Assignment 4.2	
Bubble Sort	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed:** September 9, 2025
Section: CPE11S1	**Date Submitted:** September 10, 2025
Name(s): Mendoza, Nathaniel B.	**Instructor: Engr. Jimlord M. Quejado**

6. Output

Self Made Problem / Sample : Working Bubble Sort Implementation Code and Output
Receiving a list of ages among 15 different people: *10, 14, 78, 93, 53, 74, 33, 7, 43, 18, 27, 59, 91, 66, 81.*
Goal : Create an Ascending and Descending Order using Bubble Sort

1. Code using Bubble Sort in Ascending Order

[*] BubbleSort.cpp

```
1   //MENDOZA, NATHANIEL B. CPE11S1 BUBBLE SORT
2   #include <iostream>
3   using namespace std;
4   int main () {
5       int n = 15;
6       int temp;
7       int ages[n] = {10, 14, 78, 93, 53, 74, 33, 7, 43, 18, 27, 59, 91, 66, 81};
8
9       cout<<"\nList of ages among 15 people: ";
10      for (int i = 0; i < 15; i++){
11      cout<<ages[i]<<" ";
12      }
13      cout<<endl;
14
15      for (int i = 0; i < n - 1; i++) {
16          for (int j = 0; j < n - i - 1; j++) {
17              if (ages[j] > ages[j+1]) {
18                  temp = ages[j];
19                  ages[j] = ages[j+1];
20                  ages[j+1] = temp;
21              }
22          }
23      }
24
25      cout<<"Ages in Ascending Order: ";
26      for (int i = 0; i < n; i++){
27          cout<<ages[i] << " ";
28      }
29      cout<<endl;
30
31      }
```

Code Output:

```
C:\Users\NathanieL\Documents\BubbleSort.exe

List of ages among 15 people: 10 14 78 93 53 74 33 7 43 18 27 59 91 66 81
Ages in Ascending Order: 7 10 14 18 27 33 43 53 59 66 74 78 81 91 93

--------------------------------
Process exited after 0.09849 seconds with return value 0
Press any key to continue . . .
```

2. Code using Bubble Sort in Descending Order

BubbleSort.cpp

```cpp
//MENDOZA, NATHANIEL B. CPE11S1 BUBBLE SORT
#include <iostream>
using namespace std;
int main () {
    int n = 15;
    int temp;
    int ages[n] = {10, 14, 78, 93, 53, 74, 33, 7, 43, 18, 27, 59, 91, 66, 81};

    cout<<"\nList of ages among 15 people: ";
    for (int i = 0; i < 15; i++){
    cout<<ages[i]<<" ";
    }
    cout<<endl;

    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (ages[j] < ages[j+1]) {
                temp = ages[j];
                ages[j] = ages[j+1];
                ages[j+1] = temp;
            }
        }
    }

    cout<<"Ages in Descending Order: ";
    for (int i = 0; i < n; i++){
        cout<<ages[i] << " ";
    }
    cout<<endl;

    }
```

Code Output:

```
C:\Users\NathanieL\Documents\BubbleSort.exe

List of ages among 15 people: 10 14 78 93 53 74 33 7 43 18 27 59 91 66 81
Ages in Descending Order: 93 91 81 78 74 66 59 53 43 33 27 18 14 10 7

--------------------------------
Process exited after 0.1029 seconds with return value 0
Press any key to continue . . .
```

A comprehensive discussion of how bubble sort works

Bubble sort is a simple way to sort numbers by always checking two numbers beside each other and swapping them if they're in the wrong order. After each round, the biggest number goes to the end, and this keeps repeating until the list is sorted. It's easy to understand but kind of slow if you have a lot of data.

Additional Information for Video Lectures I watched.
Bubble Sort works by repeatedly comparing adjacent elements and swapping them if they are in the wrong order, which allows smaller elements to "bubble up" toward the start of the list (Bro Code, 2021). He also concluded that it is bad for a large set of data. Bubble sorts can be understood as a process of pushing the largest value to the end of the list during each pass, which is why the algorithm gradually sorts the list from the back to the front (Sambol, 2016).
In practice, Bubble Sort is often demonstrated using nested loops in C++, where the outer loop controls the passes and the inner loop handles the comparisons and swaps between adjacent elements (Portfolio Courses, 2023).

In terms of changing the bubble sort into an ascending to descending order or descending to ascending order a lecture concludes that just change the sign ">" or "<" into its opposite if you started whether on Ascending/Descending order. Additionally, change the printed text to make it understandable.

BubbleSort.cpp

```cpp
//MENDOZA, NATHANIEL B. CPE11S1 BUBBLE SORT
#include <iostream>
using namespace std;
int main () {
    int n = 15;
    int temp;
    int ages[n] = {10, 14, 78, 93, 53, 74, 33, 7, 43, 18, 27, 59, 91, 66, 81};

    cout<<"\nList of ages among 15 people: ";
    for (int i = 0; i < 15; i++){
    cout<<ages[i]<<" ";
    }
    cout<<endl;

    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (ages[j] > ages[j+1]) {
                temp = ages[j];
                ages[j] = ages[j+1];
                ages[j+1] = temp;
            }
        }
    }

    cout<<"Ages in Ascending Order: ";
    for (int i = 0; i < n; i++){
        cout<<ages[i] << " ";
    }
    cout<<endl;

    }
```

CHANGE

Citations :
Bro Code. (2021, May 24). *Learn Bubble Sort in 7 minutes* 🤿 [Video]. YouTube. https://www.youtube.com/watch?v=Dv4qLJcxus8

Michael Sambol. (2016, July 26). Bubble sort in 2 minutes [Video]. YouTube. https://www.youtube.com/watch?v=xli_FI7CuzA

Portfolio Courses. (2023, May 4). Bubble Sort | C++ Example [Video]. YouTube. https://www.youtube.com/watch?v=62Ai0p1xUpE

Extra Reference : CPE Module 4 Part 1

7. Supplementary Activity

8. Conclusion

Today, I realized that even though it's one of the simplest sorting methods, it really helps me understand the basics of how sorting works step by step. I learned that it just compares numbers side by side, swaps them if needed, and keeps repeating until everything is in order. I also found out that it's easy to switch between ascending and descending by just changing the comparison sign. Overall, it's not the fastest method for big data, but it's a good starting point to understand sorting algorithms better. Though the understanding of the code may be confusing and complicated. But, it will be much easier once you have done it.