Algorithm HW

1. What is the Big-O Time Complexity Analysis of BubbleSort?

Please use [Loop Analysis](https://npu85.npu.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++)

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

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)

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);

}

}

https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/exercise\_geeksforgeeks.html  
Q29 ==> What is the Big-O Time Complexity Analysis of Linear Search?

What is the Big-O Time Complexity Analysis of Linear Search?

Please use [Loop Analysis](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/analyze_loop.html) method to analyze

public static int search(int arr[], int x)

Please explain your answer.

// Java code for linearly search x in arr[]. If x

// is present then return its location, otherwise

// return -1

class GFG

{

public static int search(int arr[], int x)

{

int n = arr.length;

for(int i = 0; i < n; i++)

{

if(arr[i] == x)

return i;

}

return -1;

}

public static void main(String args[])

{

int arr[] = { 2, 3, 4, 10, 40 };

int x = 10;

int result = search(arr, x);

if(result == -1)

System.out.print("Element is not present in array");

else

System.out.print("Element is present at index " + result);

}

}

Time Complexity Analysis Questions [.](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/hw/q41)

|  |  |
| --- | --- |
| [**Question 1**](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/Example)  What is the time complexity of following code:  int a = 0;  for (i = 0; i < N; i++) {  for (j = N; j > i; j--) {  a = a + i + j;  }  }  Options:  1. O(N)  2. O(N\*log(N))  3. O(N \* Sqrt(N))  4. O(N\*N)    [**Question 2**](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/Example)  What is the time complexity of following code:  int i, j, k = 0;  for (i = n / 2; i <= n; i++) {  for (j = 2; j <= n; j = j \* 2) {  k = k + n / 2;  }  }  Options:  1. O(n)  2. O(nLogn)  3. O(n^2)  4. O(n^2Logn) | [**Question 3**](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/Example)  What does it mean when we say that an algorithm X is asymptotically more efficient than Y?  Options:  1. X will always be a better choice for small inputs  2. X will always be a better choice for large inputs  3. Y will always be a better choice for small inputs  4. X will always be a better choice for all inputs    [**Question 4**](https://npu85.npu.edu/~henry/npu/classes/algorithm/geeksforgeeks/slide/Example)  What is the time complexity of following code:  int a = 0, i = N;  while (i > 0) {  a += i;  i /= 2;  }  Options:  1. O(N)  2. O(Sqrt(N))  3. O(N / 2)  4. O(log N) |