Week -2 Homework-1

Name: Himavanth reddy putta Id:18759

Q28) What is the Big-O Time Complexity Analysis of BubbleSort? LC

* Process
  + Step 1: Please use [Loop Analysis](https://hc.labnet.sfbu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/analyze_loop.html) method to analyze the function

void bubbleSort(int arr[])

Please explain your answer.

class BubbleSort

{

void bubbleSort(int arr[])

{

int n = arr.length;

for (int i = 0; i < n-1; i++) **=> O(n)\*O(n)=O(n2)**

for (int j = 0; j < n-i-1; j++) **=>O(n)**

if (arr[j] > arr[j+1])

{

// swap arr[j+1] and arr[i]

int temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = temp;

}

}

/\* Prints the array \*/

void printArray(int arr[])

{

int n = arr.length;

for (int i=0; i<n; ++i) **=>O(n)**

System.out.print(arr[i] + " ");

System.out.println();

}

// Driver method to test above

public static void main(String args[])

{

BubbleSort ob = new BubbleSort();

int arr[] = {64, 34, 25, 12, 22, 11, 90};

ob.bubbleSort(arr);

System.out.println("Sorted array");

ob.printArray(arr);

}

}

Big O = O(n2)+O(n) { as n2 dominates}

Big O = O(n2)

My explanation:

->Outer loop in First loop Runs n-1 times, from 0 to n-1.

->An inner loop that executes n-i-1 times for every time the outer loop