```java
public class TimeSpaceComplexityQuestions {

    public static void question1() {
        for (int i = 0; i < n; i++) {
            System.out.println(i);
        }
    }

    public static int question2(int n) {
        if (n == 0) return 0;
        return n + question2(n - 1);
    }

    public static void question3() {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.println(i + j);
            }
        }
    }

    public static void question4(int[] arr) {
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }

    public static void question5(int n) {
        int[] array = new int[n];
        for (int i = 0; i < n; i++) {
            array[i] = i * i;
        }
    }

    public static int question6(int[] arr, int target) {
        int left = 0, right = arr.length - 1;
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (arr[mid] == target) return mid;
            else if (arr[mid] < target) left = mid + 1;
            else right = mid - 1;
        }
        return -1;
    }

    public static void question7(int n) {
        if (n <= 0) return;
        System.out.println(n);
        question7(n - 1);
    }

    public static void question8(int[] arr) {
        Arrays.sort(arr);
    }

    public static void question9(String[] words) {
        HashMap<String, Integer> wordCount = new HashMap<>();
        for (String word : words) {
            wordCount.put(word, wordCount.getOrDefault(word, 0) + 1);
        }
    }

    public static int question10(int n, Map<Integer, Integer> memo) {
        if (n <= 1) return n;
        if (!memo.containsKey(n)) {
            memo.put(n, question10(n - 1, memo) + question10(n - 2, memo));
        }
        return memo.get(n);
    }
}
```