Segregate Algorithm
Task number (9)

Team members

Team leader	سمير نشأت محروس شحاته	20210428
	عبدالرحمن عمرو فتوح عبدالفتاح	20210518
	دنيا سيد محمد محمدي	202000297
	طارق وجيه حمزه	201900389
	عبدالله عصام محمد	20210550

Segregate positive and negative integers

Given an array of positive and negative integers, segregate them without changing the relative order of elements. The output should contain all positive numbers follow negative numbers while maintaining the same relative ordering.

we have made two algorithms, recursive and non recursive:

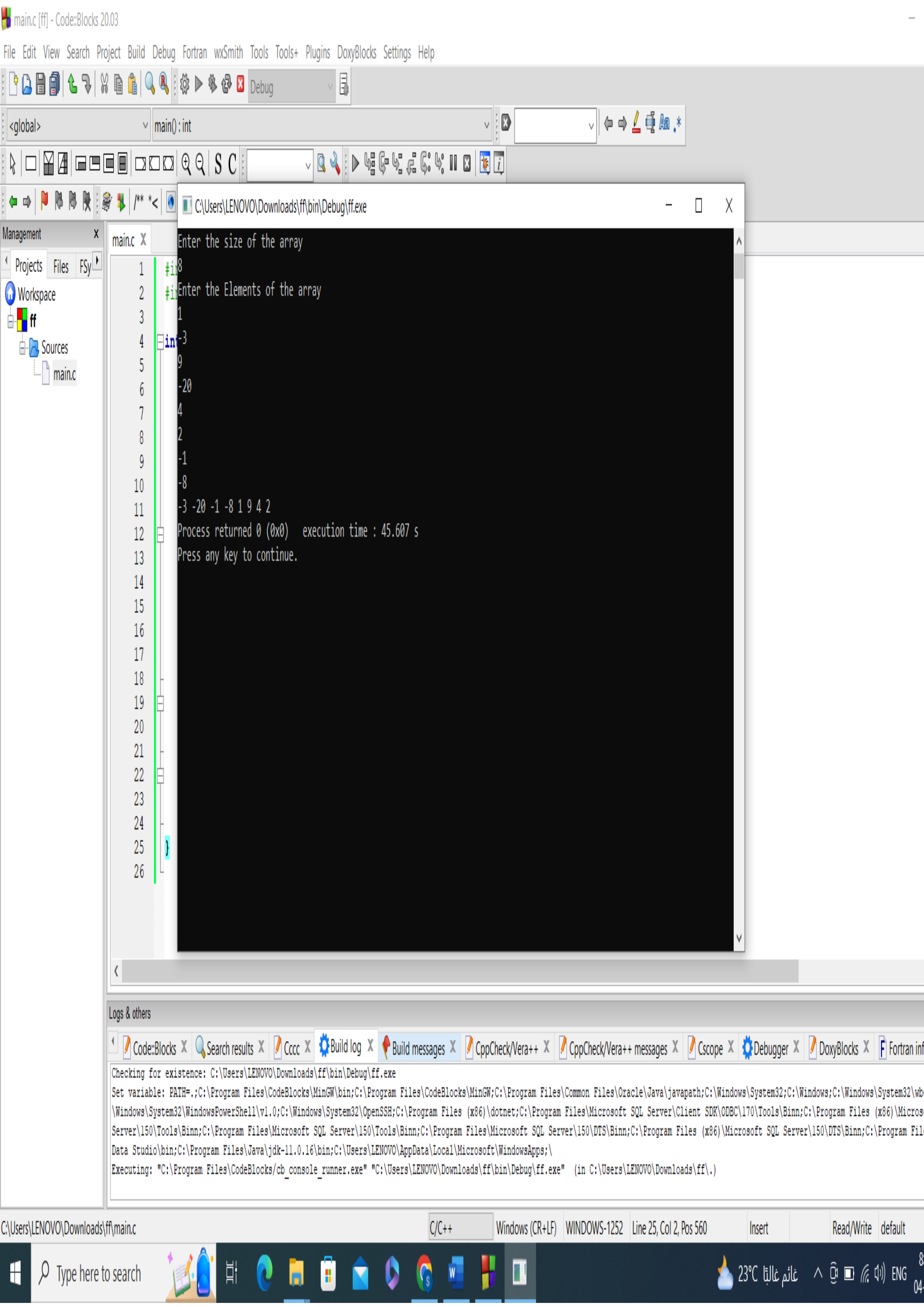
————————————the non recursive one——————————

```
1-Declare variables size, number, positive, negative, i as integers
2-Declare arrays x and y of size ’size’
3-Prompt the user to enter the size of the array and read the input into ’size’
4-Prompt the user to enter the elements of the array and iterate from i = 0 to i = size-1:
    a. Read the input into ’number’
    b. If ’number’ is positive, add it to the ’y’ array and increment ’positive’
    c. If ’number’ is negative, add it to the ’x’ array and increment ’negative’
5-Copy the elements of the ’y’ array to the end of the ’x’ array starting from x[negative]
6-Print all elements of Array x
```

————————————

Complexity

The time complexity of this algorithm is O(n), where n is the size of the input array.
The program iterates through the input array only once (in the for loop from i=0 to i=size-1) to separate the positive and negative elements into two separate arrays. This takes O(n) time.
Then, the program concatenates the two arrays by copying the elements of the y[] array to the end of the x[] array. This also takes O(n) time, since it iterates through the y[] array once (in the for loop from i=0 to i=positive-1) and copies each element to the x[] array.
Finally, the program iterates through the merged array once (in the for loop from i=0 to i=size-1) to print its elements. This also takes O(n) time.
Therefore, the total time complexity of the algorithm is O(n) + O(n) + O(n) = O(n)


main.c [ff] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
main() : int
Management
Projects Files FSy
Workspace
ff
Sources
main.c
C:\Users\LENOVO\Downloads\ff\bin\Debug\ff.exe
Logs & others
Code::Blocks Search results Cccc Build log Build messages CppCheck/Vera++ CppCheck/Vera++ messages Cscope Debugger DoxyBlocks Fortran inf
C:\Users\LENOVO\Downloads\ff\main.c
C/C++ Windows (CR+LF) WINDOWS-1252 Line 25, Col 2, Pos 560 Insert Read/Write default
Type here to search
23°C غائم غالبا
ENG

```
Enter the size of the array
8
Enter the Elements of the array
1
-3
9
-20
4
2
-1
-8
-3 -20 -1 -8 1 9 4 2
Process returned 0 (0x0)   execution time : 45.607 s
Press any key to continue.
```

```
Checking for existence: C:\Users\LENOVO\Downloads\ff\bin\Debug\ff.exe
Set variable: PATH=.;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\CodeBlocks\MinGW;C:\Program Files\Common Files\Oracle\Java\javapath;C:\Windows\System32;C:\Windows;C:\Windows\System32\wb
\Windows\System32\WindowsPowerShell\v1.0;C:\Windows\System32\OpenSSH;C:\Program Files (x86)\dotnet;C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\Tools\Binn;C:\Program Files (x86)\Micros
Server\150\Tools\Binn;C:\Program Files\Microsoft SQL Server\150\Tools\Binn;C:\Program Files\Microsoft SQL Server\150\DTS\Binn;C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn;C:\Program Fil
Data Studio\bin;C:\Program Files\Java\jdk-11.0.16\bin;C:\Users\LENOVO\AppData\Local\Microsoft\WindowsApps;\
Executing: "C:\Program Files\CodeBlocks/cb_console_runner.exe" "C:\Users\LENOVO\Downloads\ff\bin\Debug\ff.exe"  (in C:\Users\LENOVO\Downloads\ff\.)
```

-the recursive one

1-Define a function called "SegregationPosNeg" which takes an array "arr", starting index "l", and ending index "r". 2-If l < r, then do the following: a. Calculate the middle index "m" using the formula "l + (r - l) / 2".

b. Recursively call SegregationPosNeg for the left subarray using "l" as the starting index and "m" as the ending index.

c. Recursively call SegregationPosNeg for the right subarray using "m + 1" as the starting index and "r" as the ending index.

d. Merge the left and right subarrays using a helper function "merge".

3-Define a function called "merge" which takes an array "arr", starting index "l", middle index "m", and ending index "r".

4-Calculate the sizes of the left and right subarrays using the formulas "n1 = m - l + 1" and "n2 = r - m".

5-Create two temporary arrays "L" and "R" with sizes n1 and n2, respectively.

6-Copy the elements from arr[l..m] to L[0..n1-1] and from arr[m+1..r] to R[0..n2-1].

7-Initialize three index variables i, j, and k to 0, representing the current index of the left subarray, right subarray, and merged subarray, respectively.

8-Merge the negative elements of the left and right subarrays by doing the following: a. While i < n1 and L[i] < 0, set arr[k] = L[i], i = i + 1, and k = k + 1.

b. While j < n2 and R[j] < 0, set arr[k]=R[j], j = j + 1, and k = k + 1.

9-Merge the positive elements of the left and right subarrays by doing the following: a. While i < n1, set arr[k] = L[i], i = i + 1, and k = k + 1. b. While j < n2, set arr[k] = R[j], j = j + 1, and k = k + 1.

10-Define a main function which does the following: a. Prompt the user to enter the size of the array.

b. Prompt the user to enter the array elements.

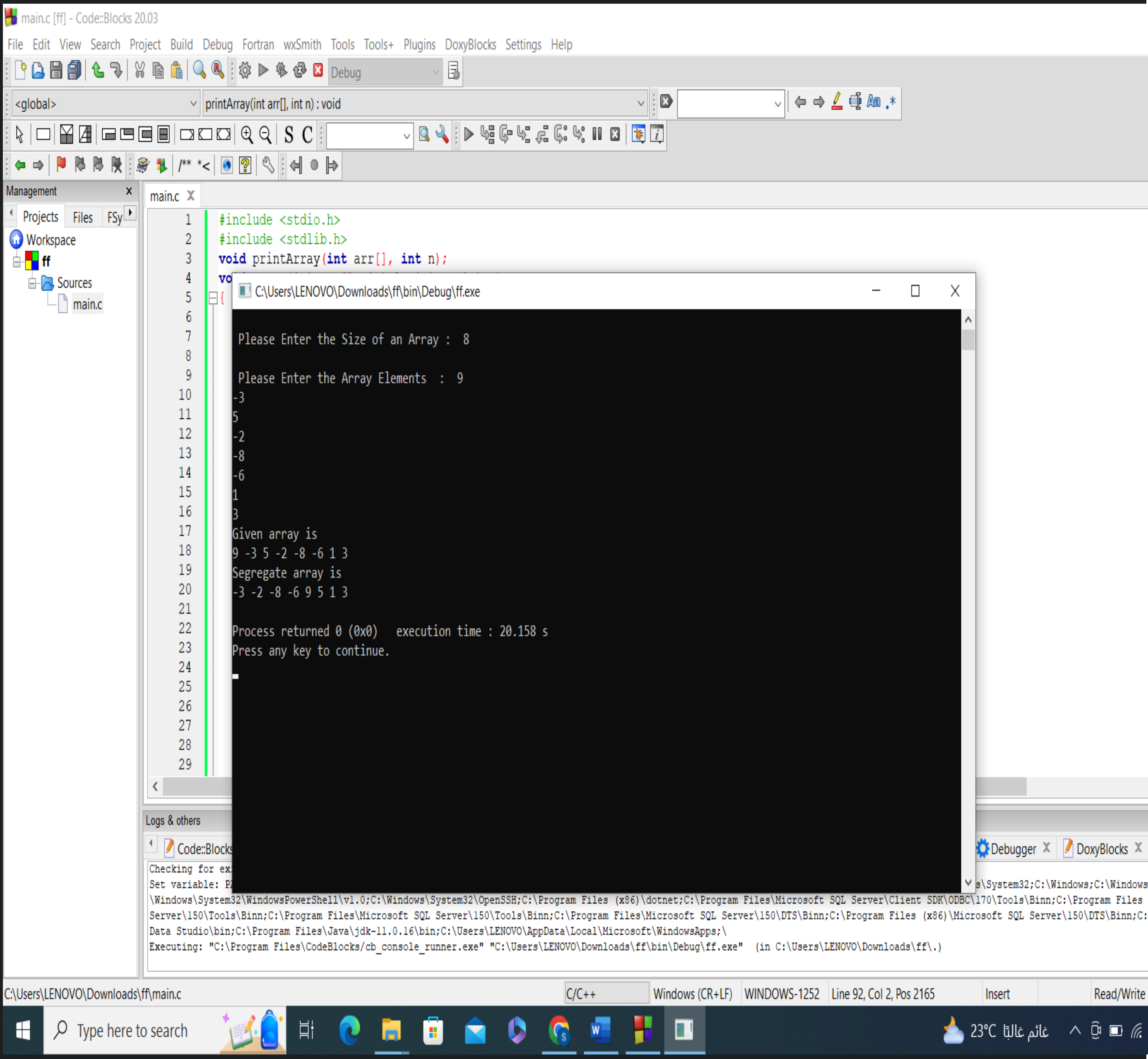
c. Call SegregationPosNeg to segregate the array.

d. Print the segregate array.

Complexity

The time complexity of this algorithm is O(nlog n), where n is the size of the array. This is because the algorithm uses merge sort to sort the array, which has a time complexity of O(nlog n), and then performs a linear scan of the array to segregate the positive and negative elements. The linear scan takes O(n) time, but is dominated by the O(nlog n) time complexity of the merge sort.

Therefore, the overall time complexity of the algorithm is O(nlog n).



main.c [ff] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
printArray(int arr[], int n) : void
Management
Projects Files FSy
Workspace
ff
Sources
main.c
C:\Users\LENOVO\Downloads\ff\bin\Debug\ff.exe
Logs & others
Code::Blocks
Debugger
DoxyBlocks
C:\Users\LENOVO\Downloads\ff\main.c
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 92, Col 2, Pos 2165
Insert
Read/Write
Type here to search
23°C غائم غالبًا

```c
#include <stdio.h>
#include <stdlib.h>
void printArray(int arr[], int n);
vo
{
```

```
 Please Enter the Size of an Array :  8

 Please Enter the Array Elements  :  9
-3
5
-2
-8
-6
1
3
Given array is
9 -3 5 -2 -8 -6 1 3
Segregate array is
-3 -2 -8 -6 9 5 1 3

Process returned 0 (0x0)   execution time : 20.158 s
Press any key to continue.
```

```
Checking for ex
Set variable: P                                                                                          s\System32;C:\Windows;C:\Windows
\Windows\System32\WindowsPowerShell\v1.0;C:\Windows\System32\OpenSSH;C:\Program Files (x86)\dotnet;C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\Tools\Binn;C:\Program Files
Server\150\Tools\Binn;C:\Program Files\Microsoft SQL Server\150\Tools\Binn;C:\Program Files\Microsoft SQL Server\150\DTS\Binn;C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn;C:
Data Studio\bin;C:\Program Files\Java\jdk-11.0.16\bin;C:\Users\LENOVO\AppData\Local\Microsoft\WindowsApps;\
Executing: "C:\Program Files\CodeBlocks/cb_console_runner.exe" "C:\Users\LENOVO\Downloads\ff\bin\Debug\ff.exe"  (in C:\Users\LENOVO\Downloads\ff\.)
```

comparison

time complexity of the first algorithm is O(n).
time complexity of the second algorithm is O(nlog n).
of course because the algorithm that use recursion is less efficient and more complex than the non recursive one so the first algorithm is better.