# 题目

给定一个由若干 0 和 1 组成的数组 A，我们最多可以将 K 个值从 0 变成 1 。

返回仅包含 1 的最长（连续）子数组的长度。

**示例 1：**

输入：A = [1,1,1,0,0,0,1,1,1,1,0], K = 2

输出：6

解释：

[1,1,1,0,0,1,1,1,1,1,1]

粗体数字从 0 翻转到 1，最长的子数组长度为 6。

**示例 2：**

输入：A = [0,0,1,1,0,0,1,1,1,0,1,1,0,0,0,1,1,1,1], K = 3

输出：10

解释：

[0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1]

粗体数字从 0 翻转到 1，最长的子数组长度为 10。

**提示：**

1、1 <= A.length <= 20000

2、0 <= K <= A.length

3、A[i] 为 0 或 1

# 分析

## 方法一：滑动窗口

class Solution {

public:

int longestOnes(vector<int>& A, int K) {

//count用来统计窗口中0的个数

int left=0,right=0,count=0,ret=0;

while(right<A.size())

{

if(A[right] == 0)

count += 1;

while(count>K)//当窗口内0的个数大于K时，需要缩小窗口

{

if(A[left] == 0)

count -= 1;

left++;

}

//窗口内0的个数小于等于k时，也就是可以该窗口内的0都可以替换，根据该窗口长度来确定是否更新result

ret=max(ret,right-left+1);

right++;

}

return ret;

}

};