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

using namespace std;

struct Item {
  int arrayIdx;
  int num;
};

Item getMinValue(vector<Item> items);

// O(nk) time | O(n + k) space - where where n is the total
// number of array elements and k is the number of arrays
vector<int> mergeSortedArrays(vector<vector<int>> arrays) {
  vector<int> sortedList;
  vector<int> elementIdxs (arrays.size(), 0);

  while (true) {
    vector<Item> smallestItems;
    for (int arrayIdx = 0; arrayIdx < arrays.size(); arrayIdx++) {
      vector<int> relevantArray = arrays[arrayIdx];
      int elementIdx= elementIdxs[arrayIdx];
      if (elementIdx== relevantArray.size()) continue;
      smallestItems.push_back(Item {arrayIdx, relevantArray[elementIdx]});
    }
    if (smallestItems.size() == 0) break;
    Item nextItem = getMinValue(smallestItems);
    sortedList.push_back(nextItem.num);
    elementIdxs[nextItem.arrayIdx]++;
  }

  return sortedList;
}

Item getMinValue(vector<Item> items) {
  int minValueIdx = 0;
  for (int i = 1; i < items.size(); i++) {
    if (items[i].num < items[minValueIdx].num) minValueIdx = i;
  }
  return items[minValueIdx];
}
```