**DAY -10**

**PROBLEM 1-** [**14. Longest Common Prefix**](https://leetcode.com/problems/longest-common-prefix/)

CODE-

class Solution {

public:

    string longestCommonPrefix(vector<string>& s) {

        int n=s.size();

        sort(s.begin(),s.end());

        int m=min(s[0].size(),s[n-1].size());

       string first=s[0];

       string last=s[n-1];

       int i=0;

       while(i<m && first[i]==last[i])

       {

        i++;

       }

       string z=first.substr(0,i);

       return z;

    }

};

# PROBLEM 2- Sort a linked list of 0s, 1s and 2s

CODE-

// C++ Program to sort a linked list 0s, 1s or 2s

#include <bits/stdc++.h>

using namespace std;

/\* Link list node \*/

class Node

{

public:

int data;

Node\* next;

};

// Function to sort a linked list of 0s, 1s and 2s

void sortList(Node \*head)

{

int count[3] = {0, 0, 0}; // Initialize count of '0', '1' and '2' as 0

Node \*ptr = head;

/\* count total number of '0', '1' and '2'

\* count[0] will store total number of '0's

\* count[1] will store total number of '1's

\* count[2] will store total number of '2's \*/

while (ptr != NULL)

{

count[ptr->data] += 1;

ptr = ptr->next;

}

int i = 0;

ptr = head;

/\* Let say count[0] = n1, count[1] = n2 and count[2] = n3

\* now start traversing list from head node,

\* 1) fill the list with 0, till n1 > 0

\* 2) fill the list with 1, till n2 > 0

\* 3) fill the list with 2, till n3 > 0 \*/

while (ptr != NULL)

{

if (count[i] == 0)

++i;

else

{

ptr->data = i;

--count[i];

ptr = ptr->next;

}

}

}

/\* Function to push a node \*/

void push (Node\*\* head\_ref, int new\_data)

{

/\* allocate node \*/

Node\* new\_node = new Node();

/\* put in the data \*/

new\_node->data = new\_data;

/\* link the old list of the new node \*/

new\_node->next = (\*head\_ref);

/\* move the head to point to the new node \*/

(\*head\_ref) = new\_node;

}

/\* Function to print linked list \*/

void printList(Node \*node)

{

while (node != NULL)

{

cout << node->data << " ";

node = node->next;

}

cout << endl;

}

/\* Driver code\*/

int main(void)

{

Node \*head = NULL;

push(&head, 0);

push(&head, 1);

push(&head, 0);

push(&head, 2);

push(&head, 1);

push(&head, 1);

push(&head, 2);

push(&head, 1);

push(&head, 2);

cout << "Linked List before Sorting\n";

printList(head);

sortList(head);

cout << "Linked List after Sorting\n";

printList(head);

return 0;

}

// This code is contributed by rathbhupendra

# PROBLEM 3- [134. Gas Station](https://leetcode.com/problems/gas-station/)

**CODE-**

class Solution {

public:

int canCompleteCircuit(vector<int>& gas, vector<int>& cost) {

int sum=0;

int temp=0;

int ind=0;

for(int i=0;i<gas.size();i++)

{

sum+=gas[i]-cost[i];

temp+=gas[i]-cost[i];

if(temp<0)

{

temp=0;

ind=i+1;

}

}

if(sum<0)

return -1;

else

return ind;

}

};

**PROBLEM 4-**

[**628. Maximum Product of Three Numbers**](https://leetcode.com/problems/maximum-product-of-three-numbers/)

class Solution {

public:

    int maximumProduct(vector<int>& nums) {

        int n=nums.size();

       sort(nums.begin(),nums.end());

       int x=nums[n-1];

       int y=nums[n-2];

       int z=nums[n-3];

      int ans=x\*y\*z;

      int res=nums[0]\*nums[1]\*nums[n-1];

       return max(ans,res);

    }

};

**PROBLEM 4-**

**MERGE SORTED ARRAY**

**CODE-**

class Solution {

public:

void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {

vector<int>v;

for(int i=0;i<m;i++)

{

v.push\_back(nums1[i]);

}

for(int i=0;i<n;i++)

{

v.push\_back(nums2[i]);

}

nums1.clear();

sort(v.begin(),v.end());

for(int i=0;i<v.size();i++)

{

nums1.push\_back(v[i]);

}

}

};