Bubble Sort:

Bubble sort is a sorting algorithm that starts from the first element of an array and compares it with the second element. If the first element is greater than the second, we swap them. It continues this process until the end of the array, with the largest elements "bubbling" to the top. Its worst-case time complexity is $O(n^2)$, and its best-case time complexity is O(n).

Each time after an element moves from the unsorted part to the sorted part one sort pass is completed.

For a list of n elements, n-1 sort passes are required (at most).

Algorithm Python Code Snippet:

```python
def bubble_sort(arr):
    for i in range(0, len(arr)):
        is_sorted = True
        for j in range(0, len(arr) - 1 - i):
            if(arr[j] > arr[j+1]):
                is_sorted = False
                arr[j], arr[j+1] = arr[j+1], arr[j]
        if is_sorted == True:
            break
    return arr
```

Time Complexity:

$$O(n^2)$$

Algorithm Complexity Explained:

Best Case →Sorted Array

Number of moves: 0 $\boldsymbol{O(1)}$

Number of comparisons: n-1 $\boldsymbol{O(n)}$

Worst Case →Reversed Array

Outer loop → Executed n-1 times

Number of moves: 3*(1+2+...+n-1) = 3 * n*(n-1)/2 $\boldsymbol{O(n^2)}$

Number of comparisons: 1+2+...+n-1 = n*(n-1)/2 $\boldsymbol{O(n^2)}$

Average Case → $\boldsymbol{O(n^2)}$