Q.1 :

Sol:

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class LeastIndexSum {
    public static String[] findRestaurant(String[] list1, String[] list2) {
        Map<String, Integer> indexMap = new HashMap<>();
        List<String> result = new ArrayList<>();
        int leastIndexSum = Integer.MAX_VALUE;

        // Populate the index map for list1
        for (int i = 0; i < list1.length; i++) {
            indexMap.put(list1[i], i);
        }

        // Check common strings and update result if the index sum is minimum
        for (int j = 0; j < list2.length; j++) {
            if (indexMap.containsKey(list2[j])) {
                int indexSum = j + indexMap.get(list2[j]);
                if (indexSum < leastIndexSum) {
                    leastIndexSum = indexSum;
                    result.clear();
                    result.add(list2[j]);
                } else if (indexSum == leastIndexSum) {
                    result.add(list2[j]);
                }
```

```
            }
        }

        // Convert the list to an array
        return result.toArray(new String[0]);
    }

    public static void main(String[] args) {
        // Example usage:
        String[] list1 = {"Shogun", "Tapioca Express", "Burger King", "KFC"};
        String[] list2 = {"Piatti", "The Grill at Torrey Pines", "Hungry Hunter Steakhouse", "Shogun"};

        String[] result = findRestaurant(list1, list2);
        System.out.println("Common Strings with Least Index Sum:");
        for (String s : result) {
            System.out.println(s);
        }
    }
}
```

Q.2:

Sol:

```
public class CanPlaceFlowers {
    public static boolean canPlaceFlowers(int[] flowerbed, int n) {
        int count = 0;
        int length = flowerbed.length;

        for (int i = 0; i < length; i++) {
            if (flowerbed[i] == 0) {
                // Check if the current plot and its adjacent plots are empty
```

```
            boolean prevEmpty = (i == 0) || (i > 0 && flowerbed[i - 1] == 0);
            boolean nextEmpty = (i == length - 1) || (i < length - 1 && flowerbed[i + 1] == 0);

            if (prevEmpty && nextEmpty) {
              // Plant a flower at the current plot
              flowerbed[i] = 1;
              count++;

              // Skip the next plot as it cannot have a flower
              i++;
            }
          }
        }

        return count >= n;
    }

    public static void main(String[] args) {
        // Example usage:
        int[] flowerbed = {1, 0, 0, 0, 1};
        int n = 1;

        boolean result = canPlaceFlowers(flowerbed, n);
        System.out.println("Can place " + n + " flowers? " + result);
    }
}
```