Q1. Find the first and last occurrence of x.

Solution:

```
vector<int> find(int arr[], int n , int x )
{
   vector<int>v;
   int low = 0, high = n - 1, res = -1;
   while (low <= high)
   {
      int mid = (low + high) / 2;
      if (arr[mid] > x)
      {
         high = mid - 1;
      }
      else if (arr[mid] < x)
      {
         low = mid + 1;
      }
      else {
         res = mid;
         high = mid - 1;
      }
   }
   v.push_back(res);
   low = 0, high = n-1, res = -1;
```

```
    while (low <= high)

    {

        int mid = (low + high) / 2;

        if (arr[mid] > x)

        {

            high = mid - 1;

        }

        else if (arr[mid] < x)

        {

            low = mid + 1;

        }

        else

        {

            res = mid;

            low = mid + 1;

        }

    }

    v.push_back(res);

    return v;

    // code here

}
```

Q2. Union of two arrays.

Solution:

```
int doUnion(int a[], int n, int b[], int m)

{

    //code here

    int c[m+n];

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

    {

        c[i]=a[i];

    }

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

    {

        c[i+n]=b[i];

    }

    sort(c,c+m+n);

    int cnt=1;

    for(int i=1;i<(n+m);i++)

    {

        if(c[i-1]!=c[i])

        {

            cnt++;

        }

    }

    return cnt;
}
```

Q4. Reverse a linked list in groups of given size.

Solution:

```
class Solution
{
  public:
  struct node *reverse (struct node *head, int k)
  {
    // Complete this method
    int count=k;
    struct node *cur=head;
    struct node *prev=NULL;
    struct node *temp;
    while(count-- && cur!=NULL)
    {
      temp=cur->next;
      cur->next=prev;
      prev=cur;
      cur=temp;
    }
    if(head!=NULL)
    {
      head->next=reverse(temp,k);
```

```
        }

        return prev;

    }

};
```

Q5. Spirally Traversing a matrix.

Solution:

```
vector<int>ar;

int top, down, left,right;
int direction = 0;

top = 0;
down = r-1;
left = 0;
right = c-1;

while(left<=right && top <= down) {
if(direction == 0) {
for(int i = left;i<=right;i++) {
ar.push_back(matrix[top][i]);
}
top++;
}

else if(direction == 1) {
for(int i = top; i<=down;i++) {
ar.push_back(matrix[i][right]);
}
right--;
}

else if(direction == 2) {
for(int i = right ; i>=left;i--) {
```

```
ar.push_back(matrix[down][i]);
}
down--;
}

else if(direction == 3) {
for(int i = down; i>=top;i--) {
ar.push_back(matrix[i][left]);
}
left++;
}

direction = (direction+1)%4;
}

return ar;
```

Q6. Mean

Solution:

```
class Solution {
 public:
   int mean(int N , int A[]) {
       // code here
       int sum=0;
       for(int i=0;i<N;i++)
       {
           sum=sum+A[i];
       }
       return sum/N;
   }
};
```

Q7. Number Series.

<u>Solution:</u>

```
class Solution {
  public:
    int findNth(int n){
        int c=0;
        while(n%2==0)
        {
            c++;
            n=n/2;
        }
        return c;
    }
};
```