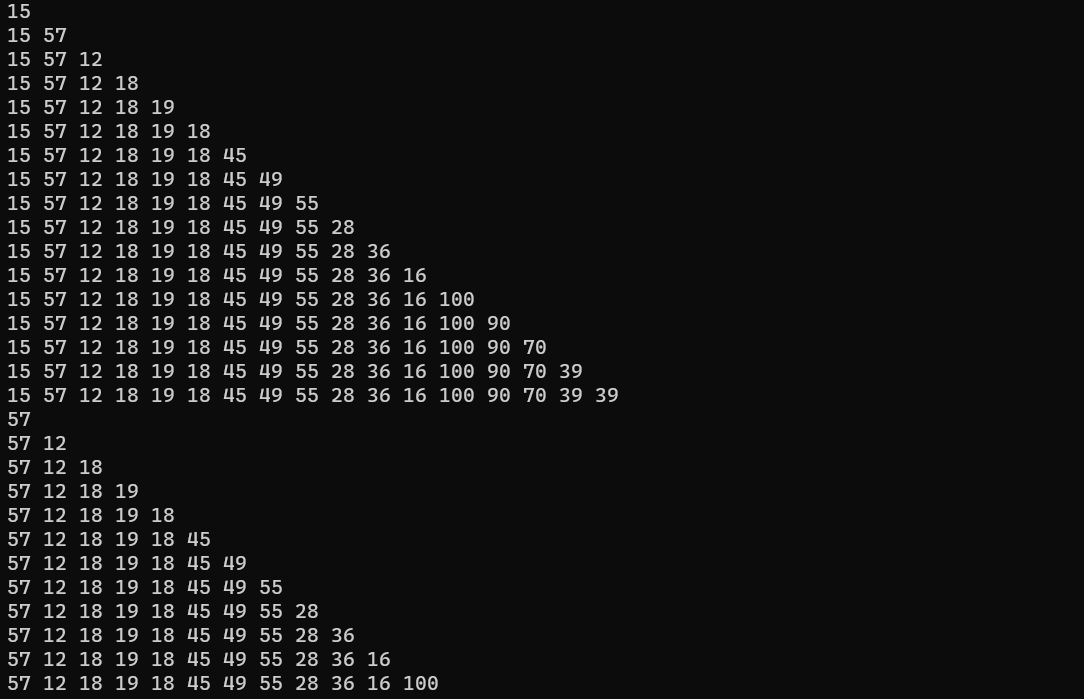
Hình 32. Test case 4 bài 1.9 ảnh 1



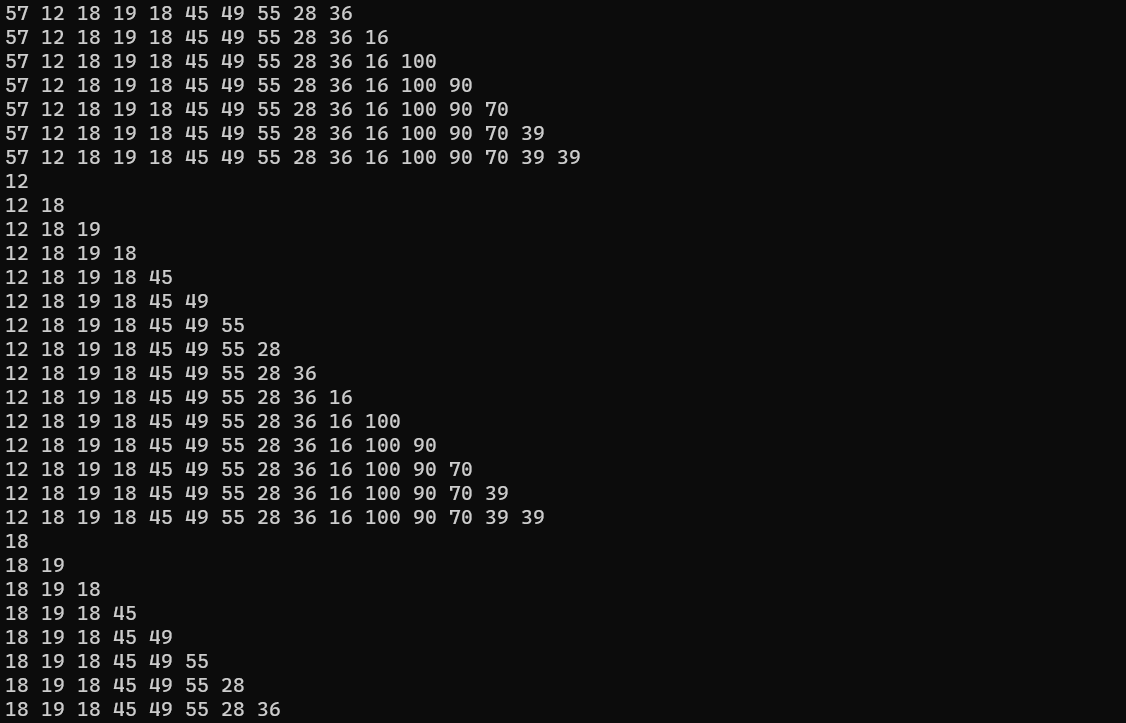
Hình 33. Test case 4 bài 1.9 ảnh 2



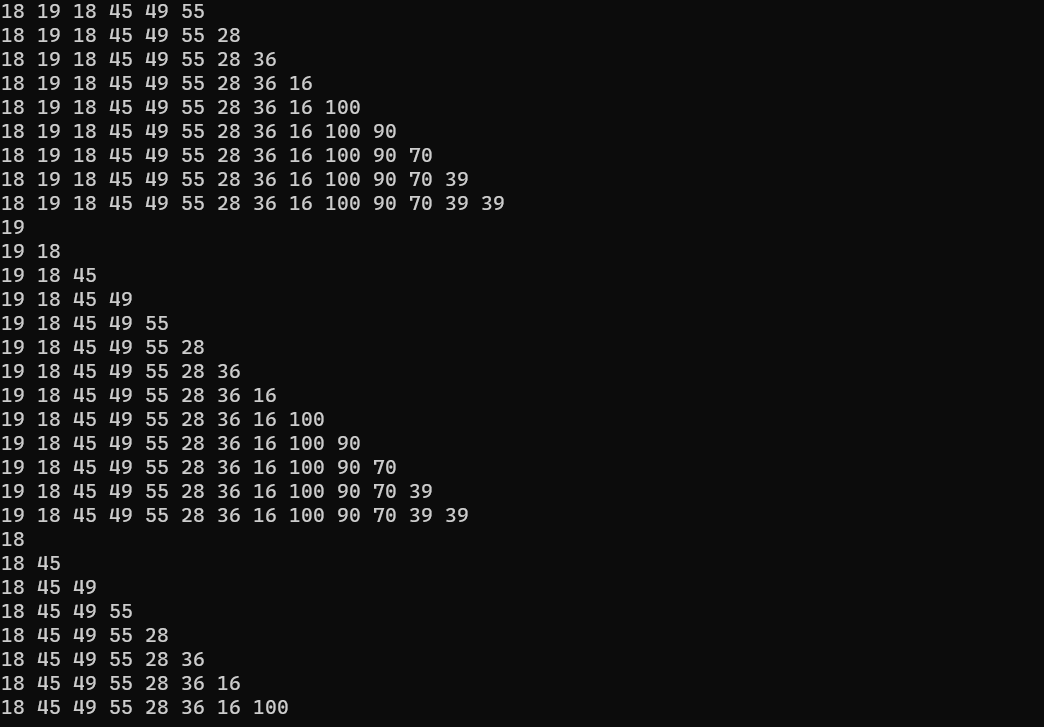
Hình 34. Test case 4 bài 1.9 ảnh 3



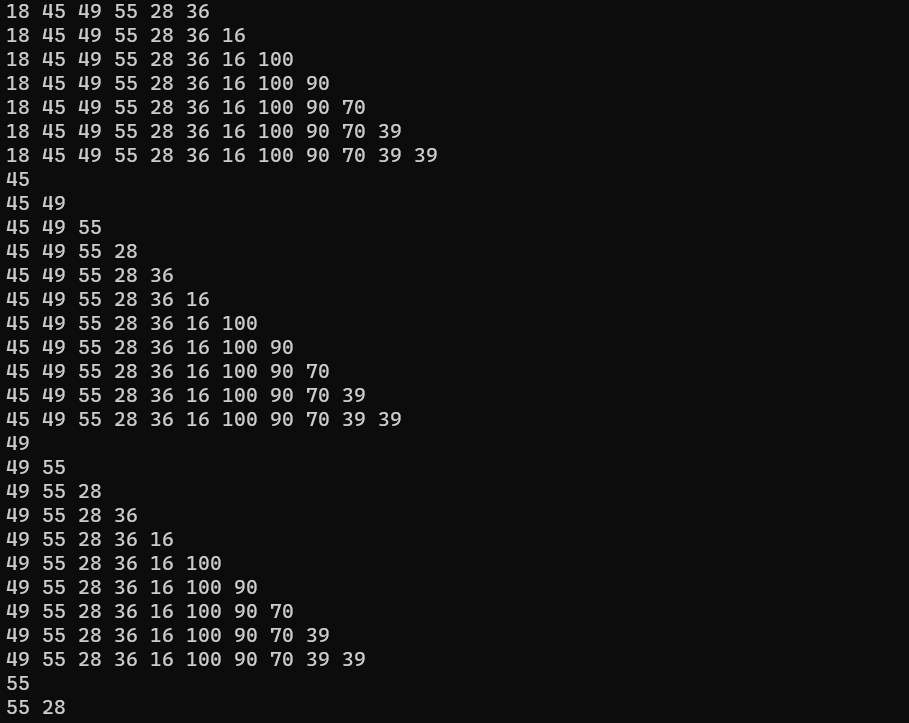
Hình 35. Test case 4 bài 1.9 ảnh 4



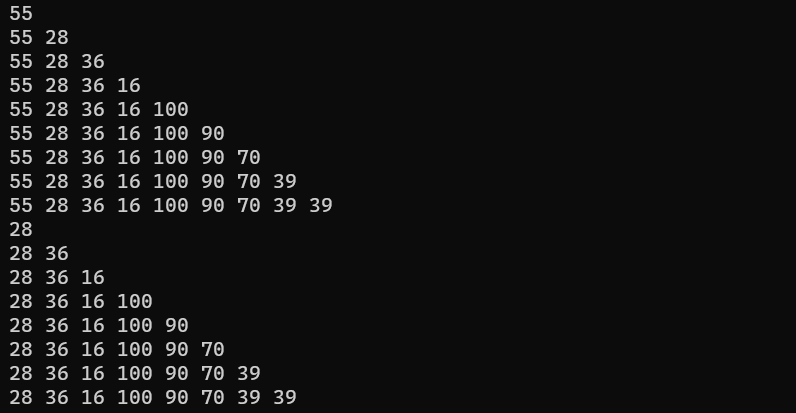
Hình 36. Test case 4 bài 1.9 ảnh 5



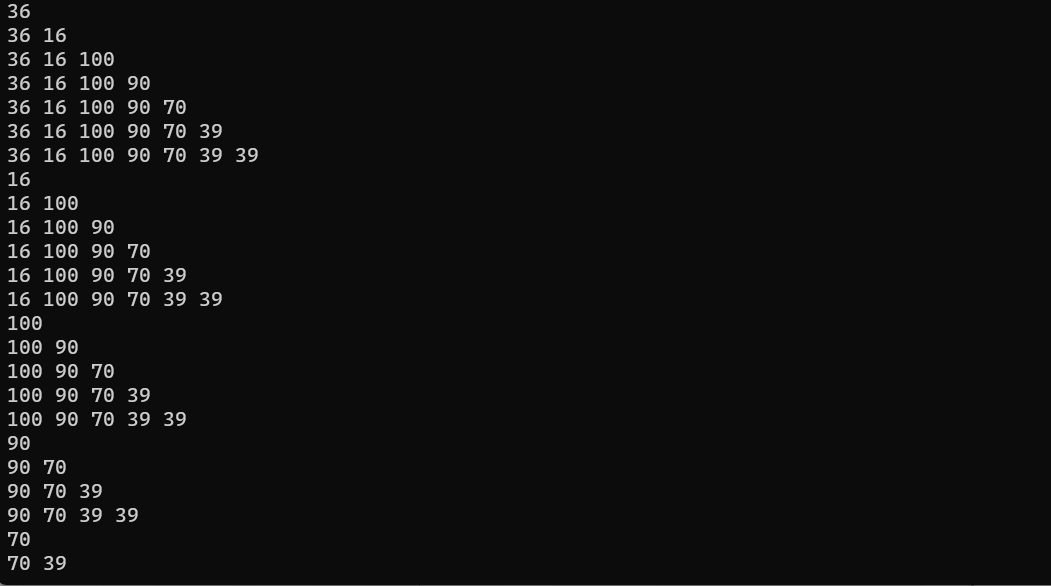
Hình . Test case 4 bài 1.9 ảnh 6



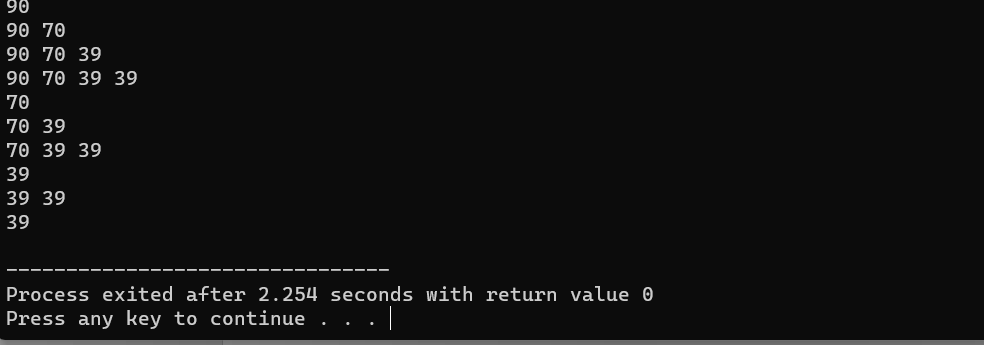
Hình . Test case 4 bài 1.9 ảnh 7



Hình 39. Test case 4 bài 1.9 ảnh 8



Hình 40. Test case 4 bài 1.9 ảnh 9



Hình 41. Test case 4 bài 1.9 ảnh 10

//PHAN HOANG NAM - 20215434

/\*Viết chương trình in ra tất cả các dãy con của một dãy cho trước. Ví dụ dãy 1 3 4 2 có các dãy con sau:

1

1 3

1 3 4

1 3 4 2

3

3 4

3 4 2

4

4 2

2

\*/

#include<stdio.h>

#include<stdlib.h>

void allocate\_mem(int \*\*arr, int n){

\*arr = (int\*)malloc(n\*sizeof(int)); //cap phat bo nho cho mang

if(arr==NULL) { //neu arr = NULL thi in ra thong bao loi cap phat va ket thuc ham

printf("memory allocation error\n");

return;

}

}

int main(){

int n; //khai bao bien n la kich thuoc mang

int \*arr; //khai bao con tro kieu int

printf("PHAN HOANG NAM - 20215434\n");

scanf("%d", &n); //nhap n

allocate\_mem(&arr, n); //cap phat bo nho cho mang

for(int i=0; i<n; i++) scanf("%d", &arr[i]); //nhap du lieu cho mang

for(int i=0; i<n; i++){ //in ra cac day con

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

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

printf("%d ", arr[k]);

}

printf("\n");

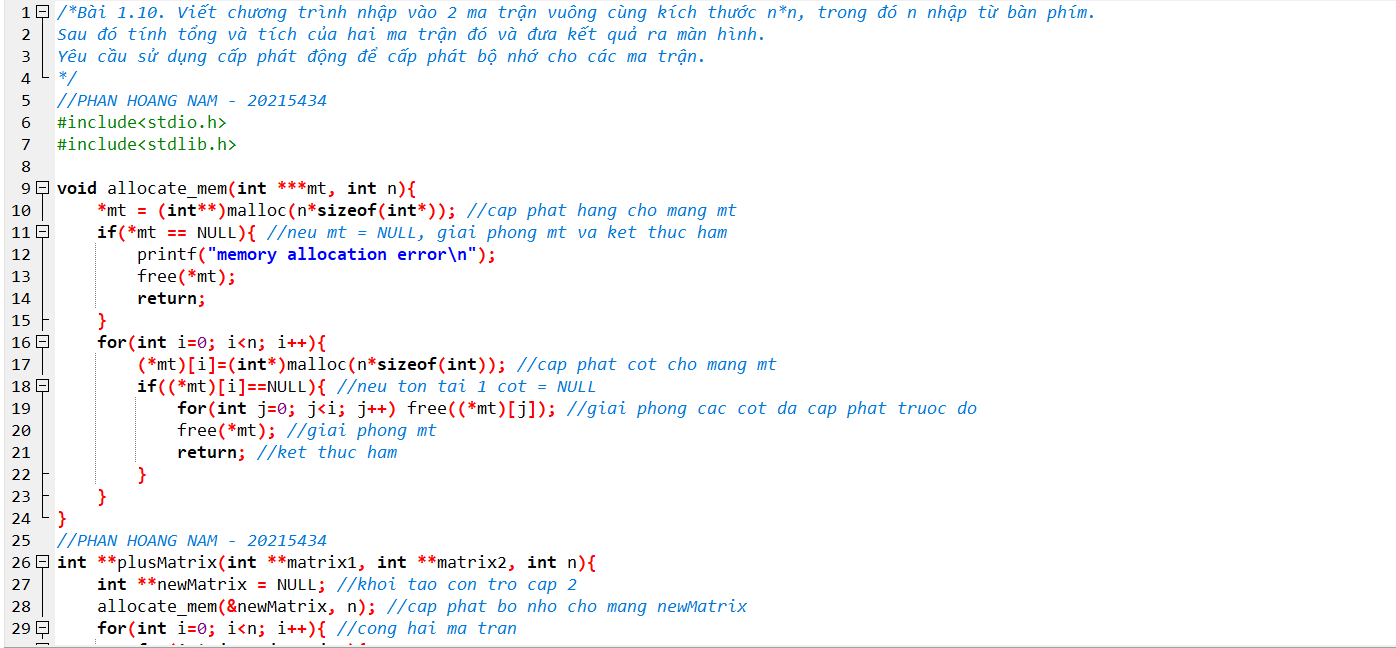
}

}

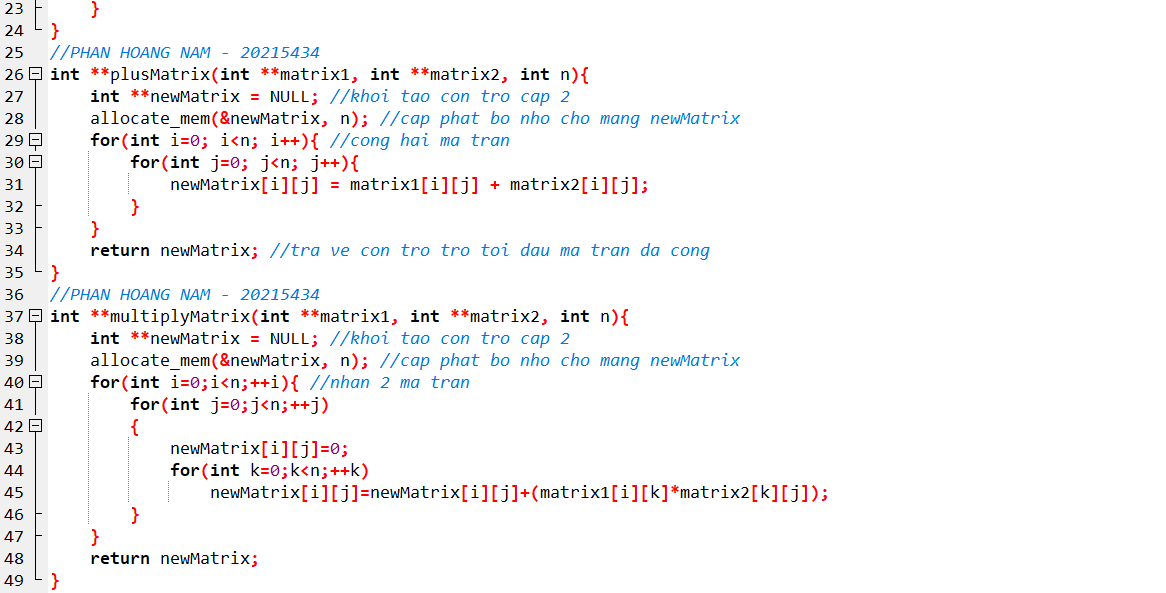
}

//PHAN HOANG NAM – 20215434

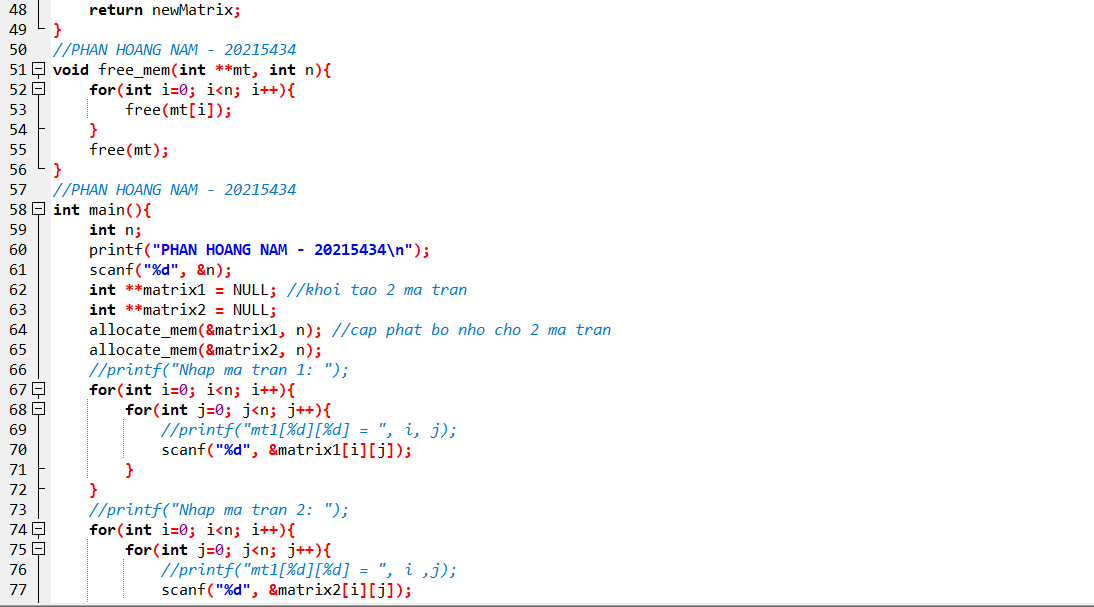
## **Bài 1.10.** Viết chương trình nhập vào 2 ma trận vuông cùng kích thước n\*n, trong đó n nhập từ bàn phím. Sau đó tính tổng và tích của hai ma trận đó và đưa kết quả ra màn hình. Yêu cầu sử dụng cấp phát động để cấp phát bộ nhớ cho các ma trận.



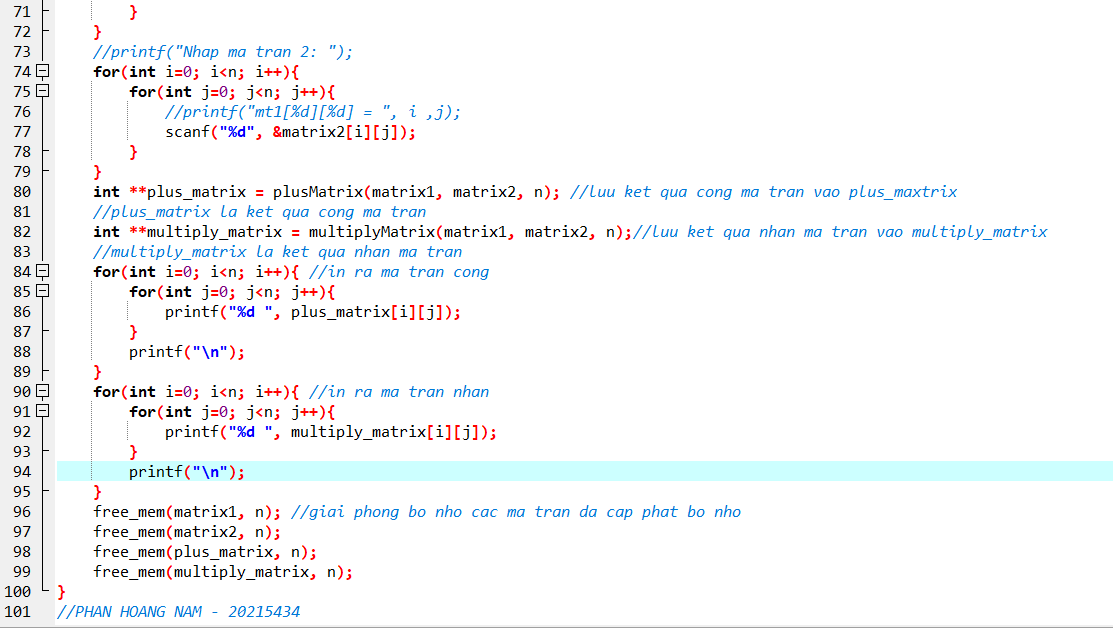
Hình 42. Code bài 1.10 ảnh 1



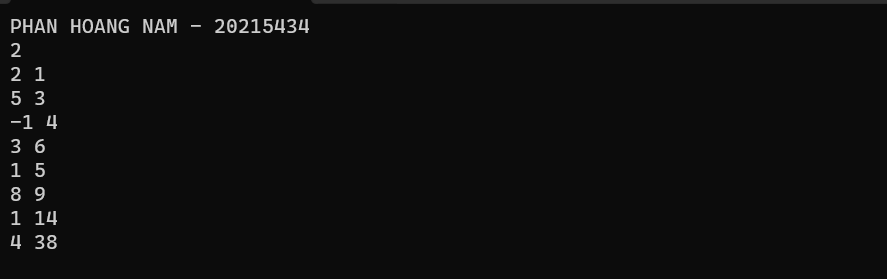
Hình 43. Code bài 1.10 ảnh 2



Hình 44. Code bài 1.10 ảnh 3



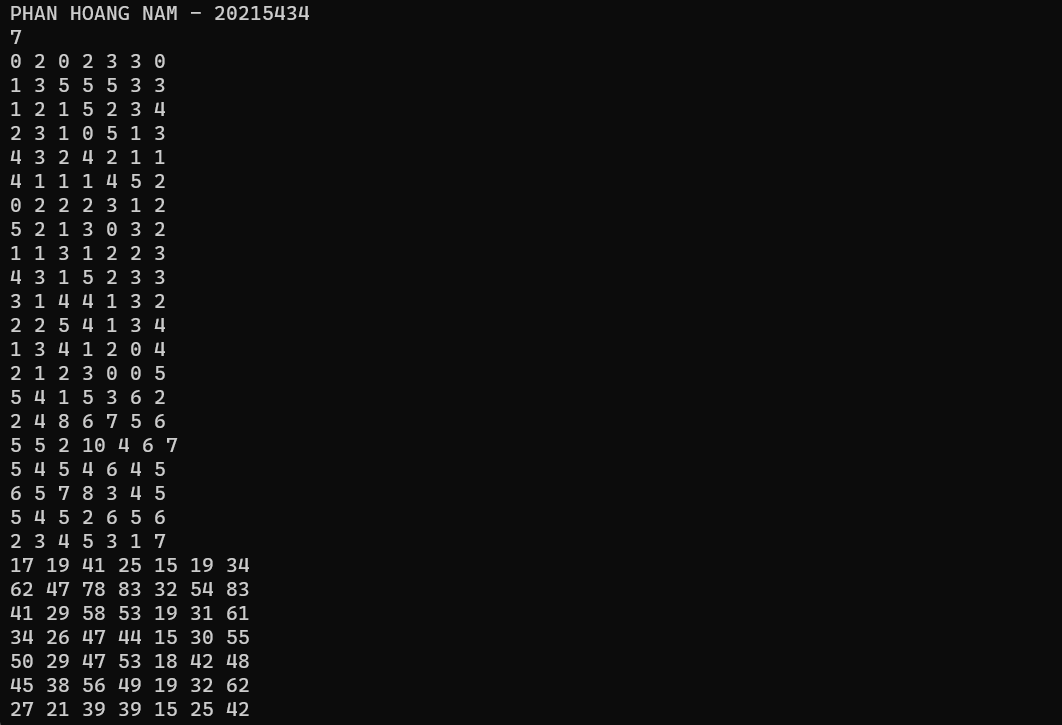
Hình 45. Code bài 1.10 ảnh 4



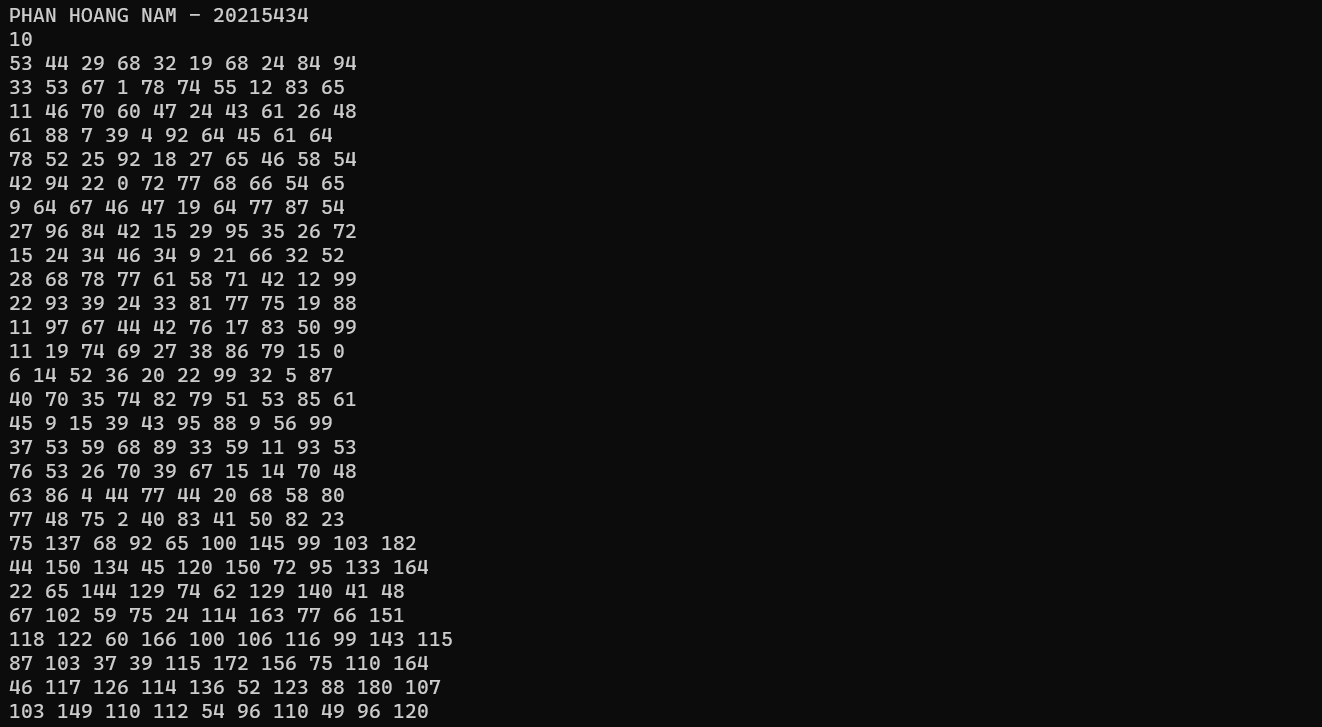
Hình 46. Test case 1 bài 1.10



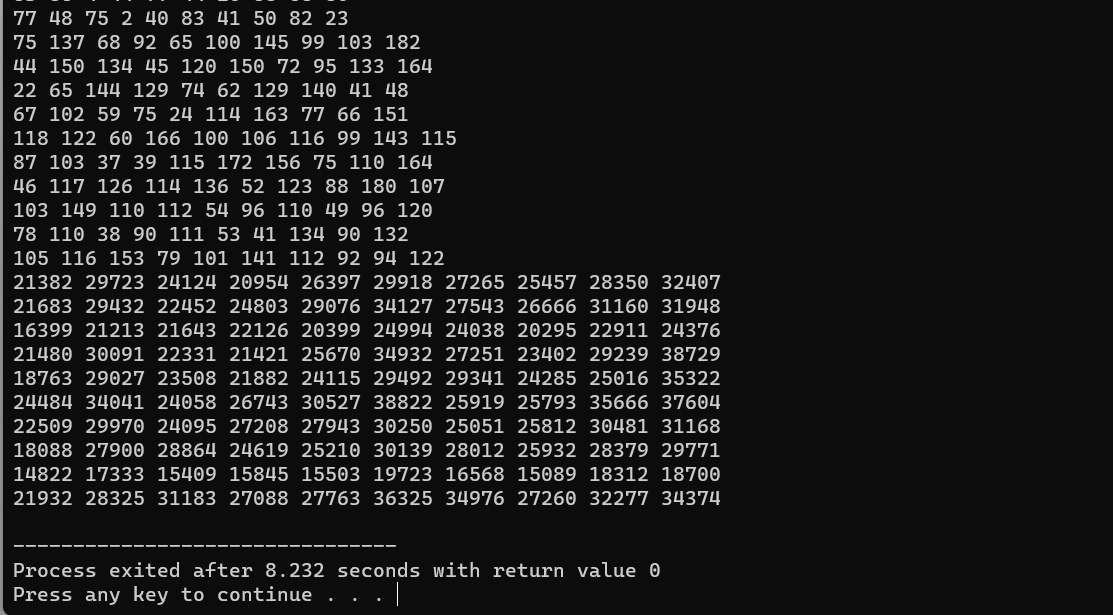
Hình 47. Test case 2 bài 1.10



Hình 48. Test case 3 bài 1.10



Hình 49. Test case 4 bài 1.10 ảnh 1



Hình 50. Test case 4 bài 1.10 ảnh 2

/\*Bài 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.

Yêu cầu sử dụng cấp phát động để cấp phát bộ nhớ cho các ma trận.

\*/

//PHAN HOANG NAM - 20215434

#include<stdio.h>

#include<stdlib.h>

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

\*mt = (int\*\*)malloc(n\*sizeof(int\*)); //cap phat hang cho mang mt

if(\*mt == NULL){ //neu mt = NULL, giai phong mt va ket thuc ham

printf("memory allocation error\n");

free(\*mt);

return;

}

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

(\*mt)[i]=(int\*)malloc(n\*sizeof(int)); //cap phat cot cho mang mt

if((\*mt)[i]==NULL){ //neu ton tai 1 cot = NULL

for(int j=0; j<i; j++) free((\*mt)[j]); //giai phong cac cot da cap phat truoc do

free(\*mt); //giai phong mt

return; //ket thuc ham

}

}

}

//PHAN HOANG NAM - 20215434

int \*\*plusMatrix(int \*\*matrix1, int \*\*matrix2, int n){

int \*\*newMatrix = NULL; //khoi tao con tro cap 2

allocate\_mem(&newMatrix, n); //cap phat bo nho cho mang newMatrix

for(int i=0; i<n; i++){ //cong hai ma tran

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

newMatrix[i][j] = matrix1[i][j] + matrix2[i][j];

}

}

return newMatrix; //tra ve con tro tro toi dau ma tran da cong

}

//PHAN HOANG NAM - 20215434

int \*\*multiplyMatrix(int \*\*matrix1, int \*\*matrix2, int n){

int \*\*newMatrix = NULL; //khoi tao con tro cap 2

allocate\_mem(&newMatrix, n); //cap phat bo nho cho mang newMatrix

for(int i=0;i<n;++i){ //nhan 2 ma tran

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

{

newMatrix[i][j]=0;

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

newMatrix[i][j]=newMatrix[i][j]+(matrix1[i][k]\*matrix2[k][j]);

}

}

return newMatrix;

}

//PHAN HOANG NAM - 20215434

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

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

free(mt[i]);

}

free(mt);

}

//PHAN HOANG NAM - 20215434

int main(){

int n;

printf("PHAN HOANG NAM - 20215434\n");

scanf("%d", &n);

int \*\*matrix1 = NULL; //khoi tao 2 ma tran

int \*\*matrix2 = NULL;

allocate\_mem(&matrix1, n); //cap phat bo nho cho 2 ma tran

allocate\_mem(&matrix2, n);

//printf("Nhap ma tran 1: ");

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

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

//printf("mt1[%d][%d] = ", i, j);

scanf("%d", &matrix1[i][j]);

}

}

//printf("Nhap ma tran 2: ");

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

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

//printf("mt1[%d][%d] = ", i ,j);

scanf("%d", &matrix2[i][j]);

}

}

int \*\*plus\_matrix = plusMatrix(matrix1, matrix2, n); //luu ket qua cong ma tran vao plus\_maxtrix

//plus\_matrix la ket qua cong ma tran

int \*\*multiply\_matrix = multiplyMatrix(matrix1, matrix2, n);//luu ket qua nhan ma tran vao multiply\_matrix

//multiply\_matrix la ket qua nhan ma tran

for(int i=0; i<n; i++){ //in ra ma tran cong

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

printf("%d ", plus\_matrix[i][j]);

}

printf("\n");

}

for(int i=0; i<n; i++){ //in ra ma tran nhan

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

printf("%d ", multiply\_matrix[i][j]);

}

printf("\n");

}

free\_mem(matrix1, n); //giai phong bo nho cac ma tran da cap phat bo nho

free\_mem(matrix2, n);

free\_mem(plus\_matrix, n);

free\_mem(multiply\_matrix, n);

}

//PHAN HOANG NAM – 20215434