Q=>1

```
def common_elements(arr1, arr2, arr3):
  i = j = k = 0
  common = []
  while i < len(arr1) and j < len(arr2) and k < len(arr3):
    if arr1[i] == arr2[j] == arr3[k]:
      common.append(arr1[i])
      i += 1
      j += 1
      k += 1
    elif arr1[i] < arr2[j]:
      i += 1
    elif arr2[j] < arr3[k]:
      j += 1
    else:
      k += 1

  return common



if __name__ == "__main__":
  arr1 = [1, 2, 3, 4, 5]
  arr2 = [1, 2, 5, 7, 9]
  arr3 = [1, 3, 4, 5, 8]
  print(common_elements(arr1, arr2, arr3))
```

Q=>2

```
def find_distinct_numbers(nums1, nums2):
  distinct_nums1 = set()
  distinct_nums2 = set()

  for num in nums1:
    distinct_nums1.add(num)

  for num in nums2:
```

```
    distinct_nums2.add(num)

  answer = []
  for num in distinct_nums1:
    if num not in distinct_nums2:
      answer.append(num)

  for num in distinct_nums2:
    if num not in distinct_nums1:
      answer.append(num)

  return answer



if __name__ == "__main__":
  nums1 = [1, 2, 3]
  nums2 = [2, 4, 6]
  print(find_distinct_numbers(nums1, nums2))
```

Q=>3

```
def transpose(matrix):
  transposed = []
  for i in range(len(matrix[0])):
    row = []
    for j in range(len(matrix)):
      row.append(matrix[j][i])
    transposed.append(row)

  return transposed



if __name__ == "__main__":
  matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
  print(transpose(matrix))
```

Q=>4

```
def max_pair_sum(nums):
  nums.sort()
  sum = 0
  for i in range(0, len(nums), 2):
    sum += min(nums[i], nums[i + 1])

  return sum
if __name__ == "__main__":
  nums = [1, 4, 3, 2]
  print(max_pair_sum(nums))
```

Q=>5

```
def staircase_rows(n):
  rows = 0
  current_row = 1
  while current_row <= n:
    rows += 1
    current_row += current_row

  return rows



if __name__ == "__main__":
  n = 5
  print(staircase_rows(n))
```

Q=>6

```
def squares_of_sorted_array(nums):
  squares = []
  for num in nums:
    squares.append(num * num)

  squares.sort()

  return squares
```

```
if __name__ == "__main__":
  nums = [-4, -1, 0, 3, 10]
  print(squares_of_sorted_array(nums))
```

Q=>7

```
def max_integers_in_matrix(m, n, ops):
  max_integer = 0
  count = 0
  for ai, bi in ops:
    for i in range(ai):
      for j in range(bi):
        if M[i][j] > max_integer:
          max_integer = M[i][j]
        M[i][j] += 1

    for i in range(ai):
      for j in range(bi):
        if M[i][j] == max_integer:
          count += 1

  return count



if __name__ == "__main__":
  m = 3
  n = 3
  ops = [[2, 2], [3, 3]]
  print(max_integers_in_matrix(m, n, ops))
```

Q=>8

```
def rearrange_array(nums, n):
  rearranged = []
  for i in range(n):
    rearranged.append(nums[i])
    rearranged.append(nums[i + n])
```

```
  return rearranged



if __name__ == "__main__":
  nums = [2, 5, 1, 3, 4, 7]
  n = 3
  print(rearrange_array(nums, n))
```