```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(nk) time | O(n + k) space - where where n is the total
  // number of array elements and k is the number of arrays
  public static List<Integer> mergeSortedArrays(List<List<Integer>> arrays) {
    List<Integer> sortedList = new ArrayList<Integer>();
    List<Integer> elementIdxs = new ArrayList<Integer>(Collections.nCopies(arrays.size(), 0));
    while (true) {
      List<Item> smallestItems = new ArrayList<Item>();
      for (int arrayIdx = 0; arrayIdx < arrays.size(); arrayIdx++) {
        List<Integer> relevantArray = arrays.get(arrayIdx);
        int elementIdx= elementIdxs.get(arrayIdx);
        if (elementIdx== relevantArray.size()) continue;
        smallestItems.add(new Item(arrayIdx, relevantArray.get(elementIdx)));
      }
      if (smallestItems.size() == 0) break;
      Item nextItem = getMinValue(smallestItems);
      sortedList.add(nextItem.num);
      elementIdxs.set(nextItem.arrayIdx, elementIdxs.get(nextItem.arrayIdx) + 1);
    }

    return sortedList;
  }

  public static Item getMinValue(List<Item> items) {
    int minValueIdx = 0;
    for (int i = 1; i < items.size(); i++) {
      if (items.get(i).num < items.get(minValueIdx).num) minValueIdx = i;
    }
    return items.get(minValueIdx);
  }

  static class Item {
    public int arrayIdx;
    public int num;

    public Item(int arrayIdx, int num) {
      this.arrayIdx = arrayIdx;
      this.num = num;
    }
  }
}

```