**Hackers with Bits**

Max. Marks: 100

Hackers love bits, so does Alex. Alex has just started his career as hacker and found a special binary array AA (containing 0s0s and 1s1s).

In one operation, he can choose any two positions in the array, and can swap their values. He has to perform exactly one operation and maximize the length of the subarray containing only 1s1s.

As Alex is a newbie in his field, help him for the same, and output the required length of the sub array containing only 1s1s.

**Input Format**:

First line consists of one integer NN, denoting the number of elements in the array.

Second line consists of N space-separated integers, denoting the elements of the array.

**Output Format**:

Print the required length of the sub array containing only 1s1s.

**Input Constraints**:  
1≤N≤1001≤N≤100  
0≤A[i]≤10≤A[i]≤1

**SAMPLE INPUT**

5

1 1 1 0 1

**SAMPLE OUTPUT**

4

**Explanation**

Here N=5N=5.

And if we choose positions 4th4th and 5th5th (1 based indexing in array), we can swap their values and the length of the sub-array (from index 11 to 44) containing only 1s1s, will be 44.

**Time Limit:**1.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded if any testcase passes.

**Allowed Languages:**C, C++, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Swift, Visual Basic

<https://www.hackerearth.com/challenge/competitive/programming-indiahacks-2017/algorithm/hack-the-string-9dce7834/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

int[] arr = Array.ConvertAll(Console.ReadLine().Trim().Split(' '), e => int.Parse(e));

//int[] arr = { 1,0,1 };

// int[] arr = { 0, 1, 0 };

//int[] arr = { 0, 0, 0, 0, 1 };

// int[] arr = { 1, 1, 1, 0, 1 };

// int[] arr = { 1, 1, 0, 1, 1, 0, 1 }; //->5

// int[] arr = { 1, 1, 0, 1, 1, 0, 1,1}; //->5

// int[] arr = { 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1 }; //->6

//cuento los grupos consecutivos de unos

int i=0;

int max\_cont = 0;

int grupos = 0;

while (i < arr.Length)

{

bool entro = false;

int cont = 0;

while (i < arr.Length && arr[i] == 1)

{

cont++;

entro = true;

i++;

}

if (entro)

{

grupos++;

}

max\_cont = Math.Max(max\_cont, cont);

i++;

}

//Console.WriteLine(grupos);

if (grupos > 2)

{

List<int> indicesCerosSolos = new List<int>();

for (i = 0; i < arr.Length; i++)

{

if (i == 0)

{

if (arr[i] == 0 && i + 1 < arr.Length && arr[i + 1] == 1)

{

indicesCerosSolos.Add(i);

}

}

else if (i + 1 < arr.Length)

{

if (i - 1 >= 0 && arr[i - 1] == 1 && arr[i] == 0 && i + 1 < arr.Length && arr[i + 1] == 1)

{

indicesCerosSolos.Add(i);

}

}

else if (i == arr.Length - 1)

{

if (i - 1 >= 0 && arr[i - 1] == 1 && arr[i] == 0)

{

indicesCerosSolos.Add(i);

}

}

}

foreach (int indice in indicesCerosSolos)

{

i = indice-1;

int cont = 0;

while (i >= 0 && arr[i] == 1)

{

cont++;

i--;

}

i = indice+1;

while (i < arr.Length && arr[i] == 1)

{

cont++;

i++;

}

max\_cont = Math.Max(max\_cont, cont);

}

Console.WriteLine(max\_cont+1);

}

else if (grupos == 2)

{

i = 0;

while (i < arr.Length)

{

int cont = 0;

while (i < arr.Length && arr[i] == 1)

{

cont++;

i++;

}

max\_cont = Math.Max(max\_cont, cont);

i++;

}

Console.WriteLine(max\_cont + 1);

}

else if (grupos == 1)

{

i = 0;

while (i < arr.Length)

{

int cont = 0;

while (i < arr.Length && arr[i] == 1)

{

cont++;

i++;

}

max\_cont = Math.Max(max\_cont, cont);

i++;

}

Console.WriteLine(max\_cont);

}

else if (grupos == 0)

{

Console.WriteLine(0);

}

Console.ReadLine();

}

}

}