**1.stock buy and sell**

int m=0;

int p=prices[0];

for (int i=1;i<len(prices);i++){

if (prices[i]<p)

p=prices[i];

else

m=max(m,prices[i]-p);}

return m;

**2.first and last occurrence**

class solution{

public:

vector<int> find(vector<int>& arr, int x) {

int ind1=-1;

int ind2=-1;

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

if (arr[i]==x && arr[i-1]!=x)

ind1=i;

if (arr[i]==x and arr[i+1]!=x)

ind2=i;

}

if (arr[0]==x)

ind1=0;

if (arr[arr.size()-1]==x)

ind2=arr.size()-1;

if (arr[arr.size()-1]==x && arr[arr.size()-2]!=x)

ind1=arr.size()-1;

return {ind1,ind2};

}

};

**3.Find Transition point**

class Solution {

public:

int transitionPoint(vector<int>& arr) {

if (arr[0]==1)

return 0;

for (int i=1;i<arr.size();i++){

if (arr[i]==1)

return i;

}

return -1;

}

};

**4.First repeating element**

#include <iostream>

#include <vector>

#include <unordered\_map>

#include <climits>

using namespace std;

int firstRepeatingElement(vector<int>& arr) {

unordered\_map<int, int> elementMap;

int firstRepeatingIndex = INT\_MAX;

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

if (elementMap.find(arr[i]) != elementMap.end()) {

firstRepeatingIndex = min(firstRepeatingIndex, elementMap[arr[i]]);

} else {

elementMap[arr[i]] = i + 1;

}

}

return (firstRepeatingIndex == INT\_MAX) ? -1 : firstRepeatingIndex;

}

**5.Remove Duplicates from sorted array**

int k=1;

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

if (nums[i]!=nums[i-1]){

nums[k]=nums[i];

k+=1;}

}

return k;