**1. Sort an array of 0’s 1’s 2’s without using extra space or sorting algo**

**Best: Time=n and space:1 by taking 3 pointers - left, right and mid**

class Solution {

public void sortColors(int[] nums) {

int l=0,r=nums.length-1,mid=0;

while(mid<=r)

{

if(nums[mid]==0)

{

nums[mid]=nums[l];

nums[l]=0;

l++;

mid++;

}

else if(nums[mid]==2)

{

nums[mid]=nums[r];

nums[r]=2;

r--;

}

else

mid++;

}

}

}

Other – by sorting=log n, by counting-count sort=2\*n

**2. Repeat and Missing Number**

**Best: Time=n and space:1 by taking sum of numbers and sum of squares**

class Solution {

public int findDuplicate(int[] nums) {

double tp=1.0,p=1.0,s=0.0,ts=0.0;

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

{

ts+=i+1;

s+=nums[i];

tp+=(i+1)\*(i+1);

p+=nums[i]\*nums[i];

}

double m,d;

m=((ts-s)+((tp-p)/(ts-s)))/2;

d=m-(ts-s);

System.out.println("Missing="+m+" Duplicate="+d);

return 0;

}

}

Other – Space=n and Time=n freq=0 for missing and freq=2 for repeated