

采样过滤

题目描述：

在做物理实验时，为了计算物体移动的速率，通过相机等工具周期性的采样物体移动距离。由于工具故障，采样数据存在误差甚至错误的情况。需要通过一个算法过滤掉不正确的采样值。不同工具的故障模式存在差异，算法的各类门限会根据工具类型做相应的调整。请实现一个算法，计算出给定一组采样值中正常值的最长连续周期。判断第 i 个周期的采样数据 $S[i]$ 是否正确的规则如下（假定物体移动速率不超过 10 个单元, 前一个采样周期 $S[i-1]$ ）： $S[i] \leq 0$ ，即为错误值 $S[i] < S[i-1]$ ，即为错误值 $S[i] - S[i-1] \geq 10$ ，即为错误值 其它情况为正常值 判断工具是否故障的规则如下：在 M 个周期内，采样数据为错误值的次数为 T （次数可以不连续），则工具故障。判断故障恢复的条件如下：产生故障后的 P 个周期内，采样数据一直为正常值，则故障恢复。错误采样数据的处理方式：检测到故障后，丢弃从故障开始到故障恢复的采样数据。在检测到工具故障之前，错误的采样数据，则由最近一个正常值代替；如果前面没有正常的采样值，则丢弃此采样数据。给定一段周期的采样数据列表 S ，计算正常值的最长连续周期。输入描述：故障确认周期数和故障次数门限分别为 M 和 T ，故障恢复周期数为 P 。第 i 个周期，检测点的状态为 S_i 输入为两行，格式如下： $M\ T\ P\ S_1\ S_2\ S_3\ \dots$ M 、 T 和 P 的取值范围为 $[1, 100000]$ S_i 取值范围为 $[0, 100000]$ ， i 从 0 开始编号

输出描述：一行输出正常值的最长连续周期

示例1
输入：10 6 3
-1 1 2 3 100 10 13 9 10
输出：8
说明：S[0]，S[4]，S[7]，S[8]为错误值。S[0]之前没有正常的采样数据，丢弃S[0]。S[4]和S[7]不满足故障条件，此值分别由S[3]和S[6]代替，即S[4]为3，S[7]为13。替换后，S[8]小于S[7]，也是错误值。

示例2
输入：5 3 3
0 1 2 -1 4 3 6 7 6 6 10 11 12
输出：9
说明：S[3]，S[5]，S[8]，S[9]为错误值。从S[3]到S[7]的5个周期内只有两个错误值S[3]和S[5]。从S[5]到S[9]的5个周期内有三个错误值S[5]、S[8]和S[9]，工具故障。丢弃S[9]到S[12]的值。

示例
示例1
输入：10 6 3
-1 1 2 3 100 10 13 9 10
输出：8
说明：S[0]，S[4]，S[7]，S[8]为错误值。S[0]之前没有正常的采样数据，丢弃S[0]。S[4]和S[7]不满足故障条件，此值分别由S[3]和S[6]代替，即S[4]为3，S[7]为13。替换后，S[8]小于S[7]，也是错误值。

示例2
输入：5 3 3
0 1 2 -1 4 3 6 7 6 6 10 11 12
输出：9
说明：S[3]，S[5]，S[8]，S[9]为错误值。从S[3]到S[7]的5个周期内只有两个错误值S[3]和S[5]。从S[5]到S[9]的5个周期内有三个错误值S[5]、S[8]和S[9]，工具故障。丢弃S[9]到S[12]的值。

示例3
输入：5 3 3
1 2 -1 -2 -3 6 7 8 9 10 11 12
输出：5
说明：S[2]，S[3]，S[4]为错误值。从S[2]到S[6]的5个周期内有三个错误值，工具故障。丢弃S[4]到S[6]的值。有两段正常连续周期，S[0]到S[3]（周期数为4）和S[7]到S[11]（周期数为5）。

```

1 import sys
2
3 M,T,P=list(map(int,input().split()))
4 nums=list(map(int,input().split()))
5
6 errs=[]
7 n=len(nums)
8
9 errs=[]
10 aerr=0
11 l=0
12 inbroken=False
13 ayes=0
14 startl=0
15 maxl=0
16 aerrs=[]
17 # 0 yes, 1 error, 2 broken,
18 for i in range(n):
19
20     if startl==i:
21         if nums[i]<0: # begin err
22             startl+=1
23             errs.append(1)
24             continue
25
26         aerr=0
27         l=i
28         inbroken=False
29         ayes=0
30         errs.append(0)
31
32     elif nums[i]<0 or nums[i]<nums[i-1] or nums[i]-nums[i-1]>10: # sequence err and modify
33         errs.append(1)
34         nums[i]=nums[i-1]
35     else:
36         errs.append(0)
37
38
39     if errs[i]>0:
40         aerr+=1
41         while l<=i-M:
42             if errs[l]:
43                 aerr-=1
44             l+=1
45         if aerr>=T and errs[i]:
46             errs[i]=2
47             if not inbroken:
48                 endr=i-1

```

```

49         inbroken=True
50
51     if inbroken :
52         if errs[i]==0:
53             ayes+=1
54         else:
55             ayes=0
56         if ayes<P:
57             pass
58             #errs[i]=2
59         else:
60             ayes=0
61             inbroken=False
62             maxl=max(maxl,endr-startl+1)
63             startl=i
64         aerrs.append(ayes)
65     if inbroken is False:
66         maxl=max(maxl,n-startl)
67     #print(errs)
68     #print(nums)
69     #print(aerrs,P)
70     print(maxl)
71
72
73
74

```