题意：给你n,m,k以及一数组，n个元素，求一个序列使得序列里最大值减去最小值>=m且<k，求这个序列的最大长度，序列必须连续。

思路：用单调栈维护一个单调递增序列和一个单调递减序列，计算两个单调栈的栈底元素的差值分别跟m，k 作比较得出序列的最大值。详细看代码，注释很详细

注意需要用到now

比如maxx[front1]是2，而maxx[front1+1]却是10

那你那段的区间长度实际是i-3的

#include<stdio.h>

#include<algorithm>

#include<iostream>

using namespace std;

const int N=1e5+5;

int a[N],maxx[N],minn[N],n,m,k;//maxx数组维护单调递减序列，存的是下标，栈底元素最大，minn数组反之

int main()

{

while(~scanf("%d%d%d",&n,&m,&k))

{

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

scanf("%d",&a[i]);

//front为头指针，tail尾指针

int front1=1,tail1=0,front2=1,tail2=0,now=1,ans=0;

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

{

//头指针小于尾指针且当前元素大于栈顶元素时让尾指针减一，即让使栈顶元素出栈，因为要维护一个单调递减序列

while(front1<=tail1&&a[i]>a[maxx[tail1]]) tail1--;

maxx[++tail1]=i;

while(front2<=tail2&&a[i]<a[minn[tail2]]) tail2--;

minn[++tail2]=i;

//两个栈的栈顶元素，一个是存的是数组中最大的元素下标一个是数组中最小的元素下标

//用最大的减去最小的分别跟m，k比较

while(a[maxx[front1]]-a[minn[front2]]>k)

{

now=min(maxx[front1],minn[front2]);//选择更小的下标让其++

if(now==maxx[front1]) front1++;

if(now==minn[front2]) front2++;

now++;

}

if(a[maxx[front1]]-a[minn[front2]]>=m)//记录答案

ans=max(ans,i-now+1);

}

printf("%d\n",ans);

}

}