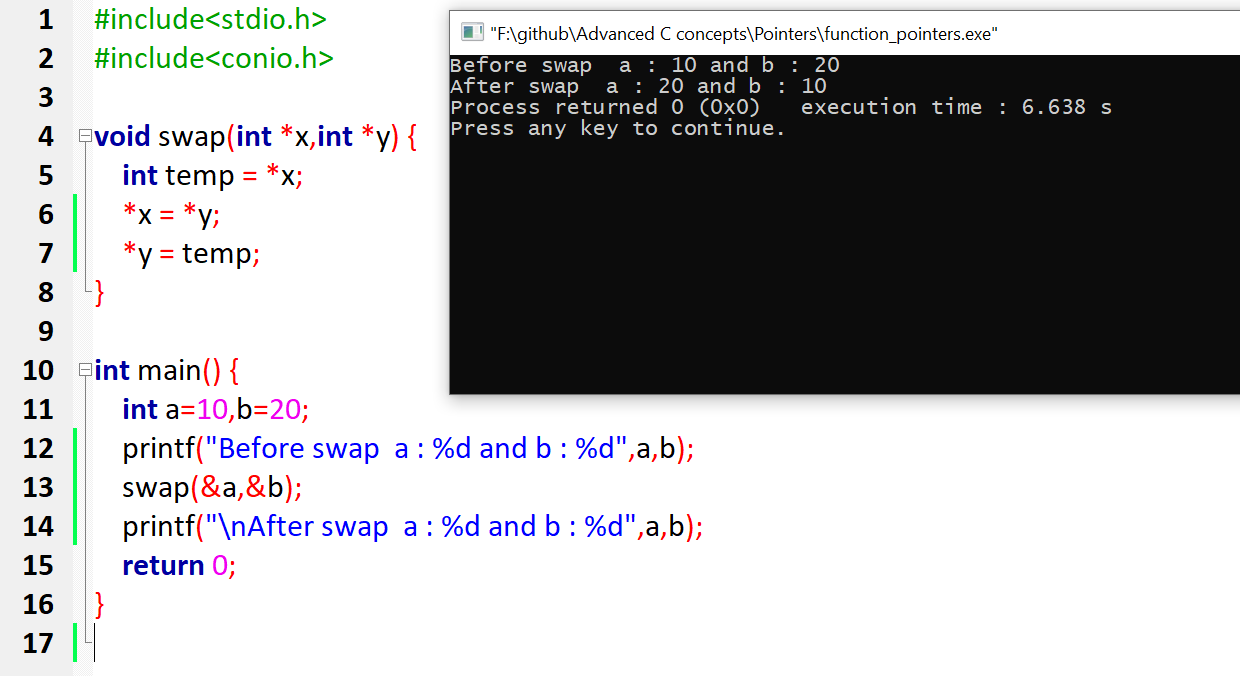
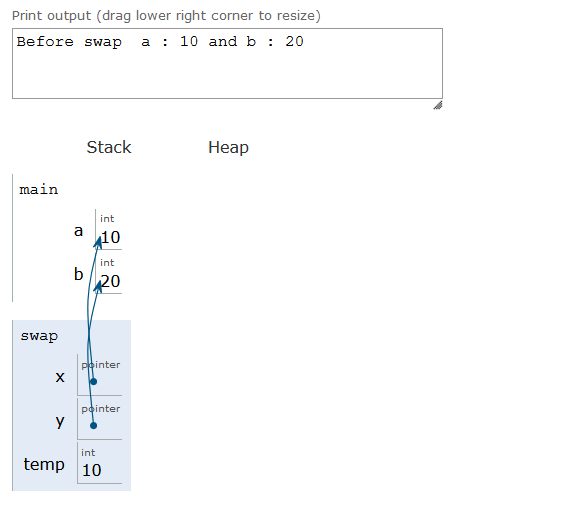
# Passing and returning values using pointers

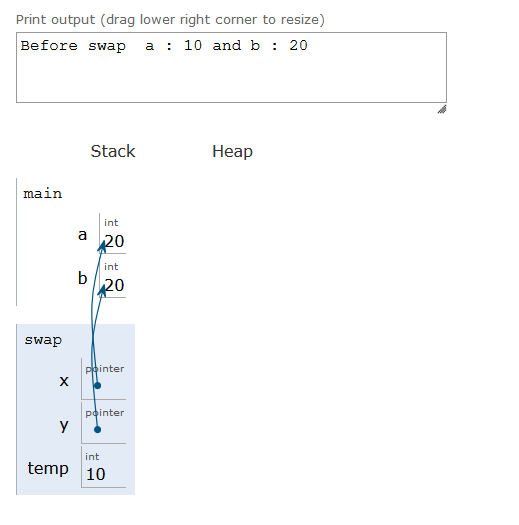
## Function pointers



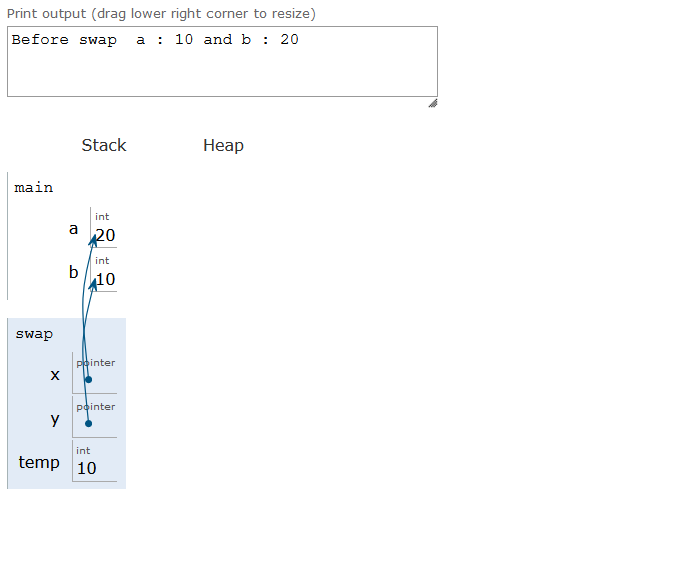
1)



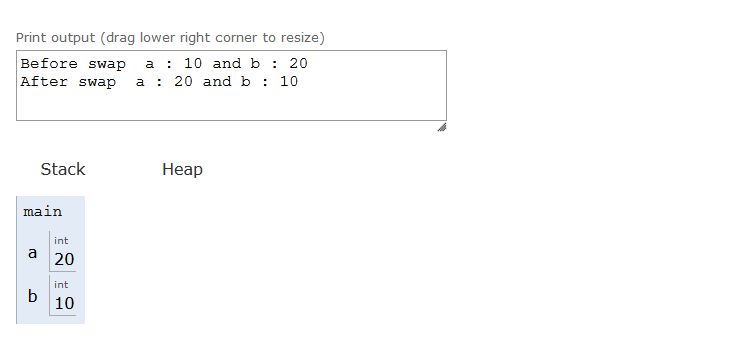
2)



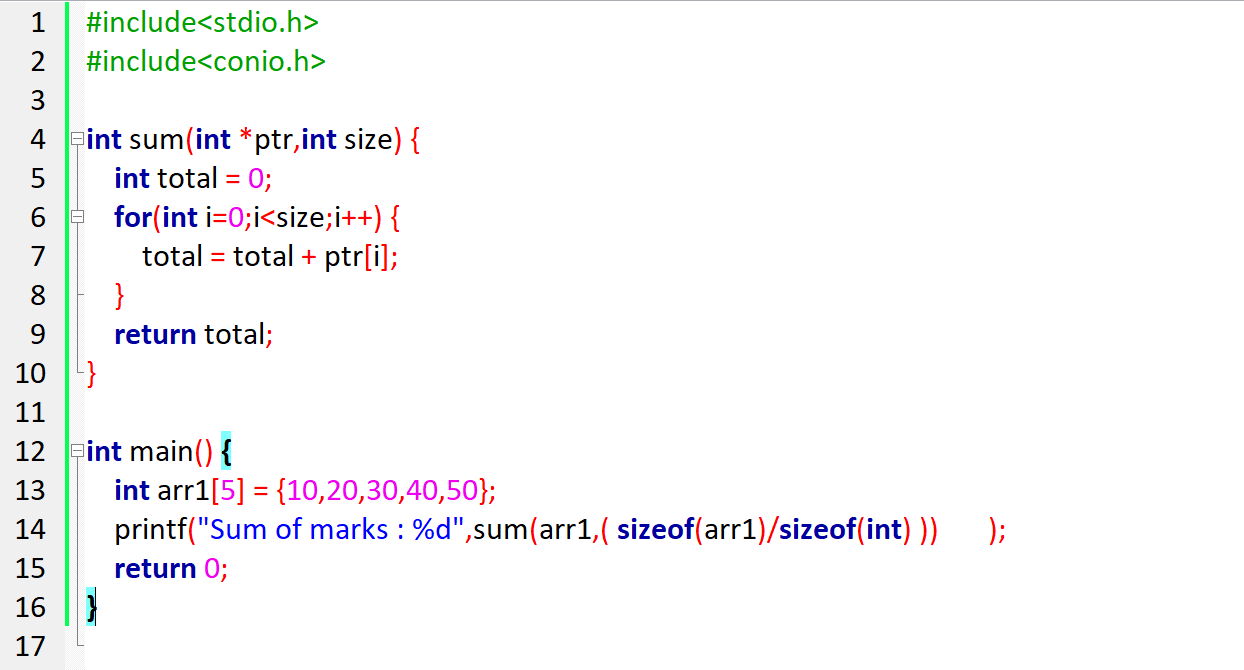
3)



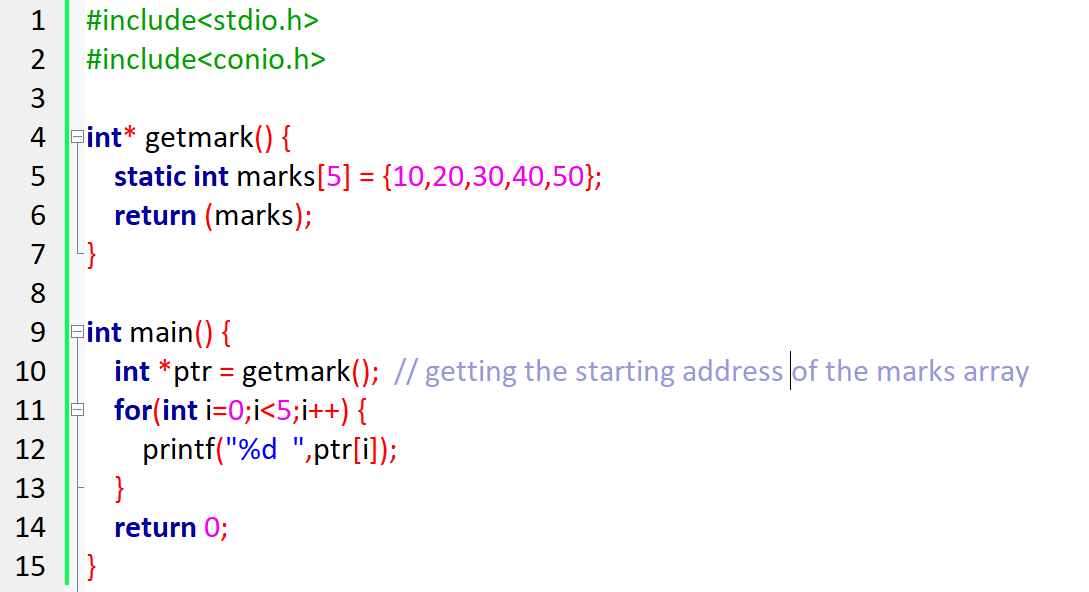
4)



## Function with Array Parameters



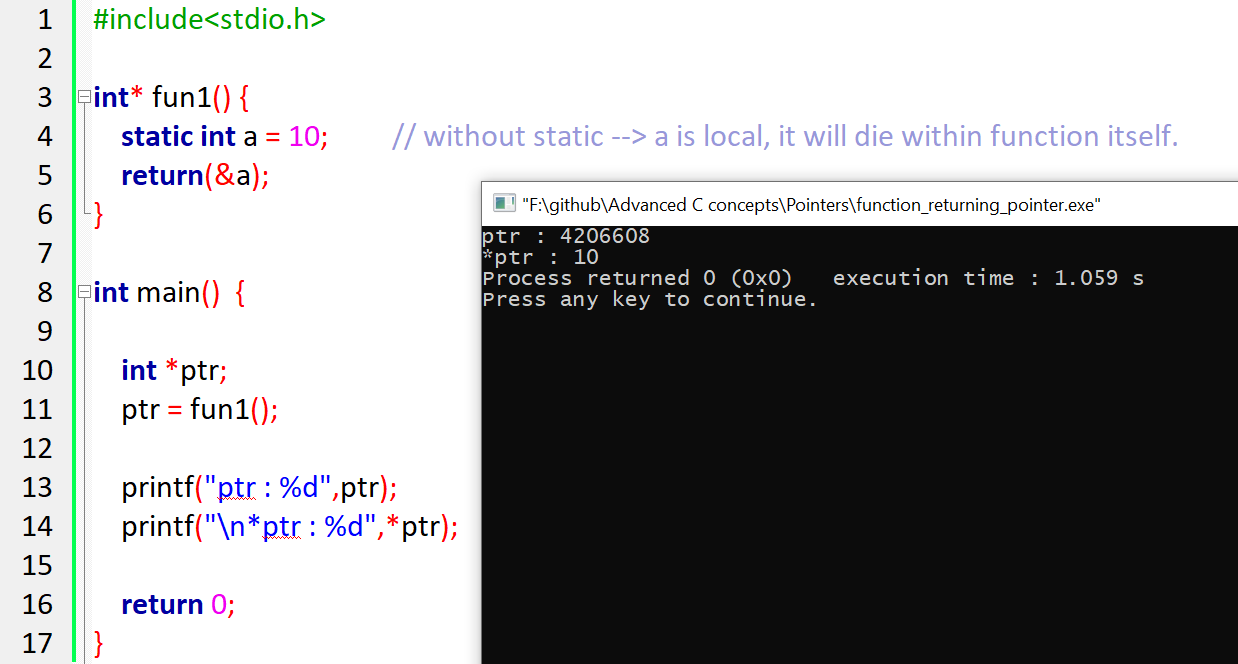
## Function that returns an Array



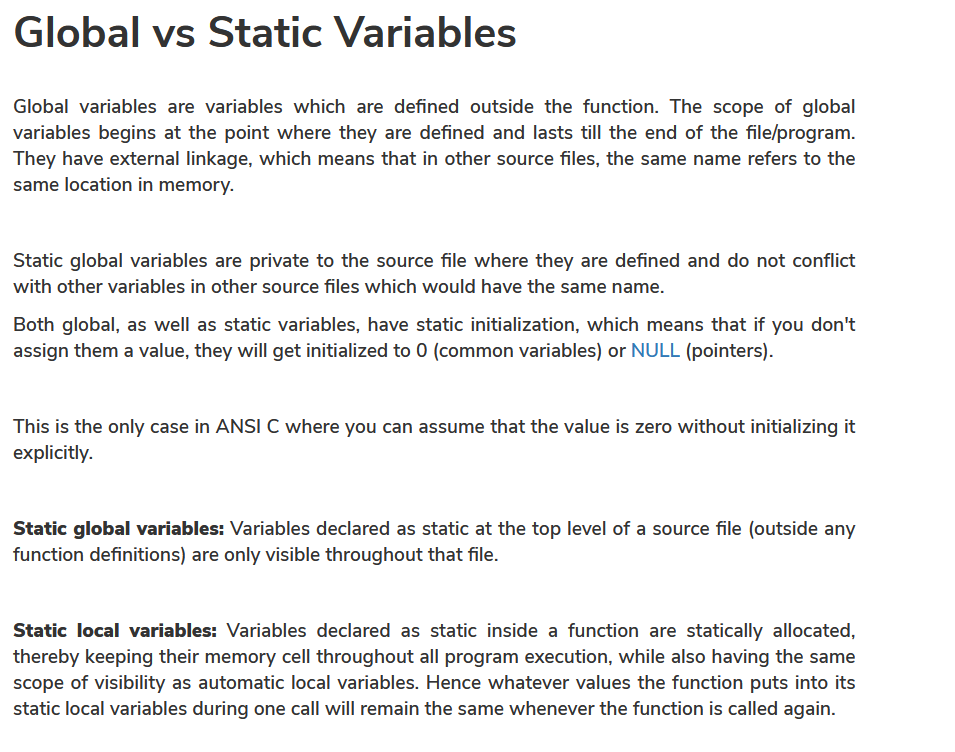
## Function returning pointer

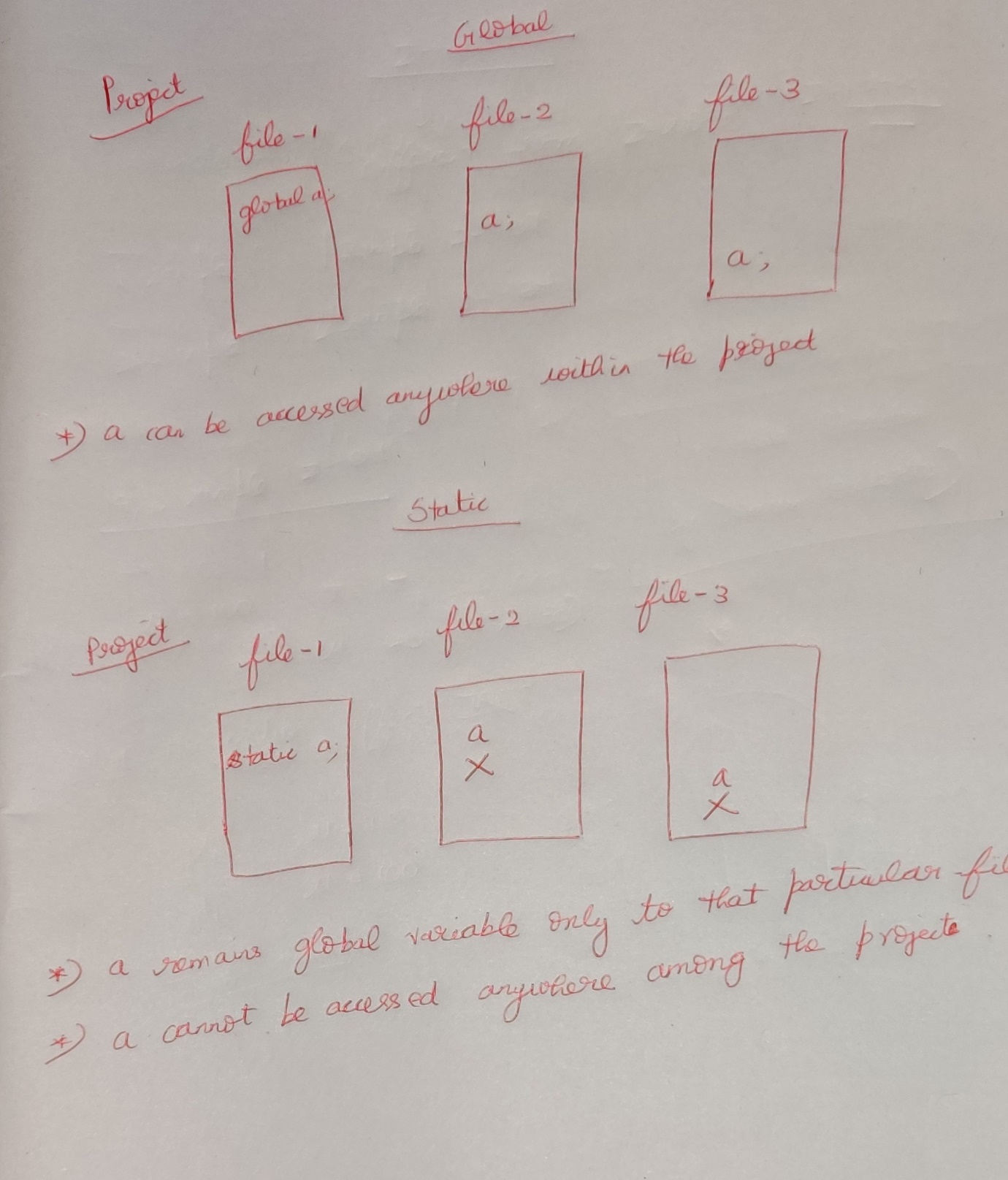
### Without static

### With static

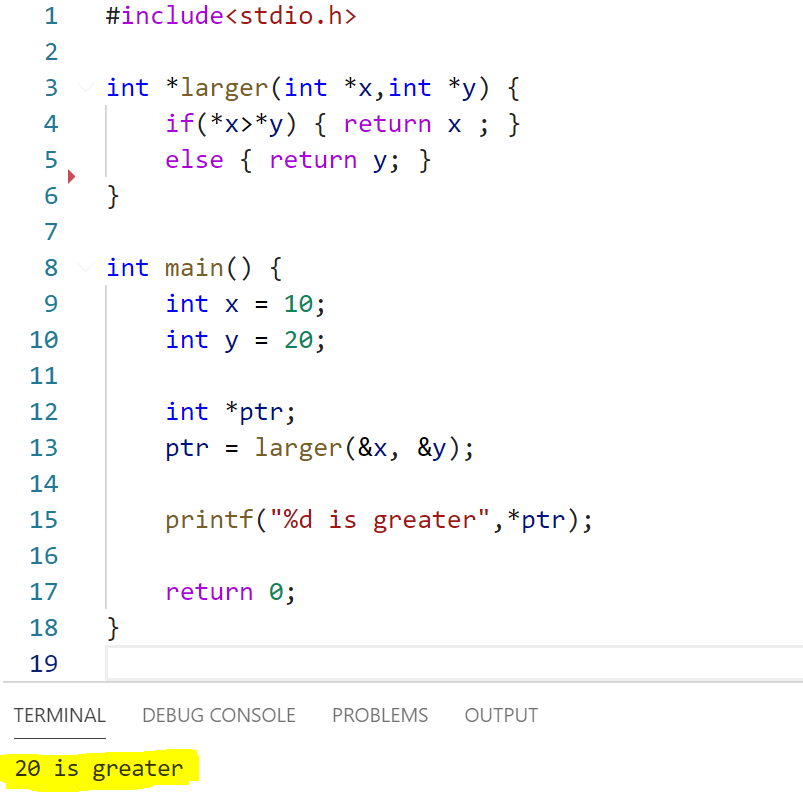


# **Difference between static and global variables**





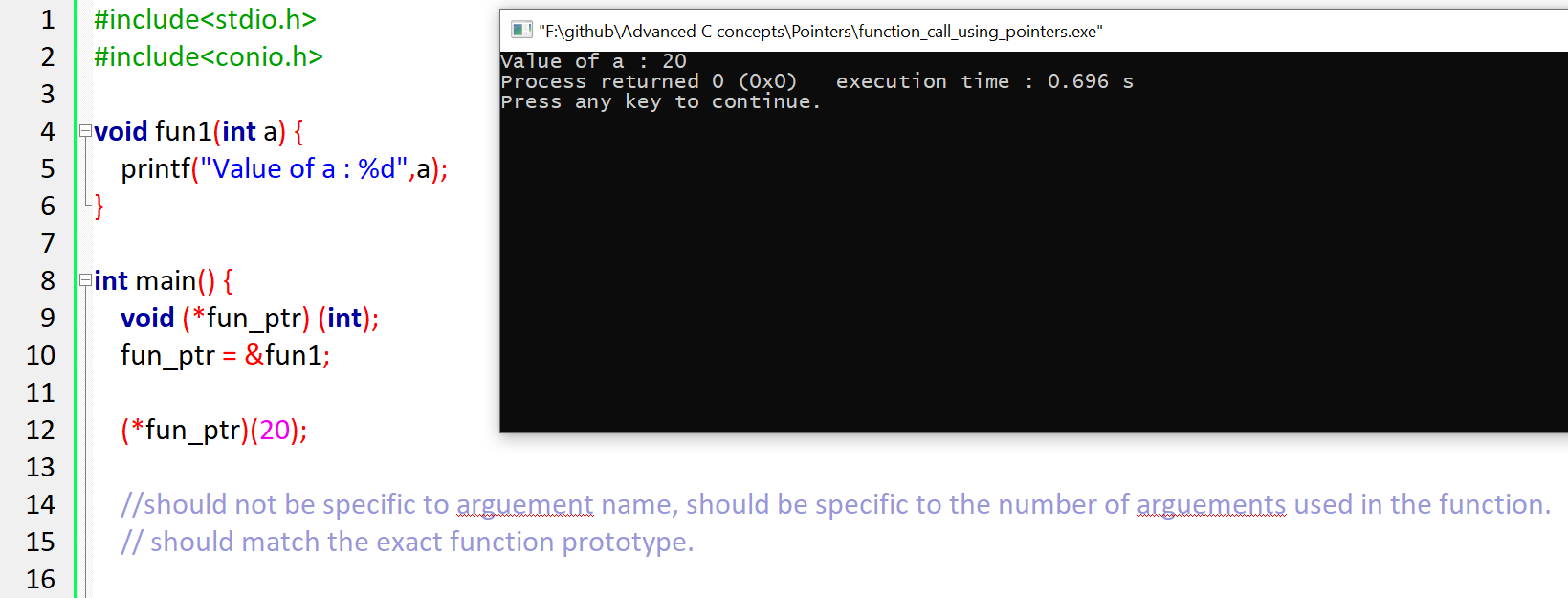
## **Function returning pointer variables**



**int\* fun1(int a, int b)** 🡪 a function returning a pointer value.

# **Pointer to a function**

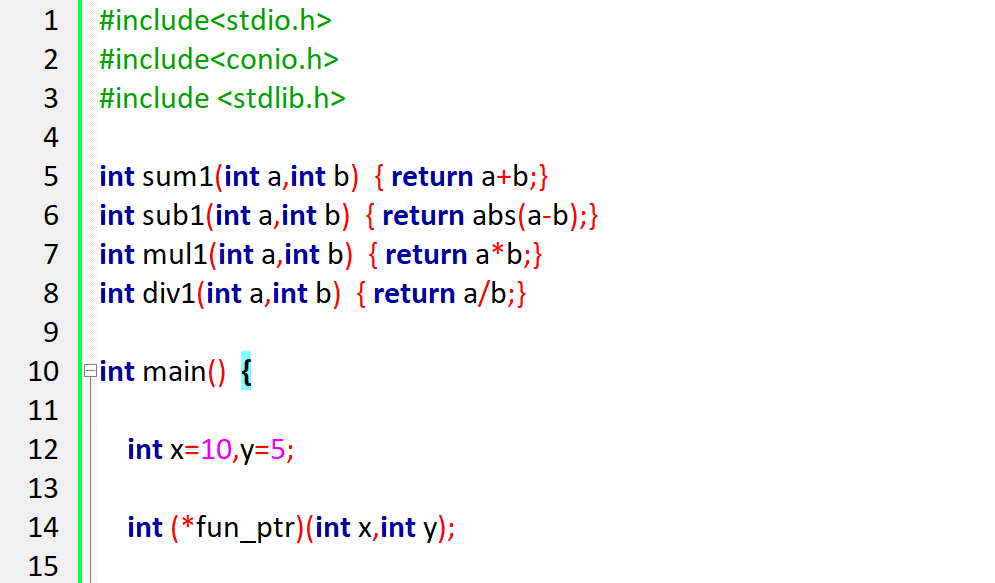
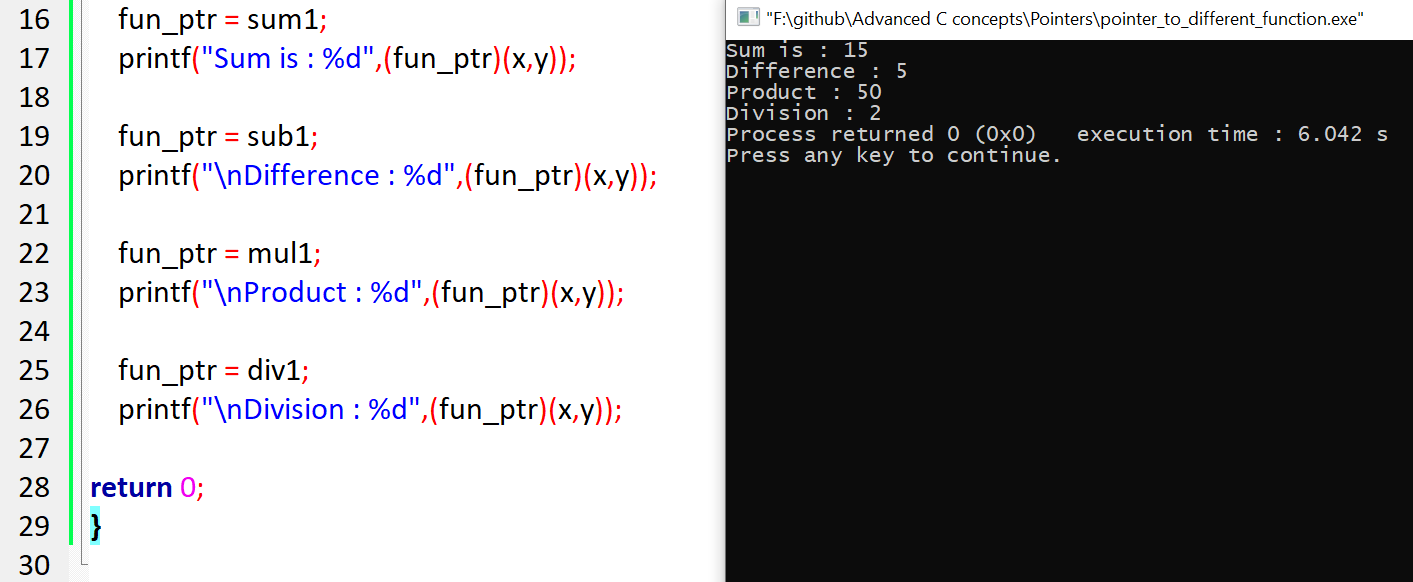
**int (\*fun1) (int a, int b)** 🡪 pointer to a function.



Illustrate how the function pointer can be changed to point to different functions during the program execution.

## **Pointer to different function**

Refer pointer\_to\_different\_function.c

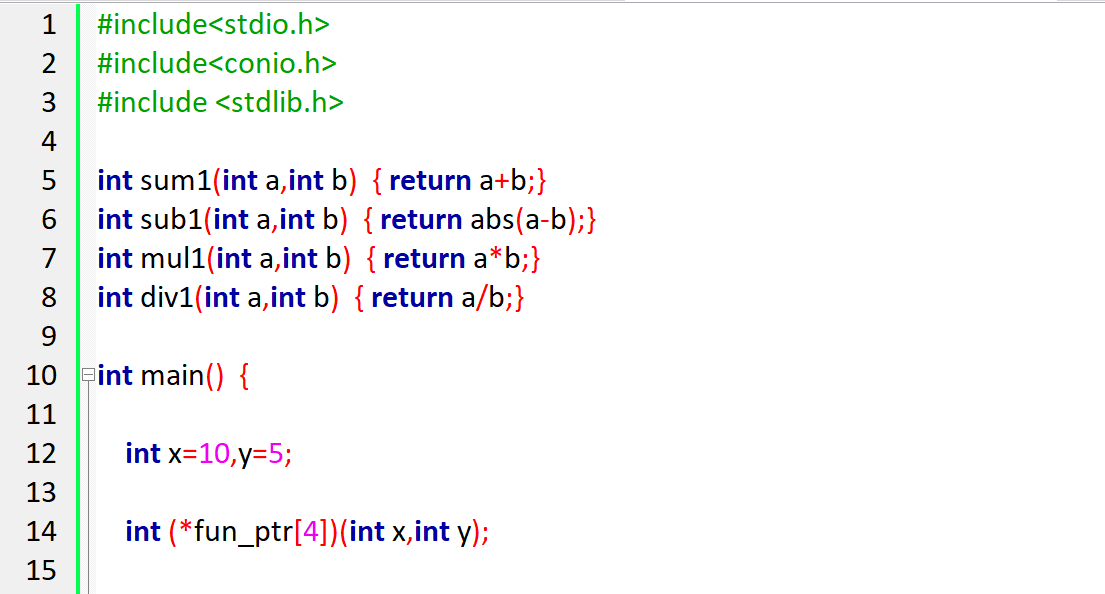
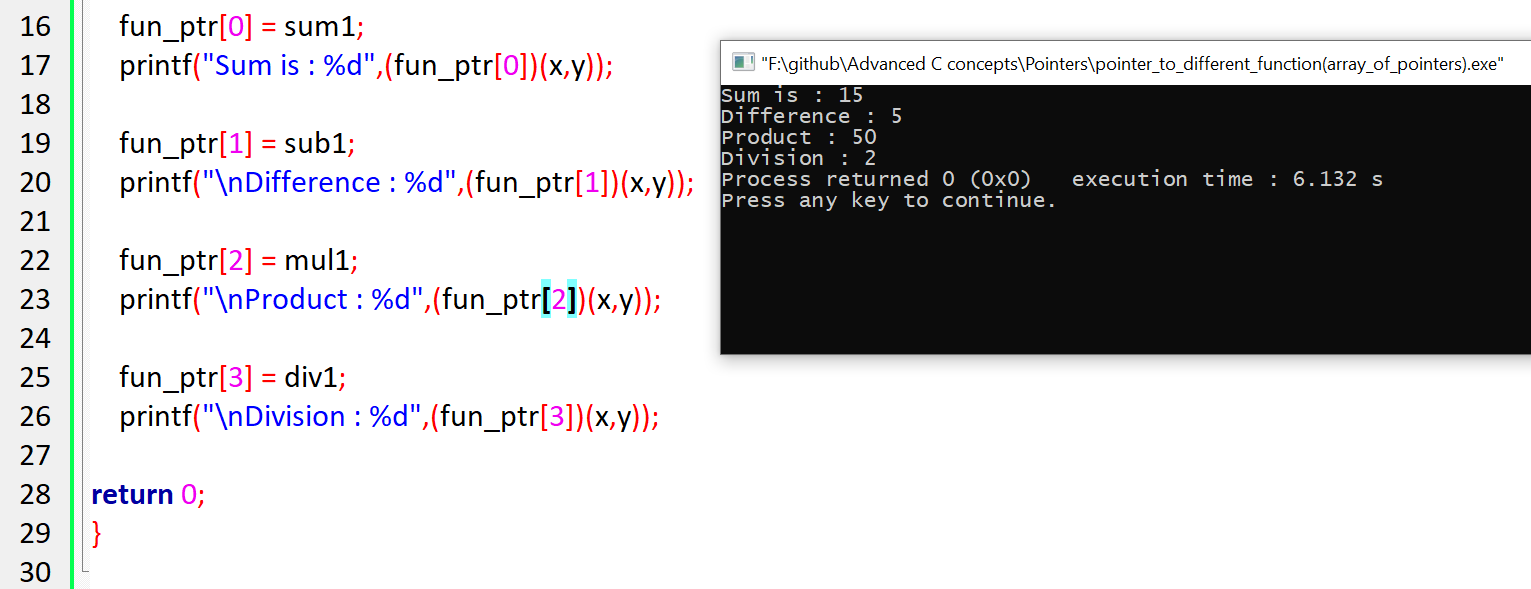
  


The same can be implemented with **array of pointers** also.

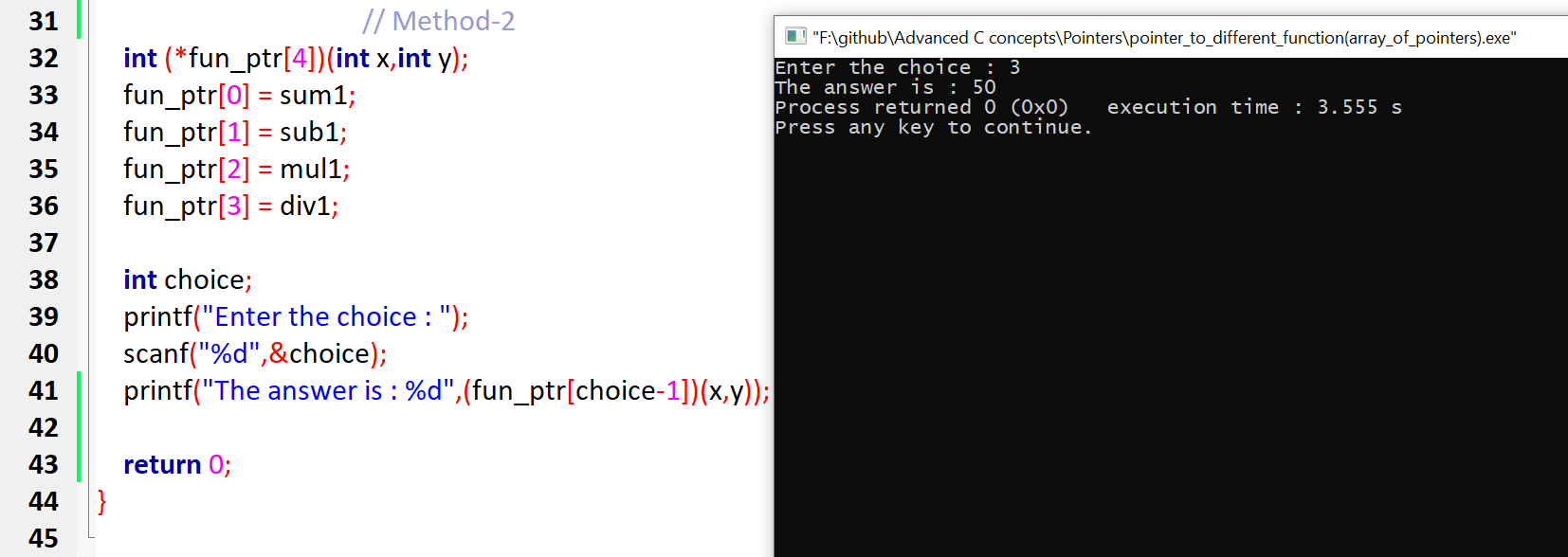
## **Pointer to different function (with array)**

Refer to pointer\_to\_different\_function(array\_of\_pointers).c

Method-1 (Array of pointers)

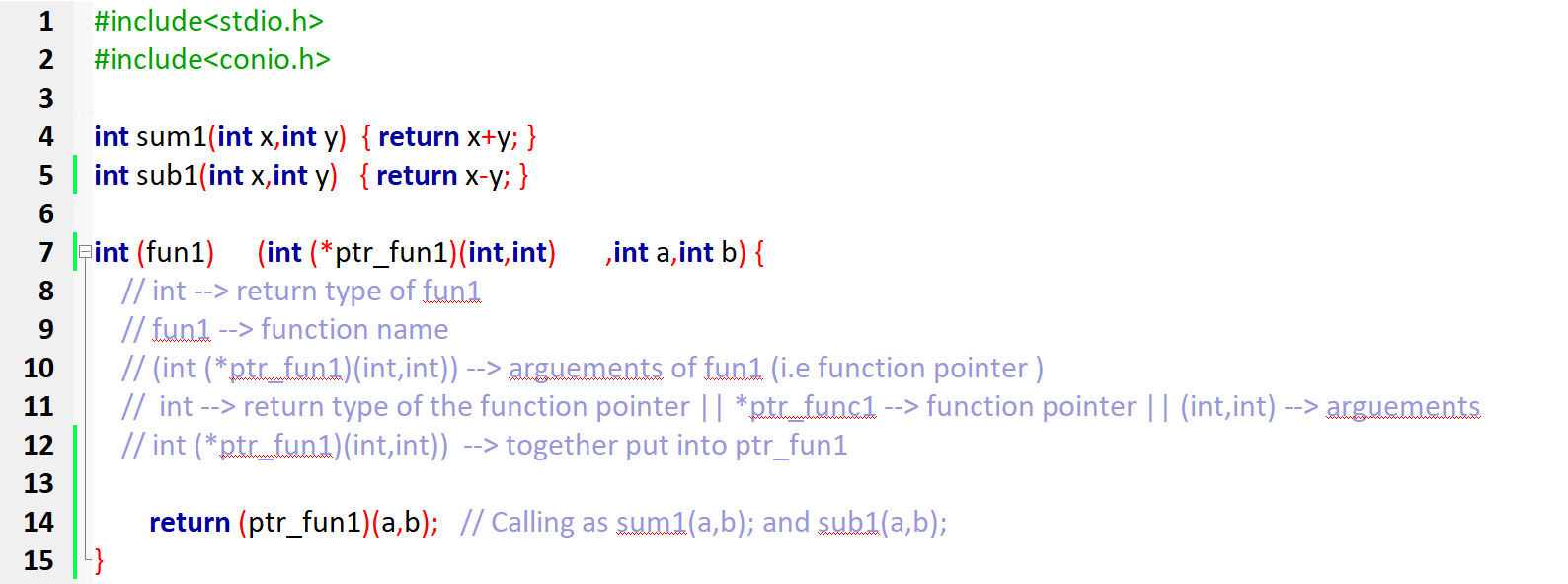
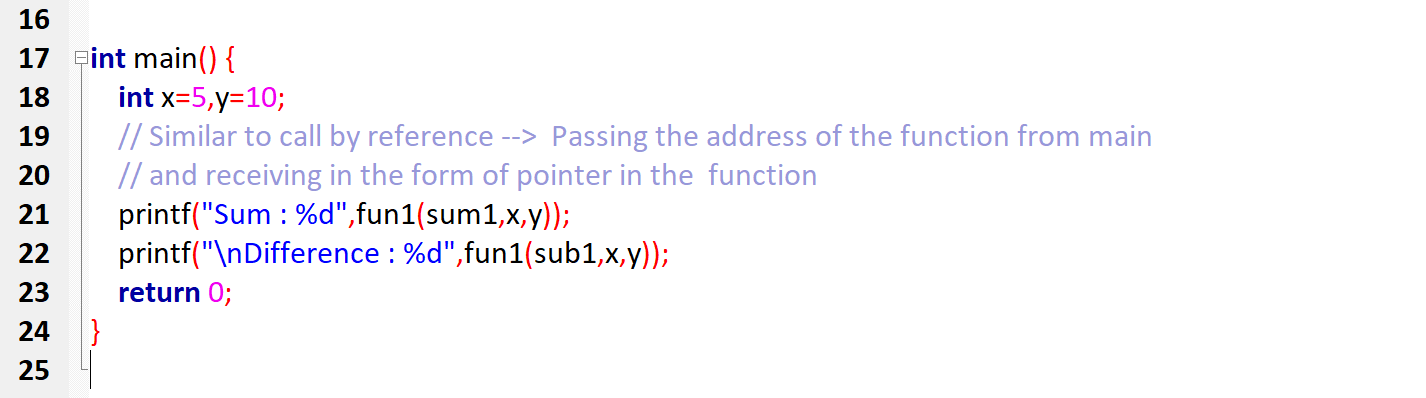
Method-2 (Array of function pointers)



# **Passing a Function pointer as an argument to a function**

This is nested function pointer (i.e function pointer within function pointer)

Refer function\_pointer\_as\_arguements.c

Output

