12th JOI Selection Final Selection Tasks (2013)

Authors: JCIOI (Japanese Committee for the IOI)

Translators: Jasmine Chua, France-ioi

License: Creative Commons Attributions-ShareAlike 4.0 International License (CC BY-SA 4.0)

**1. Illumination**

During the annual cultural festival held in JOI High School, the corridors are decorated with illuminations. The illuminations are made up of N light bulbs, which hang in one row from the west end of the corridor to the east. Some of the bulbs are turned on, while others are off.

In the storage room, there is a machine that is used to operate the light bulbs. The machine works by selecting a consecutive string of light bulbs and toggling their state. Thus, bulbs in this string which were initially turned on will go off, while the bulbs which were initially dark will turn on. However, this machine is very old, so it can only be used once.

The students of JOI High School think the best-looking pattern is where the on and off bulbs alternate. By using the machine only once, they tried to make the longest possible chain with an alternating pattern.

**EXAMPLE**

For example, the initial pattern of light bulbs from west to east is like this:



(○ represents a bulb that is on, ● represents a bulb that is off)

Now, if the machine is used to operate 4 consecutive light bulbs, from position 4 to position 7:



There will be a chain of 7 alternating bulbs, from position 2 to position 8.



Similarly if the machine is used to operate just 1 light bulb, at position 8:



There will be a chain of 7 alternating bulbs, from position 4 to position 10.



Since the machine can only be used once, the longest chain of alternating bulbs is 7.

**QUESTION**

Given information about the light bulbs, create a program to find out the maximum number of bulbs that can be made to form an alternating pattern, assuming the machine can only be used once.

**CONSTRAINTS**

Number of light bulbs in the illumination: 2 ≤ N ≤ 100 000

**INPUT**

In the first row, the integer N is written.

In the second row, a string of N characters is written with a space between them. Either '0' or '1' can be written, representing the light bulbs' state before using the machine. The ith character from the left, where 1 ≤ i ≤ N, represents the state of the light bulb in position i from the west. '1' represents a light bulb that is on, while 0 represents a light bulb that is off.

**OUTPUT**

Output one row with one integer: the maximum number of bulbs that can be made to form an alternating pattern.

**SCORING CRITERIA**

For 20% of the allocated points, fulfil N ≤ 500.

For 40% of the allocated points, fulfil N ≤ 2 000.

**INPUT/OUTPUT EXAMPLES**

|  |  |
| --- | --- |
| INPUT 1 | OUTPUT 1 |
| 10  1 1 0 0 1 0 1 1 1 0 | 7 |

This input example is the same as in the question.

|  |  |
| --- | --- |
| INPUT 2 | OUTPUT 2 |
| 10  1 0 0 0 0 1 0 1 0 1 | 8 |

The machine is used to operate just 1 light bulb, at position 4, to get a chain of 8 alternating bulbs.

|  |  |
| --- | --- |
| INPUT 3 | OUTPUT 3 |
| 5  1 1 0 1 1 | 5 |

The machine is used to operate light bulbs from position 2 to position 4, to get a complete chain of 5 alternating bulbs.

|  |  |
| --- | --- |
| INPUT 4 | OUTPUT 4 |
| 3  0 1 0 | 3 |

Note that the machine does not need to be used to achieve the output.

**2. Take the 'IOI' Train**

IOI Kingdom is constructing a new railway. The trains running on IOI Kingdom's railways are composed of several carriages linked together, and there are 2 types of carriages, I and O. Each carriage can only connect to another carriage that is of a different type. Furthermore, due to the location of the driver's seat, the carriages at both ends of the train must be of type I. Trains can be represented by a series of letters representing which carriages they are made up of. For example, train IOIOI has a total of 5 carriages, while train I only has 1 carriage. According to the rules above, train OIOI or train IOOI are not possible.

Several train carriages are currently stored in 2 garages. The carriages in both garages are lined up in 1 row. When building a train, a carriage is brought out of the garage, where it is joined to other carriages. Only the carriage closest to the garage door can be brought out, and engineers are free to choose which garage to bring a carriage out from.

Before building a train, carriages can be brought out of the garage and sent off to a holding track. Once sent off, those carriages cannot be used to build trains. Also, once the construction of a train has started, carriages cannot be sent off until the construction of the train has finished.

There is no need to use all the carriages in the garages to build a train. In other words, some carriages may be left unused in the garages after the building process.

There are many passengers who wish to use the train in IOI Kingdom, so the engineers hope to build the longest train possible.

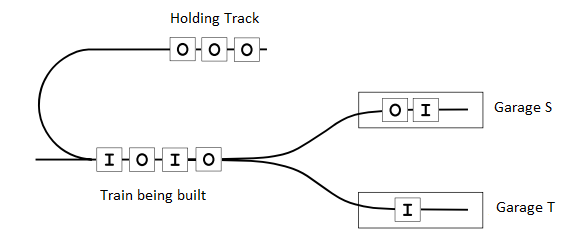


Diagram showing a train being built. During this process, carriages in the garages cannot be sent off to the holding track. This diagram corresponds to input example 1.

**QUESTION**

Given information about the carriages in the garages, create a program to find out the maximum length of the train that can be built. The carriages in each garage is represented by a string of characters, made up of 'I' and 'O'. Information about the two garages are given, a string of M characters in Garage S, and a string of N characters in Garage T. Each character represents 1 carriage, which represents the carriage type. The first character in the string represents the carriage closest to the garage door, while the last character represents the carriage at the very back of the garage.

**CONSTRAINTS**

Length of character string S: 1 ≤ M ≤ 2 000

Length of character string T: 1 ≤ N ≤ 2 000

**INPUT**

In the first row, 2 integers M and N are written with a space between them.

In the second row, character string S is written.

In the third row, character string T is written.

**OUTPUT**

Output the maximum number of carriages that can be used to build the train in one row. Output 0 if no trains can be built.

**SCORING CRITERIA**

For 20% of the allocated points, fulfil M ≤ 10 and N ≤ 10.

For 50% of the allocated points, fulfil M ≤ 50 and N ≤ 50.

**INPUT/OUTPUT EXAMPLES**

|  |  |
| --- | --- |
| INPUT 1 | OUTPUT 1 |
| 5 5  OIOOI  OOIOI | 7 |

First, send off the first carriage in Garage S and the first 2 carriages in Garage T to the holding track. Then, bring out the carriages in the following order: S, S, T, S, S, T, T, to make train IOIOIOI which is 7 carriages long.

Alternatively, send off the first carriage in Garage S and the first 2 carriages in Garage T to the holding track. Then, bring out the carriages in the following order: T, T, S, S, T, S, S, to make a train which is also 7 carriages long. Since this is the longest train that can be built, the output is 7.

|  |  |
| --- | --- |
| INPUT 2 | OUTPUT 2 |
| 5 9  IIIII  IIIIIIIII | 1 |

Note that only train I, with 1 carriage, can fulfil the given conditions.

**3. Modern Mansion**

You are lost in a huge mansion. The mansion is made up of square rooms laid out in a grid, with M rooms stretching from East to West, and N rooms stretching from North to South, for a total number of M × N rooms. The room that is xth from the West, where 1 ≤ x ≤ M, and yth from the South, where 1 ≤ y ≤ N, is labeled as Room (x,y).

Each room is joined by a door to the adjacent room in each of 4 directions. Some doors are open so you can pass through, while some doors are closed and locked. You take 1 minute to move from the center of 1 room to the center of the next room. Also, there is a switch located in the center of some rooms. If you press and hold the switch for 1 minute, all doors in the mansion will switch from open to locked, or locked to open.

Currently, all doors opening towards the East and West are closed and locked, while all doors opening towards the North and South are open. You are now in Room (1,1), and wish to get to Room (M,N) in the shortest possible time.

**QUESTION**

You are given the size of the mansion M and N, as well as the position of K rooms (X1,Y1), (X2,Y2),..., (XK,YK) where switches are located. All doors opening towards the East and West are closed and locked, while all doors opening towards the North and South are open. Create a program to find out the minimum amount to time needed to move from Room (1,1) to Room (M,N). Indicate in your answer if it is impossible to get to Room (M,N).

**CONSTRAINTS**

Number of rooms in the mansion in the East-West direction: 2 ≤ M ≤ 100 000

Number of rooms in the mansion in the North-South direction: 2 ≤ N ≤ 100 000

Number of rooms with switches: 1 ≤ K ≤ 200 000

Position of room with switch in the East-West direction: 1 ≤ Xi ≤ M

Position of room with switch in the North-South direction: 1 ≤ Yi ≤ N

**INPUT**

In the first row, 3 integers are written with a space between them.

* M, representing the number of rooms in the mansion in the East-West direction.
* N, representing the number of rooms in the mansion in the North-South direction.
* K, representing the number of rooms with switches.

In the next K rows, in the ith row, where 1 ≤ i ≤ K, 2 integers are written with a space between them.

* Xi and Yi, representing Room (Xi,Yi) where a switch is located.
* Ensure that the K groups of (X1,Y1), (X2,Y2),..., (XK,YK) are all different.

**OUTPUT**

Output the minimum time needed to move in one row. Output -1 if Room (M,N) cannot be reached.

**SCORING CRITERIA**

For 20% of the allocated points, fulfil M ≤ 1 000 and N ≤ 1 000.

For 30% of the allocated points, fulfil K ≤ 2 000.

For 50% of the allocated points, fulfil at least 1 of the 2 constraints: for M and N, or for K. No points are allocated to fulfilling both conditions.

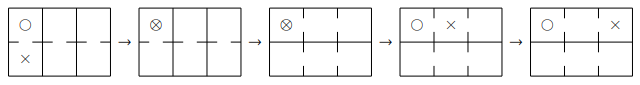
**INPUT/OUTPUT EXAMPLES**

|  |  |
| --- | --- |
| INPUT 1 | OUTPUT 1 |
| 3 2 1  1 2 | 4 |

In this example, the minimum time needed to move from Room (1,1) to Room (3,2) is 4 minutes.

1. Move to Room (1,2).
2. Press and hold the switch in Room (1,2).
3. Move to Room (2,2).
4. Move to Room (3,2).

The diagram below shows this movement. The right side of the diagram is East, the top is North, X is your position, and O represents a switch.

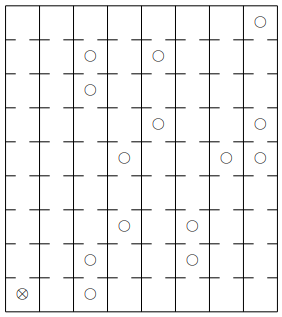


|  |  |
| --- | --- |
| INPUT 2 | OUTPUT 2 |
| 3 2 1  2 1 | -1 |

In this example, you cannot reach Room (3,2).

|  |  |
| --- | --- |
| INPUT 3 | OUTPUT 3 |
| 8 9 15  3 1  3 2  3 7  3 8  1 1  4 5  4 3  5 6  5 8  6 3  6 2  7 5  8 9  8 6  8 5 | 25 |

The diagram below shows the initial set-up inside the mansion. Note that switches can also be found in Room (1,1) and Room (M,N).



**4. JOIOI Tower**

JOIOI Tower is a 1-player game that makes use of disks.

There are various disks, each labeled with 1 character: J, O, or I. The disks are all of different diameters, and at the start of the game, they are stacked in order, from largest at the bottom to smallest on top. Your task is to use these disks to create as many mini JOIOI Towers as you can. A mini JOIOI Tower must be at least 3 disks tall, and when read from smallest disk to largest, it should spell JOI or IOI. Each disk can only be used once.

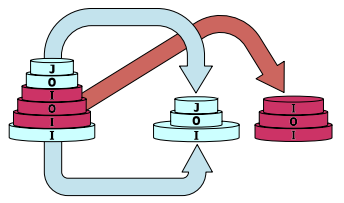


Diagram showing 2 mini JOIOI Towers built from JOIOII

**QUESTION**

You are given character string S, with N characters, representing the characters written on the disks from smallest to largest. Create a program to find out the maximum number of mini JOIOI Towers you can build from these disks.

**CONSTRAINTS**

Length of character string S: 1 ≤ N ≤ 1 000 000

**INPUT**

In the first row, 1 integer is written.

* N, representing the length of character string S.

In the second row, character string S is written.

**OUTPUT**

Output the maximum number of mini JOIOI Towers that can be built in one row.

**SCORING CRITERIA**

For 10% of the allocated points, fulfil N ≤ 15.

For 30% of the allocated points, fulfil N ≤ 50.

For 50% of the allocated points, fulfil N ≤ 3 000.

**INPUT/OUTPUT EXAMPLES**

|  |  |
| --- | --- |
| INPUT 1 | OUTPUT 1 |
| 6  JOIIOI | 2 |

JOIIOI contains the disks needed for both JOI and IOI, so 2 mini JOIOI Towers can be built.

|  |  |
| --- | --- |
| INPUT 2 | OUTPUT 2 |
| 5  JOIOI | 1 |

JOIIOI contains the disks needed for both JOI and IOI, but since each disk can only be used once, only 1 mini Tower can be built.

|  |  |
| --- | --- |
| INPUT 3 | OUTPUT 3 |
| 6  JOIOII | 2 |

This input example is the same as the question.

|  |  |
| --- | --- |
| INPUT 4 | OUTPUT 4 |
| 15  JJOIIOOJOJIOIIO | 4 |

**5. Bubble Sort**

Bubble sort is a type of sorting algorithm. Imagine you want to sort A, a string with N numbers, in ascending order. Bubble sort compares a pair of adjacent numbers and swaps them if they are in the wrong order. The algorithm will go through the entire row of numbers. In other words, if Ai > Ai+1, where i = 1, 2,..., N − 1, these two numbers will swap positions. After going through the row (N − 1) times, the numbers will be sorted in ascending order.

The Swap number is the total number of swaps the above algorithm needs to make to put string A in the correct order. (The bubble sort algorithm and its implementations can have minor differences, such as loop order and range, and termination conditions. However, the swap number should not change despite these differences.)

For example, the program below is written in C programming language, instructing a bubble sort to be done on row a with a length of n characters.

void bubble\_sort(int \*a, int n) {

int i, j;

for (i = 0; i < n - 1; ++i) {

for (j = 0; j < n - 1; ++j) {

if (a[j] > a[j + 1]) {

/\* Perform 1 swap on the 3 rows below \*/

int x = a[j];

a[j] = a[j + 1];

a[j + 1] = x;

}

}

}

}

**QUESTION**

You are given a sequence A, with N numbers. Before running bubble sort, you can do one single swap can between 2 numbers anywhere in the sequence. The new sequence is called A'. Create a program to find out the minimum number of swaps needed to bubble sort A' into the correct order, after making this initial swap. (Note that the initial swap **NEED NOT** be between 2 adjacent numbers.)

**CONSTRAINTS**

Length of sequence A: 1 ≤ N ≤ 100 000

Value of numbers in sequence A: 1 ≤ Ai ≤ 1 000 000 000

**INPUT**

In the first row, 1 integer is written.

* N, representing the length of sequence A.

For the next N rows, in the ith row, where 1 ≤ i ≤ N, 1 integer is written.

* Ai, representing the ith number in sequence A.

**OUTPUT**

Output one row, with the minimum number of swaps needed to bubble sort A' into the correct order.

**SCORING CRITERIA**

For 10% of the allocated points, fulfil N ≤ 1 000, and Ai ≠ Aj, where 1 ≤ i < j ≤ N.

For 30% of the allocated points, fulfil N ≤ 5 000, and Ai ≠ Aj, where 1 ≤ i < j ≤ N.

For 80% of the allocated points, fulfil Ai ≠ Aj, where 1 ≤ i < j ≤ N.

**INPUT/OUTPUT EXAMPLES**

|  |  |
| --- | --- |
| INPUT 1 | OUTPUT 1 |
| 5  10  3  6  8  1 | 0 |

By swapping the positions of 1 and 10 in the sequence A, the new sequence A' is already in the correct order, so the bubble sort swap number is 0.

|  |  |
| --- | --- |
| INPUT 2 | OUTPUT 2 |
| 5  3  1  7  9  5 | 2 |

By swapping the positions of 7 and the 2nd 5 in sequence A, sequence A' reads 5,3,1,5,9,7. Bubble sort will make 2 swaps.

|  |  |
| --- | --- |
| INPUT 3 | OUTPUT 3 |
| 3  1  2  3 | 1 |

Even without making an initial swap, bubble sort still needs to make swaps to put the sequence A' in the correct order.