Class Notes 3

- Python
- JavaScript
- Java
- Cpp

Question 1

Implement pow(x, n), which calculates x raised to the power n (i.e., xn).

Example 1:

Input: x = 2.00000, n = 10

Output: 1024.00000

TC : O(log *n*)

SC : O(1)

```
class Solution {
public:
    double myPow(double x, int n) {
        if (n == 0)
            return 1;
        double half = myPow(x, n / 2);
        if (n % 2 == 0)
            return half * half;
        else if (n > 0)
            return half * half * x;
        else
            return half * half / x;
    }
};
```

Question 2

A permutation of an array of integers is an arrangement of its members into a sequence or linear order.

- For example, for arr = [1,2,3], the following are all the permutations of arr: [1,2,3], [1,3,2], [2, 1, 3], [2, 3, 1], [3,1,2], [3,2,1].

The next permutation of an array of integers is the next lexicographically greater permutation of its integer. More formally, if all the permutations of the array are sorted in one container according to their lexicographical order, then the next permutation of that array is the permutation that follows it in the sorted container. If such arrangement is not possible, the array must be rearranged as the lowest possible order (i.e., sorted in ascending order).

Given an array of integers nums, find the next permutation of nums.

The replacement must be in place and use only constant extra memory.

Example 1:

Input: nums = [1,2,3]
Output: [1,3,2]

Time complexity: O(n). In worst case, only two scans of the whole array are needed.

Space complexity: O(1). No extra space is used. In place replacements are done.

```
class Solution {
public:
    void nextPermutation(vector<int>& nums) {
        int n = nums.size(), i, j;
        for (i = n - 2; i >= 0; i--) {
            if (nums[i] < nums[i + 1]) {
                for (j = n - 1; j > i; j--) {
                    if (nums[j] > nums[i]) {
                        break;
                    }
                }
                swap(nums[i], nums[j]);
                reverse(nums.begin() + i + 1, nums.end());
                return;
            }
        }
        reverse(nums.begin(), nums.end());
    }
};
```

Question 3

Given an array arr[] of distinct elements size N that is sorted and then around an unknown point, the task is to check if the array has a pair with a given sum X.

Examples :

Input: arr[] = {11, 15, 6, 8, 9, 10}, X = 16

Output: true

Explanation: There is a pair (6, 10) with sum 16

Time Complexity: O(n), where n is the length of the input array.

Space Complexity: O(1).

```
#include <bits/stdc++.h>
using namespace std;

bool findPair(int arr[], int n, int x) {
    int i;
    for (i = 0; i < n - 1; i++)
        if (arr[i] > arr[i + 1])
            break;
    int l = (i + 1) % n, r = i;

    while (l != r) {
        if (arr[l] + arr[r] == x)
            return

### Question 3

Given an array arr[] of distinct elements size N that is sorted and then aro
und an unknown point, the task is to check if the array has a pair with a gi
ven sum X.

#### Examples :

Input: arr[] = {11, 15, 6, 8, 9, 10}, X = 16

Output: true

Explanation: There is a pair (6, 10) with sum 16

Time Complexity: O(n), where n is the length of the input array.

Space Complexity: O(1).

```cpp
#include <bits/stdc++.h>
using namespace std;

bool findPair(int arr[], int n, int x) {
    int i;
    for (i = 0; i < n - 1; i++)
        if (arr[i] > arr[i + 1])
            break;
    int l = (i + 1) % n, r = i;

    while (l != r) {
        if (arr[l] + arr[r] == x)
            return true;
        if (arr[l] + arr[r] < x)
            l = (l + 1) % n;
        else
            r = (n + r - 1) % n;
    }
    return false;
}
```

Question 4

Given an array nums with n objects colored red, white, or blue, sort them in-place so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

Input: nums = [2,0,2,1,1,0]

Output: [0,0,1,1,2,2]

Example 2:

Input: nums = [2,0,1]

Output: [0,1,2]

```
class Solution {
public:
    void sortColors(vector<int>& nums) {
        int low = 0, mid = 0, high = nums.size()-1;
        while(mid <= high){
            if(nums[mid] == 0){
                swap(nums[low], nums[mid]);
                low++;
                mid++;
            }
            else if(nums[mid] == 1){
                mid++;
            }
            else{
                swap(nums[mid], nums[high]);
                high--;
            }
        }
    }
};
```

Question 5

Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.

Example 1:

Input: nums = [1,2,3,4,5,6,7], k = 3

Output: [5,6,7,1,2,3,4]

Explanation:

rotate 1 steps to the right: [7,1,2,3,4,5,6]

rotate 2 steps to the right: [6,7,1,2,3,4,5]

rotate 3 steps to the right: [5,6,7,1,2,3,4]

Solution:

TC: O(n)

SC: O(1)

```
class Solution {
public:
    void rotate(vector<int>& nums, int k) {
        int n = nums.size();
        k %= n;
        reverse(nums.begin(), nums.end());
        reverse(nums.begin(), nums.begin() + k);
        reverse(nums.begin() + k, nums.end());
    }
};
```

Question 6

Given a binary array nums, return *the maximum number of consecutive* 1's in the array*.*

Example 1:

Input: nums = [1,1,0,1,1,1]

Output: 3

Explanation: The first two digits or the last three digits are consecutive 1s. The maximum number of consecutive 1s is 3.

Solution:

TC : O(n)

SC : O(1)

```
class Solution {
public:
    int findMaxConsecutiveOnes(vector<int>& nums) {
        int count = 0;
        int maxCount = 0;
        for (int i = 0; i < nums.size(); i++) {
            if (nums[i] == 1) {
                // Increment the count of 1's by one.
                count += 1;
            } else {
                // Find the maximum till now.
                maxCount = max(maxCount, count);
                // Reset count of 1.
                count = 0;
            }
        }
        return max(maxCount, count);
    }
};
```
```