Q=>1

```
def two_sum(numbers, target):
  i = 0
  j = len(numbers) - 1
  while i < j:
    if numbers[i] + numbers[j] == target:
      return [i + 1, j + 1]
    elif numbers[i] + numbers[j] < target:
      i += 1
    else:
      j -= 1
  return []
```

Q=>2

```
def search_range(nums, target):
  start = 0
  end = len(nums) - 1
  while start <= end:
    mid = (start + end) // 2
    if nums[mid] == target:
      left = mid
      while left > 0 and nums[left - 1] == target:
        left -= 1
      right = mid
      while right < len(nums) - 1 and nums[right + 1] == target:
        right += 1
      return [left, right]
    elif nums[mid] < target:
      start = mid + 1
    else:
      end = mid - 1
  return [-1, -1]
```

Q=>3

```
def find_peak_element(nums):
  start = 0
  end = len(nums) - 1
  while start <= end:
    mid = (start + end) // 2
    if mid == 0 or nums[mid - 1] < nums[mid]:
      if mid == len(nums) - 1 or nums[mid + 1] < nums[mid]:
        return mid
      else:
        end = mid - 1
    else:
      start = mid + 1
  return -1
```

Q=>4

```
def search_insert(nums, target):
  start = 0
  end = len(nums) - 1
  while start <= end:
    mid = (start + end) // 2
    if nums[mid] == target:
      return mid
    elif nums[mid] < target:
      start = mid + 1
    else:
      end = mid - 1
  return start
```

Q=>5

```python
def find_majority_element(nums):
  count = 0
  candidate = nums[0]
  for num in nums:
    if count == 0:
      candidate = num
      count = 1
    elif num == candidate:
      count += 1
    else:
      count -= 1
  return candidate if count > 0 else None
```

Q=>7

```python
def find_inversions(nums):
  inversions = 0
  for i in range(len(nums)):
    for j in range(i + 1, len(nums)):
      if nums[i] > nums[j]:
        inversions += 1
  return inversions
```

Q=>8

```python
def find_common_elements(ar1, ar2, ar3):
  common_elements = []
  i = j = k = 0
  while i < len(ar1) and j < len(ar2) and k < len(ar3):
    if ar1[i] == ar2[j] == ar3[k]:
      common_elements.append(ar1[i])
      i += 1
      j += 1
      k += 1
    elif ar1[i] < ar2[j]:
      i += 1
    elif ar2[j] < ar3[k]:
      j += 1
    else:
      k += 1
  return common_elements
```