**LC#66:PLUS ONE(start from last)**

**IP :digits = [4,3,2,1] , OP: [4,3,2,2]**

**IP: digits = [9] ,OP: [1,0]**

**APP1:NORMAL**

**->calculate the value in the array and add 1 to it and assign it to the array**

**CODE:❌when len exceeds 9 overflow so no error wil come**

class Solution {

    public int[] plusOne(int[] digits) {

        int sum = 0;

        int n = digits.length;

        for(int i = 0 ; i< n ;i++)

        {

            sum = sum\*10+digits[i];

        }

        sum++;

        int arr[] = new int [n+1];

        int j = n;

        while(sum>0)

        {

            arr[j] = sum%10;

            sum/=10;

            j--;

        }

        if(arr[0]==0)

            return Arrays.copyOfRange(arr,1,n+1);

        return arr;

    }

}

**APP2: BIGINTEGER APPROACH**

**->import biginteger**

**->copy the arr value to stringbuilder and convert it to BIGINTEGER**

**-> add 1 to the BIGINTEGER using res.add(BigInteger.valueof(1))**

**->assign a new char array to copy from the biginteger**

**-> assign it to a new resultant array**

**CODE:**

import java.math.BigInteger;

class Solution {

    public int[] plusOne(int[] digits) {

        StringBuilder sb = new StringBuilder();

        for(int n : digits)

            sb.append(n);

        BigInteger res = new BigInteger(sb.toString());

        res = res.add(BigInteger.valueOf(1));

        char chartonum[] =res.toString().toCharArray();

        int resarr[] = new int[chartonum.length];

        for(int i = 0 ; i<chartonum.length;i++)

        {

            resarr[i] = chartonum[i]-'0';

        }

        return resarr;

    }

}

**APP3:OPTIMAL**

**->if nums[i]+1<10 -> add 1 return the array ✅[1,2,3]->1,2,4]**

**->else just convert the current to 0**

**Ex:[8,9,9,9] ->[9,0,0,0]**

**->assign a new array of size n+1 with the arr[0]=1**

**Ex:[9,9,9,9] ->[0,0,0,0]->[1,0,0,0,0]**

**CODE:**

class Solution {

    public int[] plusOne(int[] digits) {

        int n = digits.length;

        for(int i =n-1;i>=0;i--)

        {

            if(digits[i]+1<10)

            {

                digits[i]++;

                return digits;

            }

            else

{

                digits[i]=0;

            }

            int res[] = new int[n+1];

            res[0] = 1;//Arrays.fill(res , 1 , n+1,0);**//optional**

            return res;

    }

}