Question 1:

```
class Solution {
  public int[] twoSum(int[] nums, int target) {
    HashMap<Integer, Integer> map = new HashMap<>();
    for(int i =0; i< nums.length; i++){
      if(map.containsKey(nums[i])){
        return new int[]{map.get(nums[i]), i};
      }else{
        map.put(target-nums[i], i);
      }
    }

    return new int[]{};


  }
}
```

Question 2:

```
class Solution {
  public int removeElement(int[] nums, int val) {
    int j=0;
    for(int i=0; i< nums.length; i++){
      if(nums[i] != val){
        nums[j] = nums[i];
        j++;
      }
    }

    return j;

  }
}
```

Question 3:

```java
class Solution {
  public int searchInsert(int[] nums, int target) {

    int i=0;
    int j= nums.length-1;
    while(i<=j){
      int mid= (i+j) /2;
     // System.out.println(mid);
      if(target== nums[mid]){
        return mid;
      } else if(target< nums[mid]){
        j= mid-1;
      }else
        i= mid+1;




    }
    return j+1;

  }
}
```

Question 4:

```java
class Solution {
  public int[] plusOne(int[] digits) {
    int n = digits.length;
    for(int i = n - 1; i >= 0; i --) {
      if(digits[i] < 9) {
        digits[i] ++;
        return digits;
      } else {
        digits[i] = 0;
      }
    }
    int[] res = new int[n + 1];
    res[0] = 1;

    return res;
  }
}
```

Question 5:

```
class Solution {
  public void merge(int[] nums1, int m, int[] nums2, int
n) {
    int i = m-1;
    int j= n-1;
    int k = m+n-1;
    while(i>=0 && j>=0){
      if(nums1[i]> nums2[j]){
        nums1[k--]= nums1[i--];
       // i--;k--;
      }else{
        nums1[k--]= nums2[j--];
       // k--;j--;
      }
    }

    while(j>=0){
      nums1[k--]= nums2[j--];
    }
  }
}
```

Question 6:

```
class Solution {
  public boolean containsDuplicate(int[] nums) {
    HashSet<Integer> set = new HashSet<Integer>();

    for(int i: nums){
      if(!set.add(i)){
        return true;
      }
      set.add(i);
```

```java
    }

    return false;

  }
}
```

Question 7:

```java
class Solution {
  public void moveZeroes(int[] nums) {
    int k=0;
    for(int i=0; i< nums.length;i++){
      if(nums[i]!=0){
        nums[k++]= nums[i];
      }
    }
    for(int j=k; j< nums.length;j++){
      nums[j]=0;
    }

  }
}
```

Question 8:

```java
class Solution {
  public int[] findErrorNums(int[] nums) {


    int[] result = new int[2];
    HashSet<Integer> set = new HashSet<>();

    int duplicate = -1;
    int missing = -1;

    for (int num : nums) {
      if (set.contains(num)) {
        duplicate = num;
      }
      set.add(num);
    }

    for (int i = 1; i <= nums.length; i++) {
      if (!set.contains(i)) {
        missing = i;
        break;
      }
    }

    result[0] = duplicate;
    result[1] = missing;

    return result;
  }
}
```