# Assignment 4

## Student Name: Manish Kumar UID:22BCS10249

## Branch: BE-CSE (General) Section/Group: FL\_IOT-602 A

## Semester:6th Date of Performance: 21-02-25

## Subject Name: Advanced Programming Lab-2 Subject Code: 22CSP-351

# Aim: 1763. Longest Nice Substring

# Implementation/ Code:

# // 1763. Longest Nice Substring

# class Solution {

# public:

# bool check(string s){

# vector<int> t1(26, 0), t2(26, 0);

# for(int i=0; i<s.length(); i++) if(s[i] >= 'a' && s[i] <= 'z') t1[s[i] - 'a']++;

# for(int i=0; i<s.length(); i++) if(s[i] >= 'A' && s[i] <= 'Z') t2[s[i] - 'A']++;

# for(int i=0; i<26; i++){

# if(t1[i] == 0 && t2[i] == 0) continue;

# else if(t1[i] == 0 && t2[i] > 0) return false;

# else if(t1[i] > 0 && t2[i] == 0) return false;

# }

# return true;

# }

# string longestNiceSubstring(string s) {

# if(s == "") return "";

# if(check(s)) return s;

# unordered\_map<char,int> m;

# vector<int> t1(26, 0), t2(26, 0);

# for(int i=0; i<s.length(); i++) if(s[i] >= 'a' && s[i] <= 'z') t1[s[i] - 'a']++;

# for(int i=0; i<s.length(); i++) if(s[i] >= 'A' && s[i] <= 'Z') t2[s[i] - 'A']++;

# for(int i=0; i<26; i++){

# if(t1[i] == 0 && t2[i] == 0) continue;

# else if(t1[i] == 0 && t2[i] > 0) m[i + 'A']++;

# else if(t1[i] > 0 && t2[i] == 0) m[i + 'a']++;

# }

# vector<string> ans; string temp = "";

# for(int i=0; i<s.length(); i++){

# if(m.find(s[i]) == m.end()) temp += s[i];

# else {

# ans.push\_back(temp); temp = "";

# }

# }

# if(temp != "") ans.push\_back(temp);

# string kk = "";

# for(int i=0; i<ans.size(); i++){

# if(ans[i] == "") continue;

# string t = longestNiceSubstring(ans[i]);

# if(t == "") continue;

# if(kk == "" || kk.length() < t.length()) kk = t;

# 

# }

# return kk;

# }

# };

# Output:

# 

# Aim: 190. Reverse Bits

# Implementation/ Code:

# class Solution {

# public:

# uint32\_t reverseBits(uint32\_t n) {

# uint32\_t result = 0;

# for (int i = 0; i < 32; i++) {

# int bit = n & 1;       // Extract the least significant bit

# result = (result << 1) | bit; // Append the bit to the result

# n = n >> 1;           // Right-shift n to process the next bit

# }

# return result;

# }

# };

# Output:

# 

# Aim: 191. Number of 1 Bits

# Implementation/ Code:

# class Solution {

# public:

# int hammingWeight(uint32\_t n) {

# int res = 0;

# for (int i = 0; i < 32; i++) {

# if ((n >> i) & 1) {

# res += 1;

# }

# }

# return res;

# }

# };

# Output:

# 

# Aim: 53. Maximum Subarray

# Implementation/ Code:

# class Solution {

# public:

# int maxSubArray(vector<int>& nums) {

# int res = nums[0];

# int total = 0;

# for (int n : nums) {

# if (total < 0) {

# total = 0;

# }

# total += n;

# res = max(res, total);

# }

# return res;

# }

# };

# Output:

# 

# Aim: 240. Search a 2D Matrix II

# Implementation/ Code:

# class Solution {

# public:

# bool searchMatrix(vector<vector<int>>& matrix, int target) {

# int n=matrix.size();

# int m=matrix[0].size();

# int row=0,col=m-1;

# while(row<n && col>=0){

# if(matrix[row][col]==target){

# return true;

# }else if(matrix[row][col]<target){

# row++;

# }else{

# col--;

# }

# }

# return false;

# }

# };

# Output:

# 

# Aim: 372. Super Pow

# Implementation/ Code:

# class Solution {

# private:

# int solve(int base, int power, int mod) {

# int ans = 1;

# while (power > 0) {

# if (power & 1) {

# ans = (ans \* base) % mod;

# }

# base = (base \* base) % mod;

# power >>= 1;

# }

# return ans;

# }

# public:

# int superPow(int a, vector<int>& b) {

# a%=1337;

# int n = b.size();

# int m = 1140;

# int expi = 0;

# for(int i : b){

# expi = (expi\*10+i)%m;

# }

# if (expi == 0) {

# expi = m;

# }

# return solve(a,expi,1337);

# }

# };

# Output:

# 