Program:2a

```
import java.util.*;
public class IntervalListIntersections {
    public static int[][] intervalIntersection(int[][] firstList, int[][] secondList) {
        List<int[]> result = new ArrayList<>();
        int i = 0, j = 0;
             while (i < firstList.length && j < secondList.length) {
            int start = Math.max(firstList[i][0], secondList[j][0]);
            int end = Math.min(firstList[i][1], secondList[j][1]);
                if (start <= end) {
                result.add(new int[] {start, end});
            }
         if (firstList[i][1] < secondList[j][1]) {
                i++;
            } else {
                j++;
            }
}
   return result.toArray(new int[result.size()][]);
    }
public static void main(String[] args) {
        int[][] firstList = {{0, 2}, {5, 10}, {13, 23}, {24, 25}};
        int[][] secondList = {{1, 5}, {8, 12}, {15, 24}, {25, 26}};
        int[][] intersections = intervalIntersection(firstList, secondList);
 System.out.println("Intersections:");
        for (int[] interval : intersections) {
            System.out.println(Arrays.toString(interval));
        }
    }
}
```

Output:

```
Intersections:

[1, 2]

[5, 5]

[8, 10]

[15, 23]

[24, 24]

[25, 25]
```

Program:2b

```
import java.util.Arrays;
public class MergeSortedArray {
  public static void merge(int[] nums1, int m, int[] nums2, int n) {
    int p1 = m - 1;
    int p2 = n - 1;
    int p = m + n - 1;
     while (p1 >= 0 && p2 >= 0) {
      if (nums1[p1] > nums2[p2]) {
        nums1[p] = nums1[p1];
        p1--;
      } else {
        nums1[p] = nums2[p2];
        p2--;
      }
      p--;
    }while (p2 >= 0) {
      nums1[p] = nums2[p2];
      p2--;
      p--;}}
public static void main(String[] args) {
    int[] nums1 = {1, 3, 5, 0, 0, 0};
    int m = 3;
    int[] nums2 = {2, 4, 6};
    int n = 3;merge(nums1, m, nums2, n);
    System.out.println("Merged array: " + Arrays.toString(nums1));
  }}
```

Output:

```
Merged array: [1, 2, 3, 4, 5, 6]
```

Program:2c

```
import java.util.*;


public class ThreeSum {
    public static List<List<Integer>> threeSum(int[] nums) {
        Arrays.sort(nums);
        List<List<Integer>> result = new ArrayList<>();


        for (int i = 0; i < nums.length - 2; i++) {
            if (i > 0 && nums[i] == nums[i - 1]) continue;


            int left = i + 1, right = nums.length - 1;


            while (left < right) {
                int sum = nums[i] + nums[left] + nums[right];


                if (sum == 0) {
                    result.add(Arrays.asList(nums[i], nums[left], nums[right]));
                    left++;
                    right--;



                    while (left < right && nums[left] == nums[left - 1]) left++;


                    while (left < right && nums[right] == nums[right + 1]) right--;
                } else if (sum < 0) {
                    left++;
                } else {
                    right--;
                }
            }
        }
        return result;
    }

    public static void main(String[] args) {
        int[] nums = {-1, 0, 1, 2, -1, -4};
        List<List<Integer>> triplets = threeSum(nums);

        System.out.println("Triplets that sum to zero:");
        for (List<Integer> triplet : triplets) {
            System.out.println(triplet);
        }
    }
}
```

Output:

```
Triplets that sum to zero:
[-1, -1, 2]
[-1, 0, 1]
```