Juanito is lost in the woods and apparently the only way out is through a marsh in the middle of the trees. In the mud, there are a number of stones in the water to use for jumping. Juanito can jump on any stone having a number that is equal to the number of the current stone plus 1 or 2. He must avoid the pits in the water.

Determine the minimum number of jumps it will take to Juanito to jump from his starting position to the last stone. It is always possible to escape from the marsh.

Juanito will get an array of stones/pits numbered 0 if they are safe or 1 if that’s a pit and must be avoided. For example, C=[0,1,0,0,0,1,0] indexed from 0…6.

The number on each stone is its index in the list so he must avoid the pits at indexes 1 and 5. He can follow the next two paths: 0-> 2->4->6 or 0->2->3->4->6. The first path takes 3 jumps while the second one takes 4.

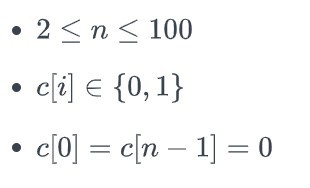
**Function Description**

Complete the jumpOnMarsh function. It should return the minimum number of jumps required.

**Input Parameters**

The function receives a parameter that contains “n” binary integers describing stones/pits in the water C[i] where 0<=i<n.

**Constraints**



**Output Format**

Print the minimum number of jumps needed to win the game.

|  |  |
| --- | --- |
| **Sample Input 0**  0 0 1 0 0 1 0  **Sample Output 0**  4  **Explanation 0:** Juanito must avoid C[2] and C[5]. He can win the game with a minimum of 4 jumps: | **Graphical user interface, application  Description automatically generated** |
| **Sample Input 1**  0 0 0 0 1 0  **Sample Output 1**  3  **Explanation 1:** The only pit to avoid is C[4]. Juanito can win the game in 3 jumps: | **Graphical user interface, application, icon  Description automatically generated** |