Experiment 3

Student Name: Ranit Pal | **UID: 22BCS14455**

Branch: CSE | **Section/Group: 617-A**

Semester: 6th | **Date of Performance: 23/01/2025**

Subject Name: AP Lab-II | **Subject Code: 22CSP-351**

1) Aim:

Longest-nice-substring

2) Implementation/Code:

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        string output = "";
        int count = 0;
        for(int i = 0;i<s.length();i++){
            int smallMask=0;
            int largeMask = 0;
            char ch = s[i];
            int chint = 0;
            if(ch>=65 && ch<=90){
                chint = ch-'A';
                largeMask = 1<<chint;
            }
            else{
```

```
chint = ch-'a';
                smallMask = 1<<chint;
            }
            for(int j = i+1;j<s.length();j++){
                ch = s[j];
                if(ch>=65 && ch<=90){
                    chint = ch-'A';
                    largeMask |= 1<<chint;
                }
                else{
                    chint = ch-'a';
                    smallMask |= 1<<chint;
                }
                //checking for nice
                if((smallMask^largeMask) == 0){
                    if(count<j-i+1){
                        count = j-i+1;
                        string temp(s.begin()+i,s.begin()+j+1);
                        output = temp;
                    }
                }
            }
        }
        return output;
    }
};
```

Problem:2

Aim: Reverse-bits

Implementation:

```
class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t result = 0;
        for (int i = 0; i < 32; i++) {
            int bit = n & 1;
            result = (result << 1) | bit;
            n = n >> 1;
        }
        return result;
    }
};
```

Problem:3

Aim:

Number of 1 Bits

Code:

```
class Solution {

public:

    int hammingWeight(uint32_t n) {

        int count = 0;
```

```
 while(n != 0){
           n &= (n-1);
           count++;
        }
        return count;
    }};
```

Problem:4

Aim: Maximum-subarray

Code:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = nums[0];
        int currentSum = nums[0];
        for (int i = 1; i < nums.size(); i++) {
            currentSum = max(nums[i], currentSum + nums[i]);
            maxSum = max(maxSum, currentSum);
        }
        return maxSum;    }};
```

Problem:5

Aim: Search-a-2d-matrix-ii

Code:

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int cols = matrix[0].size() - 1;
        int n = matrix.size() - 1;
        int rows = 0;

        while(rows <= n && cols >= 0){
            int toCompare = matrix[rows][cols];
            if(toCompare > target){
                cols--;
            }else if(toCompare < target){
                rows++;
            }else{
                return true;
            }
        }

        return false;
    }};
```

Problem:6

Aim:

Super-Pow

Code:

```
class Solution {
public:
    const int MOD = 1337;

    int pow(int a, int b) {
        int result = 1;
        a %= MOD;
        for (int i = 0; i < b; i++) {
            result = (result * a) % MOD;
        }
        return result;
    }

    int superPow(int a, vector<int>& b) {
        int result = 1;
        for (int i = b.size() - 1; i >= 0; i--) {
            result = (result * pow(a, b[i])) % MOD;
            a = pow(a, 10);
        }
        return result;    }};
```

Problem:7

Aim:

Beautiful Array

Code:

```
class Solution {
public:
    vector<int> beautifulArray(int n)
    {
        vector<int> res = {1};
        while (res.size() < n)
        {
            vector<int> temp;
            for (int it : res)
            {
                if (it * 2 - 1 <= n)
                  temp.push_back(it * 2 - 1);
            }
            for (int it : res)
            {
                if (it * 2 <= n)
                  temp.push_back(it * 2);
            }
            res = temp;} return res;  }};
```