```python
# ==========Q1==========


def check(arr, n):


    count = 0


    index = -1

    for i in range(1, n):

        if (arr[i - 1] >= arr[i]):

            count += 1


            index = i


    if (count > 1):
        return False


    if (count == 0):
        return True

    if (index == n - 1 or index == 1):
        return True


    if (arr[index - 1] < arr[index + 1]):
        return True


    if (arr[index - 2] < arr[index]):
        return True

    return False
```

```
if __name__ == '__main__':
    arr = [1, 2, 5, 3, 5]
    N = len(arr)
    if (check(arr, N)):
        print("Yes")
    else:
        print("No")

# ==========Q2==========

class Solution:
    def findDifference(self, nums1: List[int], nums2: List[int]) ->
List[List[int]]:
        return [[i for i in set(nums1)-set(nums2)], [j for j in
set(nums2)-set(nums1)]]



    # ============Q3=================
    class Solution:
    def transpose(self, A):
        R = len(A)
        C = len(A[0])
        transpose = []
        for c in range(C):
            newRow = []
            for r in range(R):
                newRow.append(A[r][c])
            transpose.append(newRow)
        return transpose


    # ====================Q4==========

    class Solution {
public:
    int arrayPairSum(vector<int>& nums) {
        sort(nums.begin(),nums.end());
```

```
        int sum=0;
        for(int i=0;i<nums.size();i+=2)
            sum+=nums[i];
        return sum;
    }
};

    # ==========Q5=============

    def arrangeCoins(self, n: int) -> int:
    return int((-1 + (1 + 8*n) ** 0.5) // 2)


# =============Q6==============

class Solution:
    def sortedSquares(self, A: List[int]) -> List[int]:
        result = [None for _ in A]
        left, right = 0, len(A) - 1
        for index in range(len(A)-1, -1, -1):
            if abs(A[left]) > abs(A[right]):
                result[index] = A[left] ** 2
                left += 1
            else:
                result[index] = A[right] ** 2
                right -= 1
        return result


    #=================Q7================
    class Solution:
    def maxCount(self, m: int, n: int, ops: List[List[int]]) -> int:
        length = len(ops)
        if length == 0:
            return m*n
        result = [ops[0][0] , ops[0][1]]
        for i in range(1,length):
            result[0] = min(result[0] , ops[i][0])
            result[1] = min(result[1] , ops[i][1])
        return result[0]*result[1]
```

```python
# ==============Q8================

class Solution:
def shuffle(self, nums: List[int], n: int) -> List[int]:
    ## APPROACH : GREEDY ##
    ans = []
    for i in range(n):
        ans.append(nums[i])
        ans.append(nums[i+n])
    return ans
```