SUM NO 234: PALINDROME LINKED LIST

```java
class Solution {

       public boolean isPalindrome(ListNode head) {
    ListNode fast=head,slow=head;
    while(fast!=null && fast.next!=null)
    {
        fast=fast.next.next;
        slow=slow.next;
    }
    ListNode rev=reverse(slow);
    while (rev != null)
    {
        if(head.val != rev.val)
        {
            return false;
        }
        head=head.next;
        rev=rev.next;
    }
    return true;
       }
    public ListNode reverse(ListNode head)
    {
        ListNode prev=null;
        while(head!=null)
        {
            ListNode next=head.next;
            head.next=prev;
            prev=head;
            head=next;
        }
        return prev;
    }
}
```

SUM NO 143: REORDER LIST

```java
class Solution {
    public void reorderList(ListNode head) {
        if(head==null || head.next ==null)
        return;
        ListNode slow=head,fast=head;
        while(fast!=null && fast.next != null)
        {
            slow=slow.next;
            fast=fast.next.next;

        }
        ListNode second=slow.next;
        slow.next=null;
        second = reverse(second);
        ListNode first=head;
        while(second != null)
        {
            ListNode T1=first.next;
            ListNode T2=second.next;
            first.next=second;
            second.next=T1;
            first=T1;
            second=T2;
        }

        }
        private ListNode reverse(ListNode head) {
        ListNode prev = null, curr = head;
        while (curr != null) {
            ListNode next = curr.next;
            curr.next = prev;
            prev = curr;
            curr = next;
        }
        return prev;
    }
}
```

SUM NO 509: FIBONACCI NUMBER

```java
class Solution {
    public int fib(int n) {

        int z=0;
        int f=1;
        if (n==0)
        return z;
        if(n==1)
        return f;
        for(int i=2;i<=n;i++)
        {
            int a=z+f;
           System.out.println(a);
            z=f;
            f=a;
        }
        return f;
    }

}
```

SUM NO 198: HOUSE ROBBER

```java
class Solution {
    public int rob(int[] nums) {
        int m=0;
        int n=0;
        for(int num : nums)
        {
            int max = Math.max(m,n+num);
            n=m;
            m=max;
        }
        return m;
    }
}
```

SUM NO 70: CLIMBING STAIRS

```
class Solution {
    public int climbStairs(int n) {
        int[]dp=new int[n+1];
        dp[0]=1;
        dp[1]=1;
        for(int i=2;i<=n;++i)
        dp[i]=dp[i-1]+dp[i-2];
        return dp[n];
    }
}
```

SUM NO 1217: MINIMUM COST TO MOVE CHIPS TO THE SAME POSITION

```
class Solution {
    public int minCostToMoveChips(int[] p) {
        int even=0;
        int odd=0;
        for(int i: p)
        {
            if(i%2==0)
            even++;
            else
            odd++;
        }
        return(Math.min(even,odd));
    }
}
```

SUM NO 740: DELETE AND EARN

```java
class Solution {
    public int deleteAndEarn(int[] nums) {
        int[] val=new int[20000];
        for(int num:nums)
        val[num]+=num;
        int x=0;
        int y=0;
        for(int i : val)
        {
            int max=Math.max(x,y+i);
            y=x;
            x=max;
        }
        return x;
    }
}
```

SUM NO 62: UNIQUE PATHS

```java
class Solution {
    public int uniquePaths(int m, int n) {
      int up=m+n-2;
      int down=Math.min(m-1,n-1);
      long result=1;
      for(int i=1;i<=down;i++)
      {
        result=result*(up--)/i;
      }
      return (int)result;
    }
}
```