Home task :

Task 10  rewrite the code in such a way that it has to take unsorted list and then ,merge in an array the sorted list.

import java.util.Arrays;  
  
public class Example {  
 public static void main(String[] args) {  
 // Step 1: Unsorted arrays  
 int[] arr1 = {75, 11, 34, 66}; // Unsorted  
 int[] arr2 = {100, 1, 19, 50, 5, 89}; // Unsorted  
  
 int n1 = arr1.length;  
 int n2 = arr2.length;  
  
 // Step 2: Sort both arrays  
 Arrays.*sort*(arr1);  
 Arrays.*sort*(arr2);  
  
 // Step 3: Create merge array  
 int[] merge = new int[n1 + n2];  
  
 // Step 4: Merge sorted arrays  
 int i = 0, j = 0, k = 0;  
 while (i < n1 && j < n2) {  
 if (arr1[i] < arr2[j]) {  
 merge[k++] = arr1[i++];  
 } else {  
 merge[k++] = arr2[j++];  
 }  
 }  
  
 // Step 5: Copy remaining elements  
 while (i < n1) {  
 merge[k++] = arr1[i++];  
 }  
  
 while (j < n2) {  
 merge[k++] = arr2[j++];  
 }  
  
 // Step 6: Print arrays  
 System.*out*.print("Sorted Array 1: ");  
 for (int num : arr1) {  
 System.*out*.print(num + " ");  
 }  
  
 System.*out*.print("\nSorted Array 2: ");  
 for (int num : arr2) {  
 System.*out*.print(num + " ");  
 }  
  
 System.*out*.print("\nMerged Sorted Array: ");  
 for (int num : merge) {  
 System.*out*.print(num + " ");  
 }  
 }  
}

output:

Sorted Array 1: 11 34 66 75

Sorted Array 2: 1 5 19 50 89 100

Merged Sorted Array: 1 5 11 19 34 50 66 75 89 100

Process finished with exit code 0

Hash table advantages and disadvantages

**Advantages of Hash Table**

| **Advantage** | **Explanation** |
| --- | --- |
| **Fast Access (O(1) Average)** | Data can be accessed quickly using the key, usually in constant time. |
| **Efficient Insertion/Deletion** | Add, delete, or update operations are efficient compared to lists or arrays. |
| **Flexible Key Types** | Can store any object as a key (with proper hashCode() and equals() methods). |
| **Good for Large Datasets** | Scales well for handling large amounts of data. |
| **Avoids Duplicates** | No duplicate keys are allowed — helps enforce uniqueness. |

**Disadvantages of Hash Table**

| **Disadvantage** | **Explanation** |
| --- | --- |
| **No Order Guarantee** | Does not maintain insertion or sorted order of keys. |
| **Performance Can Degrade** | Poor hash functions or many collisions can lead to performance dropping to O(n). |
| **Memory Overhead** | Uses extra memory (buckets, linked lists, or probing slots). |
| **Not Thread-safe** | HashMap is not thread-safe unless explicitly synchronized. |
| **Debugging Can Be Hard** | Due to the hashing mechanism and lack of order, harder to trace issues. |