'''

Given a binary array nums and an integer k, return the maximum number

of consecutive 1's in the array if you can flip at most k 0's.

In a class students are given series of zeroes and ones and an integer 'k'. They have to return

maximum number of consecutive ones in the series if they can flip at most 'k' zeroes

Input Format

------------

Line1: An integer represents number of zeroes and ones

Line2: space separated integers which are zeroes and ones

Line3: An integer represents no of zeroes which can be flipped

Output Format

-------------

An integer represents maximum number of consecutive ones

Sample Input1:

--------------

11

1 1 1 0 0 0 1 1 1 1 0

2

Sample Output2:

---------------

6

Explanation:

------------

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

Bolded numbers were flipped from 0 to 1. The longest subarray is underlined.

Sample Input2:

--------------

19

0 0 1 1 0 0 1 1 1 0 1 1 0 0 0 1 1 1 1

3

Sample Output2:

---------------

10

Explanation:

------------

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

Bolded numbers were flipped from 0 to 1. The longest subarray is underlined.

'''

n=int(input())

l=list(map(int,input().split()))

k=int(input())

z=0

s=0

a=0

for e in range(n):

if(l[e]==0):

z+=1

while(z>k):

if(l[s]==0):

z-=1

s+=1

a=max(a,e-s+1)

print(a)

Mr Suresh is working with the plain text P, a list of words w[],

He is converting P into C [the cipher text], C is valid cipher of P,

if the following rules are followed:

- The cipher-text C is a string ends with '$' character.

- Every word, w[i] in w[], should be a substring of C, and

the substring should have $ at the end of it.

Your task is to help Mr Suresh to find the shortest Cipher C,

and return its length.

Input Format:

-------------

Single line of space separated words, w[].

Output Format:

--------------

Print an integer result, the length of the shortest cipher.

Sample Input-1:

---------------

kmit it ngit

Sample Output-1:

----------------

10

Explanation:

------------

A valid cipher C is "kmit$ngit$".

w[0] = "kmit", the substring of C, and the '$' is the end character after "kmit"

w[1] = "it", the substring of C, and the '$' is the end character after "it"

w[2] = "ngit", the substring of C, and the '$' is the end character after "ngit"

Sample Input-2:

---------------

abcd bcd d cd

Sample Output-2:

----------------

5

Explanation:

------------

A valid cipher C is "abcd$".

w[0] = "abcd", the substring of C, and the '$' is the end character after "abcd"

w[1] = "bcd", the substring of C, and the '$' is the end character after "bcd"

w[2] = "d", the substring of C, and the '$' is the end character after "d"

w[3] = "cd", the substring of C, and the '$' is the end character after "cd"

import java.util.\*;

public class Main{

static int count=0;

public static void main(String[] args){

Scanner sc=new Scanner (System.in);

String s=sc.nextLine();

String[] arr=s.split(" ");

Arrays.sort(arr,(a,b)->(b.length()-a.length()));

System.out.println(Arrays.toString(arr));

// count=arr[0].length()+1;

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

insert(arr[i]);

}

System.out.println(count);

}

static class Trie{

Trie children[]=new Trie[26];

boolean isend;

Trie(){

isend=false;

}

}

static Trie root=new Trie();

public static void insert(String s){

Trie curr=root;

boolean isnew=false;

for(int i=s.length()-1;i>=0;i--){

if(curr.children[s.charAt(i)-'a']==null){

Trie node=new Trie();

isnew=true;

curr.children[s.charAt(i)-'a']=node;

// curr.isend=true;

// curr.count+=1;

}

curr=curr.children[s.charAt(i)-'a'];

}

if(isnew){

count+=s.length()+1;

}

curr.isend=true;

}

}

There are floods in the eastern India.There are infinite number ofboats available

with National Disaster Response force.Where each boat can carry a maximum weight

limit.

Each boat carries at most 2 people at same time provided the sum of those people

is at most limit.

Return the minimum number of boats to carry every given person to rescue them

Input Format

------------

Line1: Two space separated integers, representing no of people and limit of boat

Line2: space separated integers represents weight of each person

Output Format

-------------

An integer represents minimum no of boats required

Example 1:

-----------

Input1: 2 3

1 2

Output: 1

Explanation: 1 boat (1, 2)

Example 2:

----------

Input2: 4 3

3 2 2 1

Output2: 3

Explanation: 3 boats (1, 2), (2) and (3)

Example 3:

----------

Input3: 4 5

3 5 3 4

Output3: 4

Explanation: 4 boats (3), (3), (4), (5)

import java.util.\*;

class ans{

public static void main(String[] rgs){

Scanner sc=new Scanner(System.in);

String [] one=sc.nextLine().split(" ");

String [] tt=sc.nextLine().split(" ");

List<Integer> l=new ArrayList<>(); List<Integer> l1=new ArrayList<>();

for(String s:one){

l.add(Integer.parseInt(s));

}

for(String s2:tt){

l1.add(Integer.parseInt(s2));

}

Collections.sort(l1);int c=0;

int i=0;int j=l.get(0)-1;

while(i<=j){

if(l1.get(i)+l1.get(j)<=l.get(1)){

i++;

j--;

c++;

}

else{

c=c+1;

j--;

}

}

System.out.println(c);

    }

}

In the present situation, most of the movies releasing in OTTs.

The Showtime OTT in US, introduced a new offer for the customers,

they can purchase either 1-day, 7-day, or 30-day subscription,

and the cost is as follows price[0], price[1], price[2].

The Subscription allows you to watch as many movies as you want with in subscribed days.

For example:

If you purchased, a 7-day subscription on day 5, then you can watch

the movies for 7 days: day 5, 6, 7, 8, 9, 10 and 11.

Your task is to find out the minimum cost, you spend to watch the movies

in the given list of days .

NOTE: Days are numbered from 1, 2, 3, ...365, in sorted order.

Input Format:

-------------

Line 1: Space separated integer days[], list of days.

Line 2: 3 space separated integer price[], cost of subscription.

Output Format:

--------------

Print an integer, minimum cost.

Sample Input-1:

---------------

1 4 6 7 8 20

2 7 15

Sample Output-1:

----------------

11

Explanation:

------------

For example, here is a way to buy subscription, to watch the movies in given days:

On day 1, buy a 1-day subscription for price[0] = $2, which cover day 1.

On day 4, buy a 7-day subscription for price[1] = $7, which cover days 4, 5, ..., 10.

On day 20, buy a 1-day subscription for price[0] = $2, which cover day 20.

In total you spent $11.

Sample Input-2:

---------------

1 2 3 4 5 6 7 8 9 10 30 31

2 7 15

Sample Output-2:

----------------

17

Explanation:

------------

For example, here is a way to buy subscription, to watch the movies in given days:

On day 1, buy a 30-day subscription for price[2] = $15, which cover days 1, 2, 3,....,30.

On day 31, buy a 1-day subscription for price[0] = $2, which cover day 31.

In total you spent $17.

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

String[] arr=s.split(" ");

int n=arr.length;

int[] d=new int[n];

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

d[i]=Integer.parseInt(arr[i]);

}

int op=sc.nextInt(),sp=sc.nextInt(),thp=sc.nextInt();

int[] ca=new int[366];

for(int i:d){

ca[i]=1;

}

for(int i=1;i<366;i++){

if(ca[i]==1){

ca[i]=Math.min(thp+ca[Math.max(i-30,0)],Math.min(sp+ca[Math.max(i-7,0)],op+ca[Math.max(i-1,0)]));

}else{

ca[i]=ca[i-1];

}

}

System.out.print(ca[365]);

}

}