



File

Edit

Selection

View

Go

Run

...

200650U.cpp - CS2023-Data-Structures-and-Algorithms---Workspace - Visual Studio Code

EXPLORER

CS2023-DATA-STRUCTURES-AND-ALGORIT...

.vscode

in-class-lab3

200650U.cpp

200650U.exe

lab1\_introduction\_to\_cpp

200650U.cpp X

in-class-lab3 > 200650U.cpp > ...

```

1  #include <iostream>
2  #include <ctime>
3  #include <cstdlib>
4
5  using namespace std;
6
7  const int max_length = 20000;
8
9  void swap(int &a, int &b) {
10     int temp = a;
11     a = b;
12     b = temp;
13 }
14
15 void insertionSort(int arr[], int n) {
16     for (int i = 1; i < n; i++) {
17         int key = arr[i];
18         int j = i - 1;
19         while (j >= 0 && arr[j] > key) {
20             arr[j + 1] = arr[j];
21             j--;
22         }
23         arr[j + 1] = key;
24     }
25 }
26
27 void bubbleSort(int arr[], int n) {
28     for (int i = 0; i < n - 1; i++) {
29         for (int j = 0; j < n - i - 1; j++) {
30             if (arr[j] > arr[j + 1]) {
31                 swap(arr[j], arr[j + 1]);
32             }
33         }
34     }
35 }

```

Shajini Majuran

Shajini Majuran

Shajini Majuran

Shajini Majuran

OUTLINE

TIMELINE

Ln 14, Col 1

Spaces: 4

UTF-8

CRLF

C++

**Insertion sort:**

Best case:  $O(n)$

Worst case:  $O(n^2)$

Average case:  $O(n^2)$

**Bubble sort:**

Best case:  $O(n)$

Worst case:  $O(n^2)$

Average case:  $O(n^2)$

**Optimized bubble sort:**

Best case:  $O(n)$

Worst case:  $O(n^2)$

Average case:  $O(n^2)$

**Selection sort:**

Best case:  $O(n^2)$

Worst case:  $O(n^2)$

Average case:  $O(n^2)$

Note that the best-case time complexity is when the input array is already sorted, and the worst case time complexity is when the input array is in reverse order. The average case time complexity approximates the time complexity for random inputs.

Again note that from the above graph, Insertion sort performs best and Bubble sort performs worst.

Among the four sorting algorithms mentioned, Insertion sort has the best-case time complexity of  $O(n)$  which means that if the input array is already sorted or almost sorted, insertion sort will perform optimally. Bubble sort has the same best-case time complexity as insertion sort, but its worst-case and average-case time complexities are  $O(n^2)$ , making it inefficient for large arrays. Optimized bubble sort is a variation of bubble sort with a slight optimization, but it still has the same time complexity as bubble sort in both the worst-case and average-case scenarios.