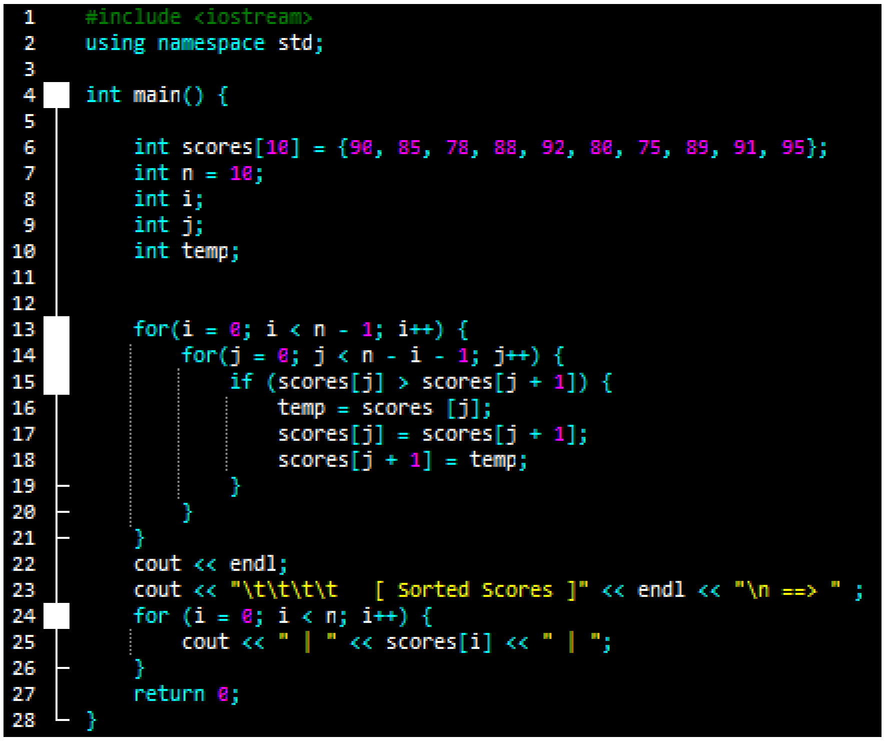
Activity No. 4.2	
Bubble Sort	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: 09/11/25**
Section: CPE11S1	**Date Submitted: 09/11/25**
Name(s): James Daniel M. Verano	**Instructor: Engr. Jimlord M. Quejado**
6. Output	

Code:



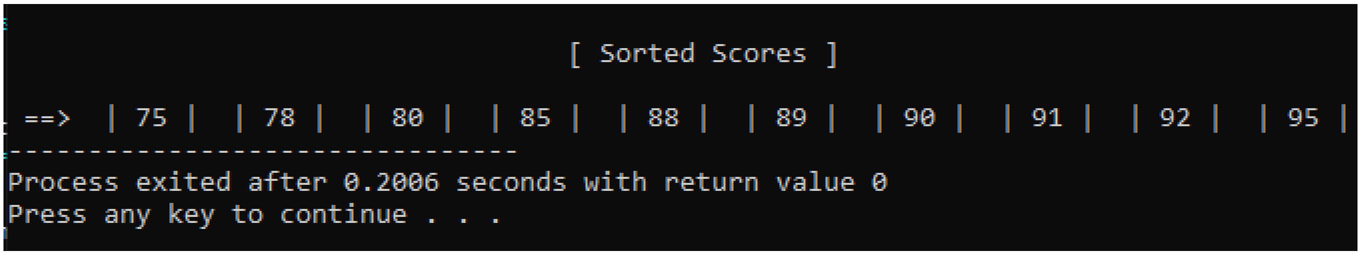
```
#include <iostream>
using namespace std;

int main() {

    int scores[10] = {90, 85, 78, 88, 92, 80, 75, 89, 91, 95};
    int n = 10;
    int i;
    int j;
    int temp;


    for(i = 0; i < n - 1; i++) {
        for(j = 0; j < n - i - 1; j++) {
            if (scores[j] > scores[j + 1]) {
                temp = scores [j];
                scores[j] = scores[j + 1];
                scores[j + 1] = temp;
            }
        }
    }
    cout << endl;
    cout << "\t\t\t\t   [ Sorted Scores ]" << endl << "\n ==> " ;
    for (i = 0; i < n; i++) {
        cout << " | " << scores[i] << " | ";
    }
    return 0;
}
```

Output:



```
                                           [ Sorted Scores ]

 ==>   | 75 |   | 78 |   | 80 |   | 85 |   | 88 |   | 89 |   | 90 |   | 91 |   | 92 |   | 95 |
--------------------------------
Process exited after 0.2006 seconds with return value 0
Press any key to continue . . .
```

How does the code work?

- The first and common part of the code, as we all know, is the declaration of the arrays that you want to sort. We also need a temporary variable to process our swapping arrays in the process of looping the arrays in the sorting stage. Thus, we declare *[i]* and *[j]*, The value that will be used for printing all arrays and the swapping of arrays, and *[n]* is the amount of the loops it will run to complete the sorting stage.

- In the first part of the code, which is the first for loop, This function is to process how many passes it makes through the array, given that *[n]* is declared to 10, n - 1 means we loop 9 times, and the sorting should be done by then.
- For the second part of the code, which is the second for loop function, This processes the comparison and the swapping of the element in each pass it makes in the first for loop function.
- inside the second for loop function, there is a condition function wherein the swapping function starts.
- The IF statement represents that, IF *scores[j]* Also known as the first INDEX of the array (arr*[0]),* is GREATER THAN the second INDEX of the array *(arr[1])*, declared in *scores[j + 1]*, THEN We will proceed to the condition codes it has below.
- IF the condition is met in the IF statement, the program will start to process the first condition code, *temp* = *scores[j]*. This temporarily inputs the GREATER value to the temp variable.
- The next part of the condition code, which is the *scores[j]* = *scores[j +1].* This line code swaps the value of the LESS THAN value to the recent GREATER THAN value variable which was the *scores[j].*
- Once the swapping has been implemented, the last line code is the transferring of the temp variable value, also known as the GREATER THAN value, to the LESS THAN value's variable *scores[j+1]* to move up to the array.
- And the function repeats to the first for loop function until the last value is already the GREATEST value in the array.
- Once the sorting stage is done, The last functioning code is the for loop function of printing all sorted arrays, which is represented as the code, *for (i = 0; i < n; i++).*
- And that's the end of the code!

7. Supplementary Activity

8. Conclusion

The bubble sort is one of the looping function codes in sorting arrays that consistently needs deep understanding on how the process of swapping and transferring works. The comparison and the swapping of arrays is the main core of the code, which is the second for loop function. On the other hand, the first for loop function is to objectively order the amount of looping that it will undergo in the process of sorting the second for loop function. Thus, without the first part of the function, the code will never work. In conclusion, even though the main core exists, without the order, the sorting will never be sorted. With this activity, it helped me to critically analyze how bubble sorting works in certain line codes. I analyzed that in the bubble sorting, each line code is reliant and dependent on one another. Bubble sorting is quite helpful in the process of organizing data and arrays. It is quite complex but once you understand the process of it, it will help you in data management system.

9. Assessment Rubric