```
1. def findMedianSortedArrays(nums1, nums2):
    if len(nums1) > len(nums2):
        nums1, nums2 = nums2, nums1

    m, n = len(nums1), len(nums2)
    imin, imax, half_len = 0, m, (m + n + 1) // 2

    while imin <= imax:
        i = (imin + imax) // 2
        j = half_len - i

        if i < m and nums1[i] < nums2[j - 1]:
            imin = i + 1
        elif i > 0 and nums1[i - 1] > nums2[j]:
            imax = i - 1
        else:
            # i is perfect
            if i == 0: max_of_left = nums2[j - 1]
            elif j == 0: max_of_left = nums1[i - 1]
            else: max_of_left = max(nums1[i - 1], nums2[j - 1])

            if (m + n) % 2 == 1:
                return max_of_left

            if i == m: min_of_right = nums2[j]
            elif j == n: min_of_right = nums1[i]
            else: min_of_right = min(nums1[i], nums2[j])

            return (max_of_left + min_of_right) / 2.0
nums1 = [1, 3]
nums2 = [2]
print(findMedianSortedArrays(nums1, nums2))
```

2.
```
def repeatedStringMatch(a, b):
    min_repeats = -(-len(b) // len(a))
    if b in a * min_repeats:
        return min_repeats
    elif b in a * (min_repeats + 1):
        return min_repeats + 1
    else:
        return -1
a = "abcd"
b = "cdabcdab"
print(repeatedStringMatch(a, b))
```

3.
```
def fourSum(nums, target):
    nums.sort()
    result = []
    n = len(nums)

    for i in range(n - 3):
        if i > 0 and nums[i] == nums[i - 1]:
            continue
        for j in range(i + 1, n - 2):
            if j > i + 1 and nums[j] == nums[j - 1]:
                continue
            left, right = j + 1, n - 1
            while left < right:
                total = nums[i] + nums[j] + nums[left] + nums[right]
                if total == target:
                    result.append([nums[i], nums[j], nums[left], nums[right]])
                    while left < right and nums[left] == nums[left + 1]:
                        left += 1
                    while left < right and nums[right] == nums[right - 1]:
                        right -= 1
```

```
                left += 1
                right -= 1
            elif total < target:
                left += 1
            else:
                right -= 1

    return result
nums1 = [1, 0, -1, 0, -2, 2]
target1 = 0
print(fourSum(nums1, target1))

nums2 = [2, 2, 2, 2, 2]
target2 = 8
print(fourSum(nums2, target2))
```

4.
```
def missingNumber(nums):
    n = len(nums)
    sum_n = n * (n + 1) // 2
    sum_nums = sum(nums)
    return sum_n - sum_nums
nums1 = [3, 0, 1]
print(missingNumber(nums1))
nums2 = [9, 6, 4, 2, 3, 5, 7, 0, 1]
print(missingNumber(nums2))
```

5.
```
def majorityElement(nums):
    candidate = None
    count = 0
    for num in nums:
        if count == 0:
            candidate = num
```

```
            count = 1
        elif num == candidate:
            count += 1
        else:
            count -= 1

    count = 0
    for num in nums:
        if num == candidate:
            count += 1

    return candidate
nums = [3, 2, 3]
print(majorityElement(nums))
```