LAB REPORT 09

Course Code:

CSC141

Submitted By:

Haider Ali

Registration number:

FA21-BEE-053

Submitted to:

Dr. Muhammad Kaleem

02/06/2022

Lab # 09 Working with Arrays and Pointers

Objectives:

- Learn the basic operations on arrays (e.g. traversal, value swapping, retrieving indices etc)
- Accessing array elements via pointers.
- Practice array operations by implementing simple sorting algorithms.

Reading Task 1: Working with Arrays

Chapter 08 Arrays (pages 269 to 288) from the book: “Let us C” by Yashavant Kanetkar

Reading Task 2: Selection Sort Algorithm

In computer science, selection sort is a sorting algorithm, specifically an in-place comparison sort. It has O(n2) time complexity, making it inefficient on large lists, and generally performs worse than the similar insertion sort. Selection sort is noted for its simplicity, and it has performance advantages over more complicated algorithms in certain situations, particularly where auxiliary memory is limited.

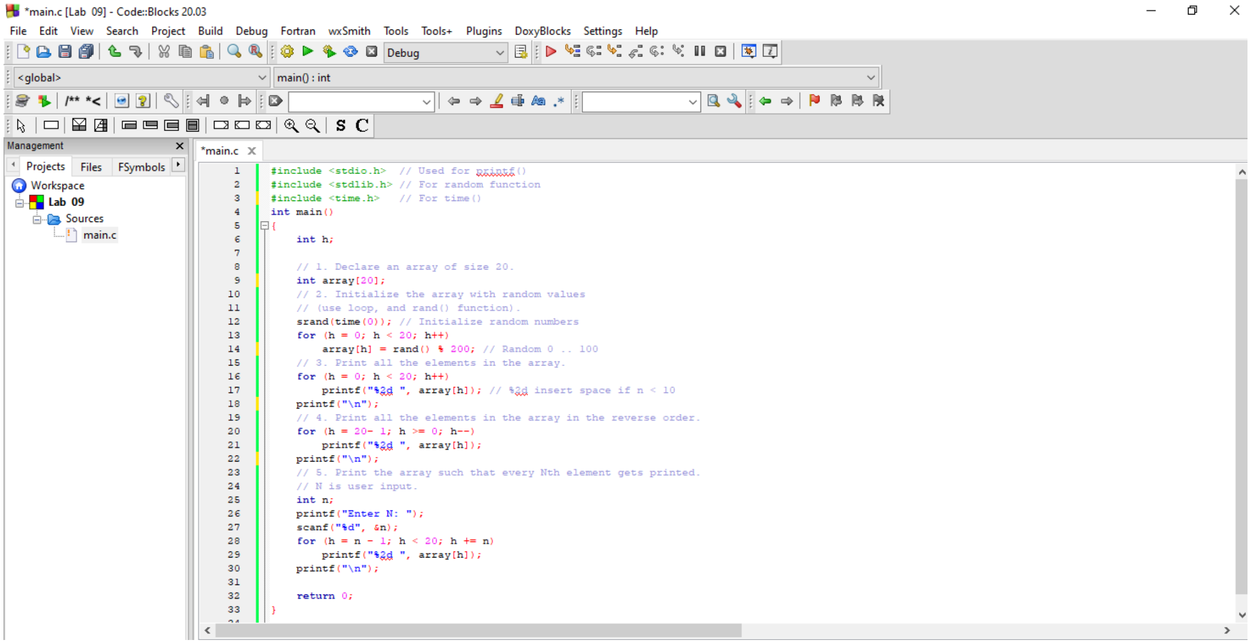
The algorithm divides the input list into two parts: the sublist of items already sorted, which is built up from left to right at the front (left) of the list, and the sublist of items remaining to be sorted that occupy the rest of the list. Initially, the sorted sublist is empty and the unsorted sublist is the entire input list. The algorithm proceeds by finding the smallest (or largest, depending on sorting order) element in the unsorted sublist, exchanging (swapping) it with the leftmost unsorted element (putting it in sorted order), and moving the sublist boundaries one element to the right

In-Lab Task 1: Finding Minimum and Maximum Values in an Array

Your task is to perform some functions on integer arrays. Specifically you will write a C program that does the following:

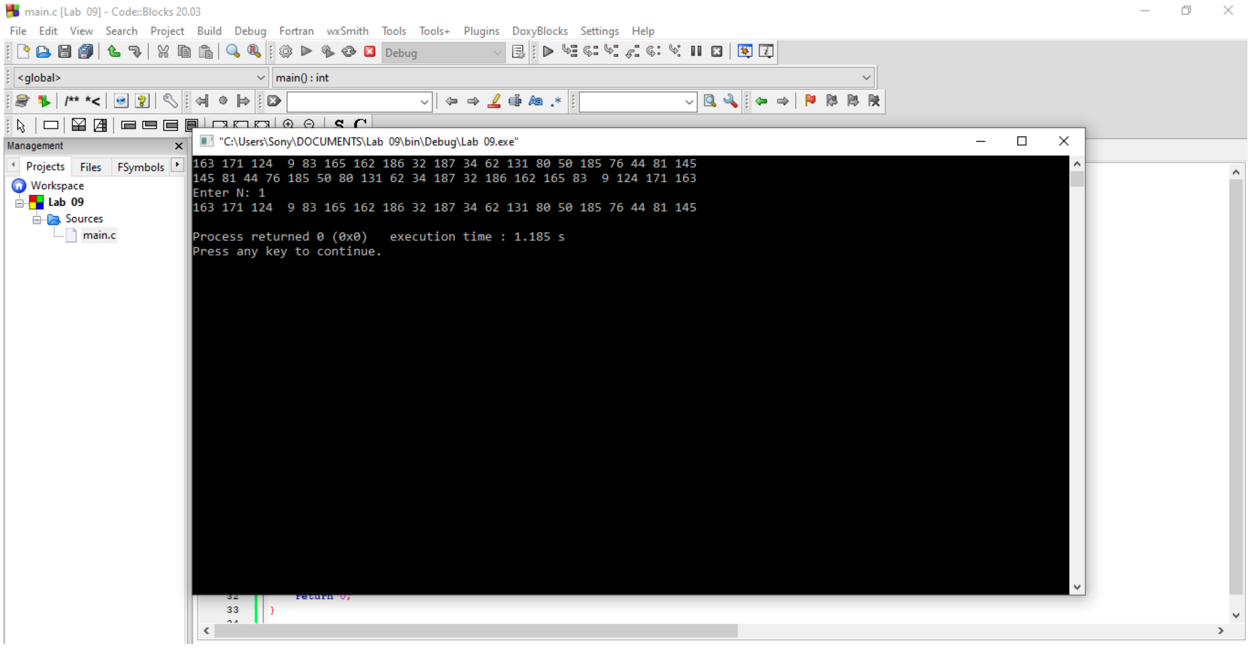
1. Declare an array of size 20.
2. Initialize the array with random values (use loop, and rand() function).
3. Print all the elements in the array.
4. Print all the elements in the array in the reverse order.
5. Print the array such that every Nth element gets printed. N is user input.

```
#include <stdio.h>  // Used for printf()
#include <stdlib.h> // For random function
#include <time.h>   // For time()
int main()
{
    int h;

    //  Declare an array of size 20.
    int array[20];
    //  Initialize the array with random values
    // (use loop, and rand() function).
    srand(time(0)); // Initialize random numbers
    for (h = 0; h < 20; h++)
        array[h] = rand() % 200; // Random 0 .. 100
    //  Print all the elements in the array.
    for (h = 0; h < 20; h++)
        printf("%2d ", array[h]); // %2d insert space if n < 10
    printf("\n");
    // Print all the elements in the array in the reverse order.
    for (h = 20- 1; h >= 0; h--)
        printf("%2d ", array[h]);
    printf("\n");
    //  Print the array such that every Nth element gets printed.
    // N is user input.
    int n;
    printf("Enter N: ");
    scanf("%d", &n);
    for (h = n - 1; h < 20; h += n)
        printf("%2d ", array[h]);
    printf("\n");


    return 0;
}
```


*main.c [Lab 09] - 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 FSymbols
Workspace
Lab 09
Sources
main.c
*main.c

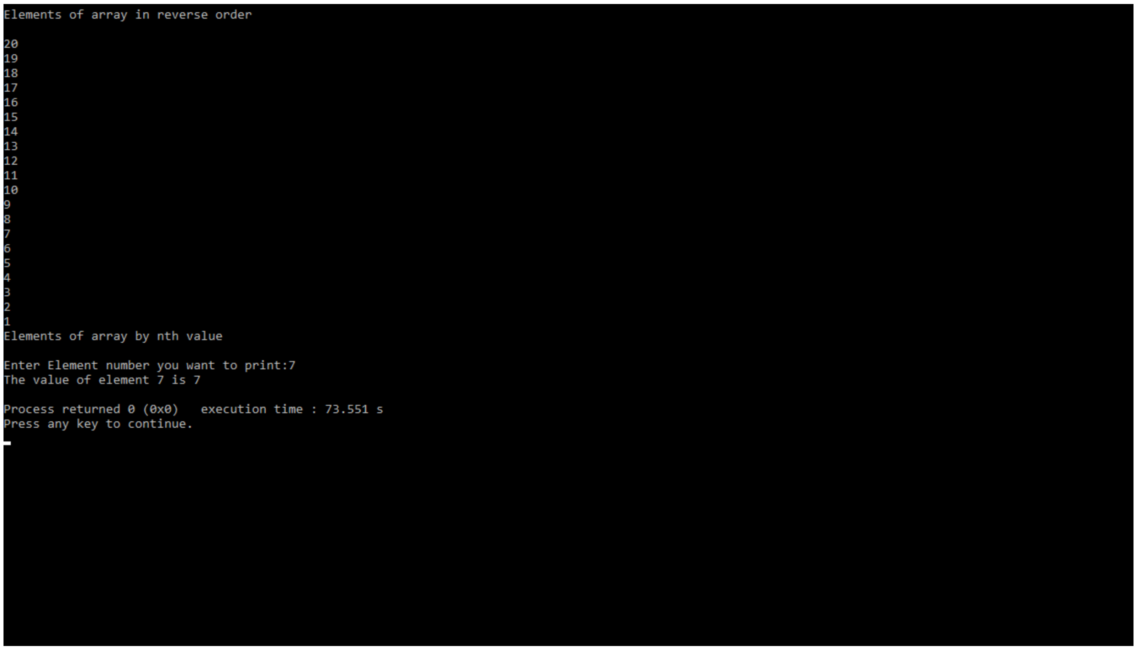
```c
#include <stdio.h>  // Used for printf()
#include <stdlib.h> // For random function
#include <time.h>   // For time()
int main()
{
    int h;

    // 1. Declare an array of size 20.
    int array[20];
    // 2. Initialize the array with random values
    // (use loop, and rand() function).
    srand(time(0)); // Initialize random numbers
    for (h = 0; h < 20; h++)
        array[h] = rand() % 200; // Random 0 .. 100
    // 3. Print all the elements in the array.
    for (h = 0; h < 20; h++)
        printf("%2d ", array[h]); // %2d insert space if n < 10
    printf("\n");
    // 4. Print all the elements in the array in the reverse order.
    for (h = 20- 1; h >= 0; h--)
        printf("%2d ", array[h]);
    printf("\n");
    // 5. Print the array such that every Nth element gets printed.
    // N is user input.
    int n;
    printf("Enter N: ");
    scanf("%d", &n);
    for (h = n - 1; h < 20; h += n)
        printf("%2d ", array[h]);
    printf("\n");

    return 0;
}
```

Output:



main.c [Lab 09] - 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 FSymbols
Workspace
Lab 09
Sources
main.c
"C:\Users\Sony\DOCUMENTS\Lab 09\bin\Debug\Lab 09.exe"

```
163 171 124  9 83 165 162 186 32 187 34 62 131 80 50 185 76 44 81 145
145 81 44 76 185 50 80 131 62 34 187 32 186 162 165 83  9 124 171 163
Enter N: 1
163 171 124  9 83 165 162 186 32 187 34 62 131 80 50 185 76 44 81 145

Process returned 0 (0x0)   execution time : 1.185 s
Press any key to continue.
```

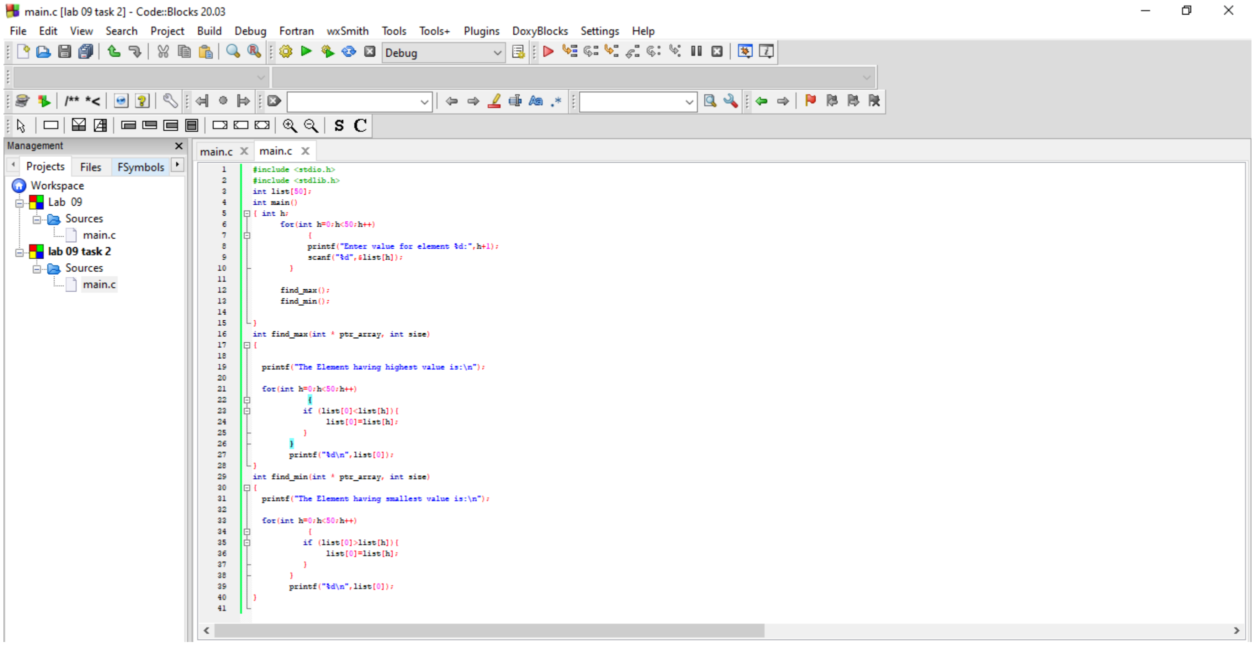


```
Elements of array in reverse order

20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
Elements of array by nth value

Enter Element number you want to print:7
The value of element 7 is 7

Process returned 0 (0x0)   execution time : 73.551 s
Press any key to continue.
```

In-Lab Task 2: Implementing Selection Sort

You are given a C program in **Code Listing 1**, that does the following:
1. Declares an integer array with 50 elements (not initialized).
2. Populates the array with random positive numbers. (Uses a loop and rand() function)
3. Calls the function ‘int **find_max**(int * ptr_array, int size)’ and prints the value and index of the largest number.

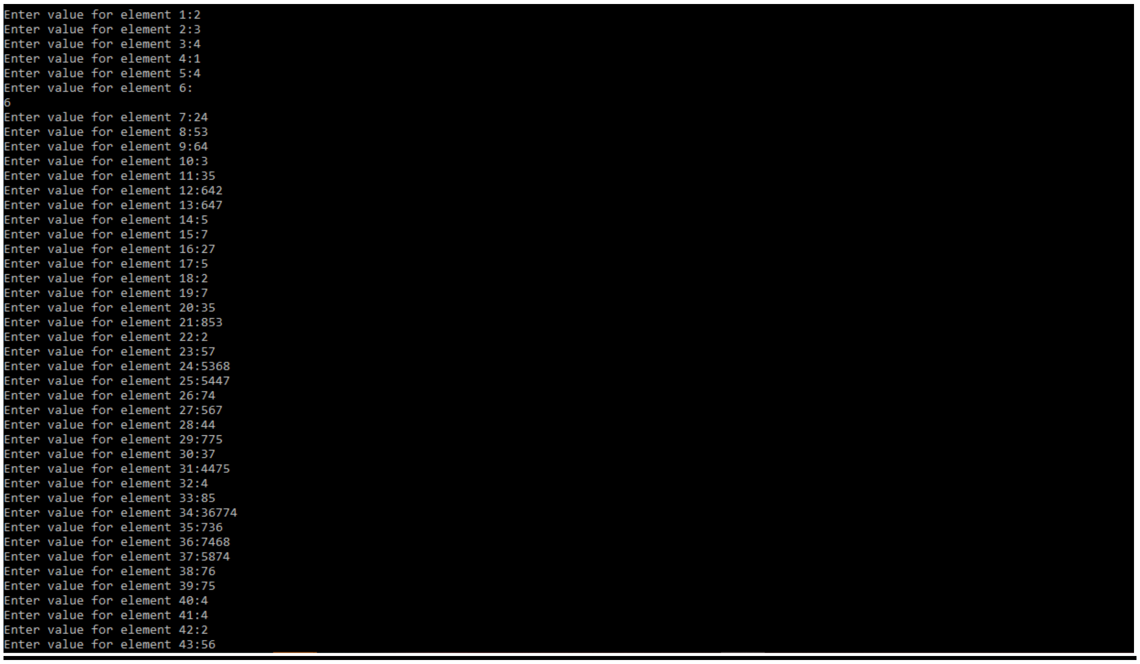
```
#include <stdio.h>

#include <stdlib.h>

int list[50];

int main()

{ int h;

    for(int h=0;h<50;h++)

        {

        printf("Enter value for element %d:",h+1);

        scanf("%d",&list[h]);
        }


      find_max();
      find_min();


}
int find_max(int * ptr_array, int size)
{


  printf("The Element having highest value is:\n");


  for(int h=0;h<50;h++)
            {
           if (list[0]<list[h]){
                list[0]=list[h];
           }
        }
        printf("%d\n",list[0]);
}
int find_min(int * ptr_array, int size)
{
  printf("The Element having smallest value is:\n");


  for(int h=0;h<50;h++)
            {
           if (list[0]>list[h]){
                list[0]=list[h];
           }
        }
        printf("%d\n",list[0]);
}
```



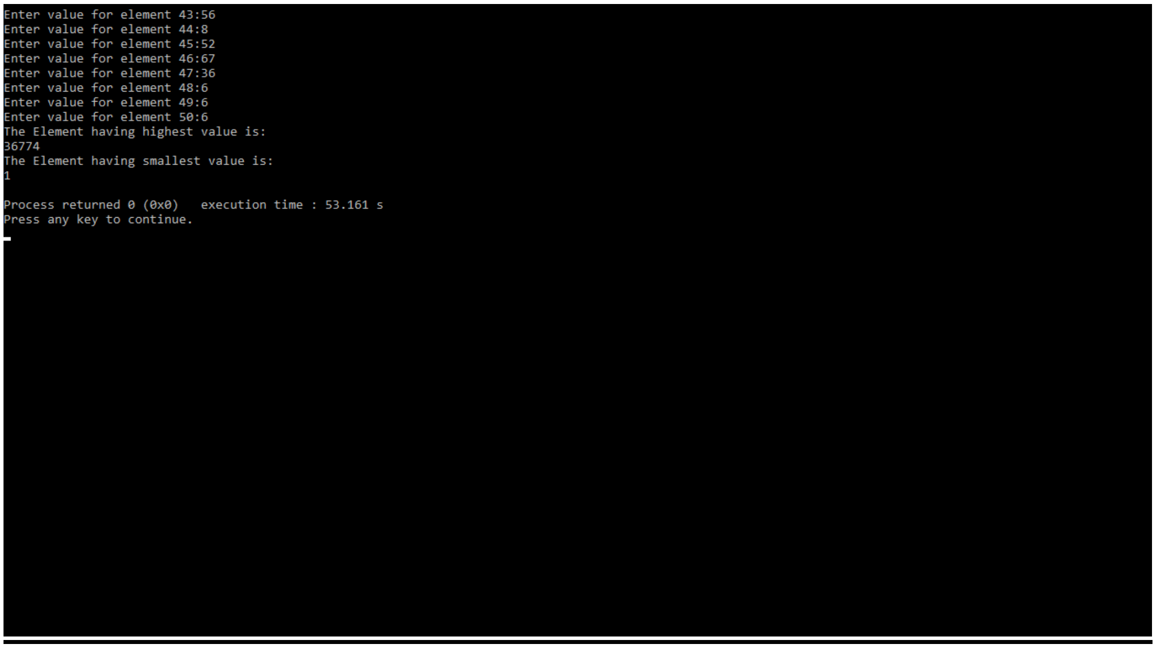
main.c [lab 09 task 2] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Management
Projects Files FSymbols
Workspace
Lab 09
Sources
main.c
lab 09 task 2
Sources
main.c
main.c
main.c

```
#include <stdio.h>
#include <stdlib.h>
int list[50];
int main()
{ int h;
      for(int h=0;h<50;h++)
            {
            printf("Enter value for element %d:",h+1);
            scanf("%d",&list[h]);
        }

      find_max();
      find_min();

}
int find_max(int * ptr_array, int size)
{

  printf("The Element having highest value is:\n");

  for(int h=0;h<50;h++)
            {
           if (list[0]<list[h]){
                list[0]=list[h];
           }
        }
        printf("%d\n",list[0]);
}
int find_min(int * ptr_array, int size)
{
  printf("The Element having smallest value is:\n");

  for(int h=0;h<50;h++)
            {
           if (list[0]>list[h]){
                list[0]=list[h];
           }
        }
        printf("%d\n",list[0]);
}
```

Output:



```
Enter value for element 1:2
Enter value for element 2:3
Enter value for element 3:4
Enter value for element 4:1
Enter value for element 5:4
Enter value for element 6:
6
Enter value for element 7:24
Enter value for element 8:53
Enter value for element 9:64
Enter value for element 10:3
Enter value for element 11:35
Enter value for element 12:642
Enter value for element 13:647
Enter value for element 14:5
Enter value for element 15:7
Enter value for element 16:27
Enter value for element 17:5
Enter value for element 18:2
Enter value for element 19:7
Enter value for element 20:35
Enter value for element 21:853
Enter value for element 22:2
Enter value for element 23:57
Enter value for element 24:5368
Enter value for element 25:5447
Enter value for element 26:74
Enter value for element 27:567
Enter value for element 28:44
Enter value for element 29:775
Enter value for element 30:37
Enter value for element 31:4475
Enter value for element 32:4
Enter value for element 33:85
Enter value for element 34:36774
Enter value for element 35:736
Enter value for element 36:7468
Enter value for element 37:5874
Enter value for element 38:76
Enter value for element 39:75
Enter value for element 40:4
Enter value for element 41:4
Enter value for element 42:2
Enter value for element 43:56
```



```
Enter value for element 43:56
Enter value for element 44:8
Enter value for element 45:52
Enter value for element 46:67
Enter value for element 47:36
Enter value for element 48:6
Enter value for element 49:6
Enter value for element 50:6
The Element having highest value is:
36774
The Element having smallest value is:
1

Process returned 0 (0x0)   execution time : 53.161 s
Press any key to continue.
```

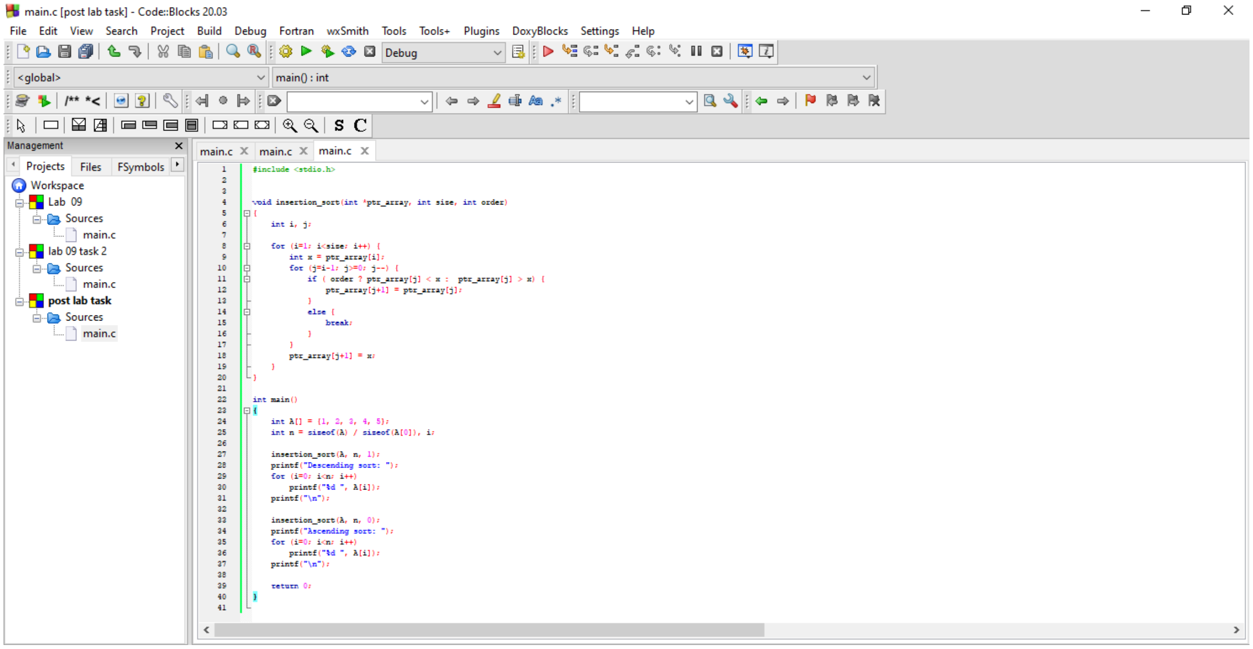
Post-Lab Task: Implement Insertion Sort Algorithm

Your second task is to implement the Insertion Sort algorithm by making a function with the following prototype;

void insertion_sort(int * ptr_array, int size, int order);

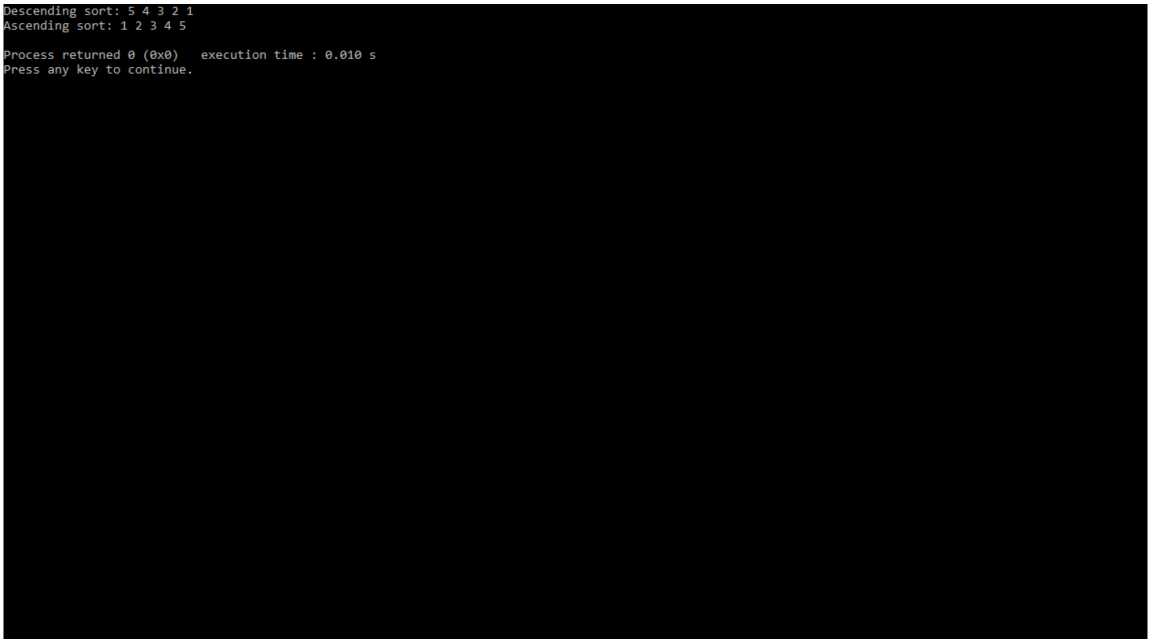
This function takes as input a pointer to the start of the array, and the array size and sorts it in place. The last input to the function is the sorting order (0 for ascending and 1 for descending).

```
#include <stdio.h>

void insertion_sort(int *ptr_array, int size, int order)

{

    int i, j;

    for (i=1; i<size; i++) {

        int x = ptr_array[i];

        for (j=i-1; j>=0; j--) {

            if ( order ? ptr_array[j] < x :  ptr_array[j] > x) {

                ptr_array[j+1] = ptr_array[j];

            }
            else {
                break;
            }
        }
        ptr_array[j+1] = x;
    }
}

int main()
{
    int A[] = {1, 2, 3, 4, 5};
    int n = sizeof(A) / sizeof(A[0]), i;

    insertion_sort(A, n, 1);
    printf("Descending sort: ");
    for (i=0; i<n; i++)
        printf("%d ", A[i]);
    printf("\n");
    insertion_sort(A, n, 0);
    printf("Ascending sort: ");
    for (i=0; i<n; i++)
        printf("%d ", A[i]);
    printf("\n");

    return 0;
}
```



```c
#include <stdio.h>


void insertion_sort(int *ptr_array, int size, int order)
{
    int i, j;

    for (i=1; i<size; i++) {
        int x = ptr_array[i];
        for (j=i-1; j>=0; j--) {
            if ( order ? ptr_array[j] < x :  ptr_array[j] > x) {
                ptr_array[j+1] = ptr_array[j];
            }
            else {
                break;
            }
        }
        ptr_array[j+1] = x;
    }
}

int main()
{
    int A[] = {1, 2, 3, 4, 5};
    int n = sizeof(A) / sizeof(A[0]), i;

    insertion_sort(A, n, 1);
    printf("Descending sort: ");
    for (i=0; i<n; i++)
        printf("%d ", A[i]);
    printf("\n");

    insertion_sort(A, n, 0);
    printf("Ascending sort: ");
    for (i=0; i<n; i++)
        printf("%d ", A[i]);
    printf("\n");

    return 0;
}

```

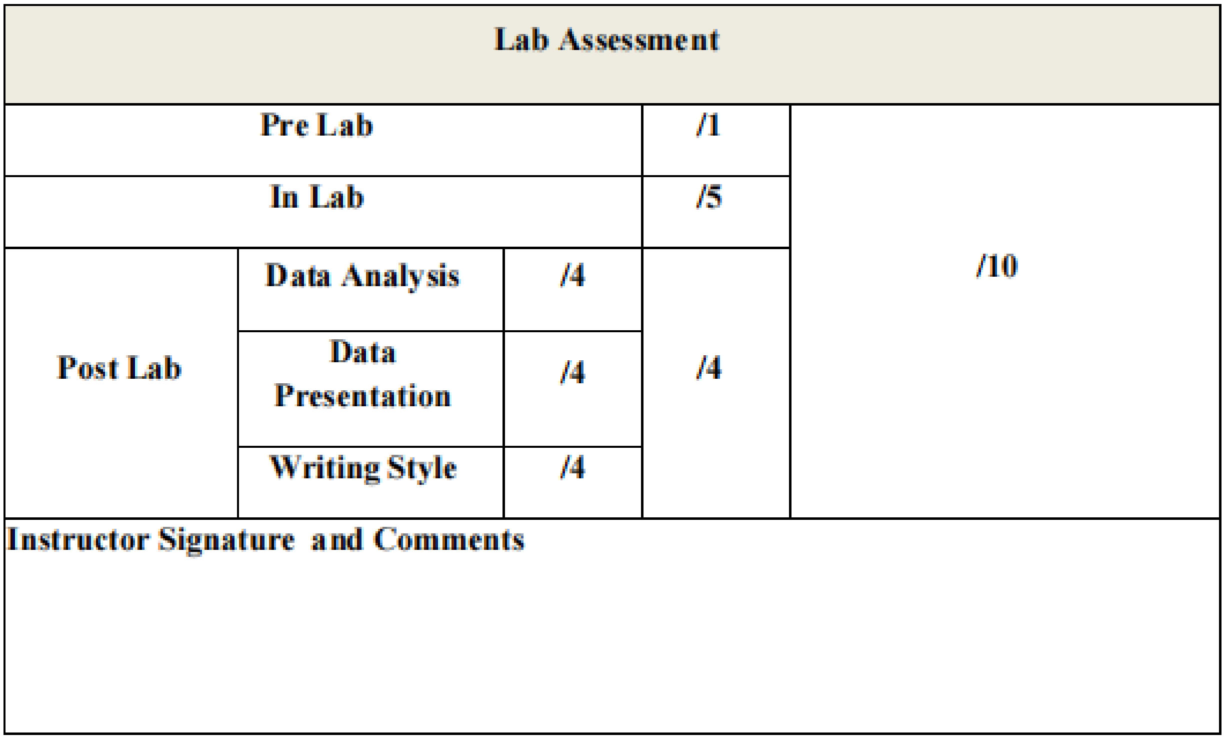
Output:



```
Descending sort: 5 4 3 2 1
Ascending sort: 1 2 3 4 5

Process returned 0 (0x0)   execution time : 0.010 s
Press any key to continue.
```

Critical Analysis:

In this lab 9 we learned how to declare arrays using pointers with this we did in lab task 1 in which we declared 20 arrays with random values we used for loop and random functions to write that program we printed all the arrays elements on the console along with reverse order we also printed the Nth value of arrays means that value that we wanted to print by just writing the array num and it printed the value of that array. Similarly in the in lab task 2 we wrote the program in which we declared 50 array elements and printed on the console the main purpose of that program was to find the maximum from those 50 arrays elements values and we did that by simply using For loop for that also.



Lab Assessment				
Pre Lab			/1	/10
In Lab			/5	
Post Lab	Data Analysis	/4	/4	
	Data Presentation	/4		
	Writing Style	/4		
Instructor Signature and Comments				