DAY 6

INTERVIEW BIT PROBLEMS :

1. PRETTYPRINT

Print concentric rectangular pattern in a 2d matrix.
Example 1:
Input: A = 4.
Output:
4 4 4 4 4 4 4
4 3 3 3 3 3 4
4 3 2 2 2 3 4
4 3 2 1 2 3 4
4 3 2 2 2 3 4
4 3 3 3 3 3 4
4 4 4 4 4 4 4

Example 2:
Input: A = 3.
Output:
3 3 3 3 3
3 2 2 2 3
3 2 1 2 3
3 2 2 2 3
3 3 3 3 3
The outermost rectangle is formed by A, then the next outermost is formed by A-1 and so on.
You will be given A as an argument to the function you need to implement, and you need to return a 2D array.

CODE :

PYTHON

```
class Solution:
```

```
    # @param A : integer
    # @return a list of list of integers
    def prettyPrint(self, A):
        n=2*A-1
        output=[[0]*n]*n
        result=[]
        for i in range(n):
            for j in range(n):
                if (abs(i-(n//2))>abs(j-(n//2))):
                    output[i][j]=abs(i-(n//2))+1
                else:
                    output[i][j]=abs(j-(n//2))+1
            result.append(list(output[0]))
        return result
```

2. Length of Last Word

Given a string s consists of upper/lower-case alphabets and empty space characters ' ', return the length of last word in the string.
If the last word does not exist, return 0.
Note: A word is defined as a character sequence consists of non-space characters only.
Example:
Given s = "Hello World",
return 5 as length("World") = 5.

CODE :

PYTHON

```
class Solution:
    # @param A : string
    # @return an integer
    def lengthOfLastWord(self, A):
```

```
        n=len(A)
        if n==0:
            return 0
        c=0
        check=False
        for i in range(n-1,-1,-1):
            if check==False:
                if A[i].isalpha():
                    check=True
            if check==True:
                if A[i]!=' ':
                    c+=1
                else:
                    break
        return c
```

3. Palindrome String

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

Example:

"A man, a plan, a canal: Panama" is a palindrome.

"race a car" is not a palindrome.

Return 0 / 1 (0 for false, 1 for true) for this problem

CODE :

PYTHON

```
class Solution:
    # @param A : string
    # @return an integer
    def isPalindrome(self, A):
        output=""
        n=len(A)
```

```
if n==0:
    return 1
for i in range(n):
    if A[i].isalpha() or A[i].isdigit():
        output+=A[i]
if len(output)==0 or len(output)==1:
    return 1
if output.lower()==output[::-1].lower():
    return 1
else:
```

4. Longest Common Prefix

Given the array of strings A,
you need to find the longest string S which is the prefix
of ALL the strings in the array.
Longest common prefix for a pair of strings S1 and S2 is the
longest string S which is the prefix of both S1
and S2.
For Example, longest common prefix
of "abcdefgh" and "abcefgh" is "abc".

Input Format

The only argument given is an array of strings A.

Output Format

Return longest common prefix of all strings in A.

For Example

Input 1:

A = ["abcdefgh", "aefghijk", "abcefgh"]

Output 1:

"a"

Explanation 1:

Longest common prefix of all the strings is "a".

Input 2:

A = ["abab", "ab", "abcd"];

Output 2:

"ab"

Explanation 2:

Longest common prefix of all the strings is "ab".

CODE :

C++

```
string Solution::longestCommonPrefix(vector<string> &A) {
    int n=A.size();
    if(n==0){
        return "";
    }
    if(n==1){
        return A[0];
    }
    string output="";
    for(int i=0;i<A[0].length();i++){
        bool pre_check=true;
        for(int j=1;j<n;j++){
            if(i>=A[j].length()||A[j][i]!=A[0][i]){
                pre_check=false;
                break;
            }
        }
        if(!pre_check){
            break;
        }
        output.push_back(A[0][i]);
    }
    return output;
}
```

PYTHON

```
class Solution:
    # @param A : list of strings
    # @return a strings
    def longestCommonPrefix(self, A):
        n=len(A)
        if n==0:
            return ""
        if n==1:
            return A[0]
        output=""
        for i in range(len(A[0])):
            pre_check=True
            for j in range(1,n):
                if (i>=len(A[j])) or (A[j][i]!=A[0][i]):
                    pre_check=False
                    break
            if(not(pre_check)):
                break
            output+=A[0][i]
        return output
```

5. Count And Say

The count-and-say sequence is the sequence of integers beginning as follows:
1, 11, 21, 1211, 111221, ...
1 is read off as one 1 or 11.
11 is read off as two 1s or 21.
21 is read off as one 2, then one 1 or 1211.
Given an integer n, generate the nth sequence.
Note: The sequence of integers will be represented as a string.
Example:

if n = 2,
the sequence is 11.

CODE :

PYTHON

```
class Solution:
    # @param A : integer
    # @return a strings
    def countAndSay(self, A):
        if A==1:
            return '1'
        if A==2:
            return '11'
        #previous term
        p="11"
        for i in range(3,A+1):
            p+='@'
            l=len(p)
            count=1
            temp=""
            for j in range(1,l):
                if p[j]!=p[j-1]:
                    temp+=str(count)
                    temp+=p[j-1]
                    count=1
                else:
                    count+=1
            p=temp
        return p
```

C++

```
string Solution::countAndSay(int A) {
```

```
    if(A==0){
        return " ";
    }
    if(A==1){
        return "1";
    }
    if(A==2){
        return "11";
    }
    string p="11";
    for(int i=3;i<A+1;i++){
        p+='@';
        int l=p.size();
        int count=1;
        string temp="";
        int j;
        for(int j=1;j<l;j++){
            if(p[j]!=p[j-1]){
                temp+=to_string(count);
                temp+=p[j-1];
                count=1;
            }
            else{
                count+=1;
            }
        }
        p=temp;
    }
    return p;
}
```