**Day-5**

**Problem1**- [**92. Reverse Linked List II**](https://leetcode.com/problems/reverse-linked-list-ii/)

Solution-

class Solution {

public:

    ListNode\* reverseBetween(ListNode\* head, int left, int right) {

        vector<int>v;

        ListNode \*temp=head;

        while(temp!=NULL)

        {

            v.push\_back(temp->val);

            temp=temp->next;

        }

        while(left<right)

        {

            swap(v[left-1],v[right-1]);

            left++;

            right--;

        }

        ListNode \*curr=new ListNode(v[0]);

        ListNode \*ptr=curr;

        for(int i=1;i<v.size();i++ )

        {

             ListNode \*temp=new ListNode(v[i]);

             ptr->next=temp;

             ptr=temp;

        }

        return curr;

    }

};

**Problem2**- [**25. Reverse Nodes in k-Group**](https://leetcode.com/problems/reverse-nodes-in-k-group/)

Solution-

class Solution {

public:

    ListNode\* reverseKGroup(ListNode\* head, int k) {

      ListNode \*temp=head;

      vector<int>v;

      while(temp !=NULL)

      {

       v.push\_back(temp->val);

       temp=temp->next;

      }

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

        if (i + k > v.size())

            break;

        reverse(v.begin() + i, v.begin() + i + k);

        i += k;

    }

    for(auto i:v)

    {

        cout<<i;

    }

    temp=head;

    int i=0;

    while(temp!=NULL)

    {

        temp->val=v[i++];

        temp=temp->next;

    }

    return head;

    }

};

**Problem 3**- [**560. Subarray Sum Equals K**](https://leetcode.com/problems/subarray-sum-equals-k/)

class Solution {

public:

    int subarraySum(vector<int>& nums, int k) {

        int n=nums.size();

        unordered\_map<int,int>m;

        m[0]=1;

        int pre=0;

         int count=0;

        for(int i=0;i<n;i++)

        {

          pre+=nums[i];

          int back= pre-k;

          count+=m[back];

          m[pre]+=1;

        }

        return count;

    }

};

**Problem 4**- [**54. Spiral Matrix**](https://leetcode.com/problems/spiral-matrix/)

class Solution {

public:

    vector<int> spiralOrder(vector<vector<int>>& mat) {

   int m=mat.size();

   int n=mat[0].size();

   int left=0,right=n-1,top=0,bottom=m-1;

    vector<int>ans;

   while(left<=right && top<=bottom)

   {

       for(int i=left;i<=right;i++)

       {

        ans.push\_back(mat[top][i]) ;

        }

        top++;

        for(int i=top;i<=bottom;i++)

        {

            ans.push\_back(mat[i][right]);

        }

        right--;

        if(top<=bottom)

        {for(int i=right;i>=left;i--)

        {

            ans.push\_back(mat[bottom][i]);

        }

        bottom--;}

        if(left<=right)

        {for(int i=bottom;i>=top;i--)

        {

            ans.push\_back(mat[i][left]);

        }

        left++;}

     }

     return ans;

    }

};

**Problem 5**- [**55. Jump Game**](https://leetcode.com/problems/jump-game/)

class Solution {

public:

    bool canJump(vector<int>& nums) {

        int mx=nums[0];

        for(int i=0;i<nums.size()-1;i++){

            mx=max(mx,nums[i]);

            mx--;

            if(mx==-1) return false;

        }

        return true;

    }

};