1.8 BUBBLE SORT

Question:

Sort an array of integers using the bubble sort technique. Analyze its time complexity using Big-O notation. Write the code.

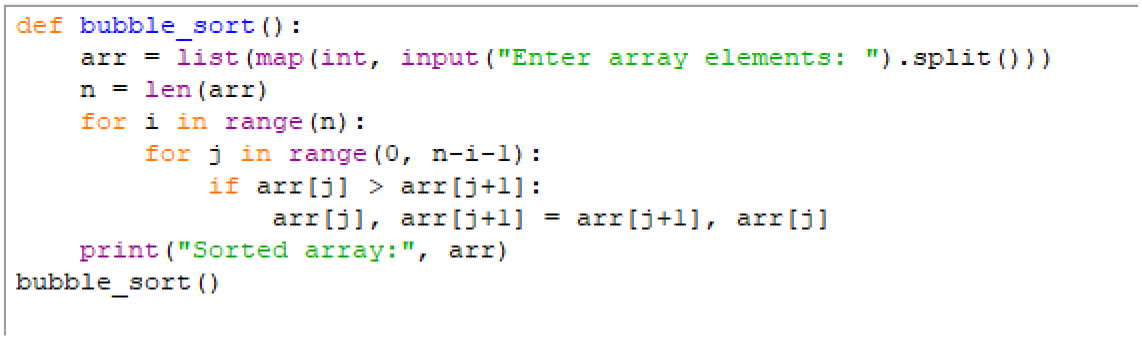
AIM:

To sort an array of integers using the bubble sort technique and analyze its time complexity using Big-O notation.

ALGORITHM:

1. Start with a list of unsorted elements.
2. Iterate through the list from the first element to the last.
3. For each element, compare it with the next element.
4. If the current element is greater than the next element, swap them.
5. Repeat steps 2-4 until the list is sorted.
6. The largest element will "bubble" to the end of the list after the first pass.
7. The process is repeated for the remaining unsorted elements until the entire list is sorted.

PROGRAM:

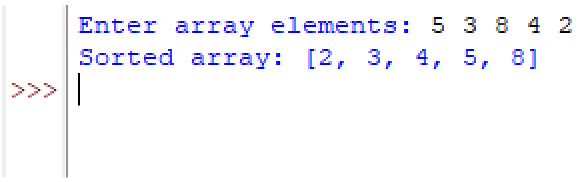


```python
def bubble_sort():
    arr = list(map(int, input("Enter array elements: ").split()))
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
    print("Sorted array:", arr)
bubble_sort()
```

Input:

5 3 8 4 2

Output:



```
    Enter array elements: 5 3 8 4 2
    Sorted array: [2, 3, 4, 5, 8]
>>> 
```

RESULT:

Thus the program is successfully executed, and the output is verified.

PERFORMANCE ANALYSIS:

- Worst-Case and Average-Case Time Complexity: O(n^2)
- Best-Case Time Complexity: O(n)
- Space Complexity: O(1) – Bubble Sort is an in-place algorithm.