**MAXIMUM LENGTH BIOTONIC SUBARRAY**

You are provided n numbers of array. You need to find the maximum length of bitonic subarray. A subarray A[i … j] is biotonic if there is a k with i <= k <= j such that A[i] <= A[i + 1] … <= A[k] >= A[k + 1] >= .. A[j – 1] > = A[j] i.e subarray is first increasing and then decreasing or entirely increasing or decreasing.

**Input Format:**

First line contains integer t which is number of test case. For each test case, it contains an integer n which is the size of array and next line contains n space separated integers.

**Constraints:**

1<=t<=100 1<=n<=1000000

**Output Format**

Print the maximum length.

**Sample Input**

2

6

12 4 78 90 45 23

4

40 30 20 10

**Sample Output**

5

4

Program-

#include<iostream>

using namespace std;

int main() {

int t;

cin>>t;

while(t--)

{

int n;

cin>>n;

int arr[n];

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

{

cin>>arr[i];

}

int inc[n];

int dec[n];

int i, max;

inc[0] = 1;

dec[n-1] = 1;

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

inc[i] = (arr[i] >= arr[i-1])? inc[i-1] + 1: 1;

for (i = n-2; i >= 0; i--)

dec[i] = (arr[i] >= arr[i+1])? dec[i+1] + 1: 1;

max = inc[0] + dec[0] - 1;

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

{

if (inc[i] + dec[i] - 1 > max)

max = inc[i] + dec[i] - 1;

}

cout<<max<<endl;

}

}