01_MeetingRooms

```
class Interval{
        int start;
        int end;
        Interval(){
                this.start = 0;
                this.end =0;
        }
        Interval(int s, int e){
                this.start = s;
                this.end = e;
        }
}

public class MeetingRooms {
    public boolean solve(Interval[] intervals) {
    }
}
```

02_MoveZeros

```
public class MoveZeros{
    public void moveZeroes(int[] nums) {
    }
}
```

03_TwoSum

```
public class TwoSum{

    public int[] twoSum(int[] nums, int target) {
    }
}
```

04_DailyTemperature

```
public class DailyTemperature {

    public int[] dailyTemperatures(int[] temperatures) {
    }
}
```

05_MergeInterval

```
class Interval {
        int start;
        int end;

        Interval() {
                start = 0;
                end = 0;
        }

        Interval(int s, int e) {
                start = s;
                end = e;
        }
}
public class MergeInterval {

    public List<Interval> merge(List<Interval> intervals) {
     }
}
```

06_MeetingRoom2

```
class Interval {
        int start;
        int end;

        Interval() {
                start = 0;
                end = 0;
        }

        Interval(int s, int e) {
                start = s;
                end = e;
        }
}
public class MeetingRoom2 {

    public int solve(Interval[] intervals)  {
    }
}
```

07_JewelsAndStones

```
public class JewelsAndStones {

    public int solve(String jew, String stone)  {
    }
}
```

08_LicenseKeyFormatting

```
public class LicenseKeyFormatting {

    public String licenseKeyFormatting(String S, int K)   {
    }
}
```

09_KClosest

```
public class KClosest {

    public int[][] kClosest(List<List<Integer>> list, int K) {
    }
}
```

10_PlusOne

```
public class PlusOne  {

    public int[] plusOne(int[] digits) {{

    }
}
```

11_UniqueEamilAddress

```
public class UniqueEamilAddress {

    public int numUniqueEmails(String[] emails){

    }
}
```

12_LongestSubMostTwoDist

```
public class LongestSubMostTwoDist {

    public int lengthOfLongestSubstringTwoDistinct(String s) {

    }
}
```

13_MaxSubArray

```
public class MaxSubArray {

    public int maxSubArray(int[] nums)  {

    }
}
```

14_FindAnagramMapping

```
public class FindAnagramMapping{

    public int[] anagramMappings(int[] A, int[] B) {

    }
}
```

15_FindAllAnagrams

```
public class FindAllAnagrams {

    public List<Integer> findAnagrams(String txt, String pat) {{

    }
}
```

16_SpiralMatrix

```
public class SpiralMatrix  {
    public static void main(String[] args) {
            int[][] matrix = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
            System.out.println(solve(matrix));
    }

   public static List<Integer> solve(int[][] matrix) {

   }
}
```

17_GroupAnagrams

```
public class GroupAnagrams  {
     public static void main(String[] args) {

               String[] list =  {"eat", "tea", "tan", "ate", "nat", "bat"};
               System.out.println(groupAnagrams(list));
     }
     public static List<List<String>> groupAnagrams(String[] strs) {
     }
}
```

18_TrappingRainWater

```
public class TrappingRainWater  {
     public static void main(String[] args) {

               String[] list =  {"eat", "tea", "tan", "ate", "nat", "bat"};
               System.out.println(groupAnagrams(list));
     }
     public static List<List<String>> groupAnagrams(String[] strs) {
     }
}
```

19_KthLargest

```
public class KthLargest  {
      public static void main(String[] args) {
                        KthLargest a = new KthLargest();
                        int[] nums = {3,2,1,5,6,4};
                        int k =2;
                        System.out.println(a.solve(nums , k));
      }
      public int solve(int[] nums , int k) {
      }
}
```

20_MissingRanges

```
public class MissingRanges   {
      public static void main(String[] args) {
            int[] nums = {2,3,5,50,75};
            int lower=0, upper=99;
            System.out.println(solve(nums, lower, upper));
//          [0->1, 4, 6->49, 51->74, 76->99]
      }
      public static List<String> solve(int[] nums, int lower, int upper) {
      }
}
```