Java 1D Array (Part 2)

Let's play a game on an array! You're standing at index 0 of an n-element array named $game$. From some index i (where $0 \leq i < n$), you can perform one of the following moves:

- Move Backward: If cell $i-1$ exists and contains a 0, you can walk back to cell $i-1$.
- Move Forward:
  - If cell $i+1$ contains a zero, you can walk to cell $i+1$.
  - If cell $i+leap$ contains a zero, you can jump to cell $i+leap$.
  - If you're standing in cell $n-1$ or the value of $i+leap \geq n$, you can walk or jump off the end of the array and win the game.

In other words, you can move from index i to index $i+1$, $i-1$, or $i+leap$ as long as the destination index is a cell containing a 0. If the destination index is greater than $n-1$, you win the game.

Given $leap$ and $game$, complete the function in the editor below so that it returns true if you can win the game (or false if you cannot).

Input Format

The first line contains an integer, q, denoting the number of queries (i.e., function calls).
The $2 \cdot q$ subsequent lines describe each query over two lines:

1. The first line contains two space-separated integers describing the respective values of n and $leap$.
2. The second line contains n space-separated binary integers (i.e., zeroes and ones) describing the respective values of $game_0, game_1, \ldots, game_{n-1}$.

Constraints

- $1 \leq q \leq 5000$
- $2 \leq n \leq 100$
- $0 \leq leap \leq 100$
- It is guaranteed that the value of $game[0]$ is always 0.

Output Format

Return true if you can win the game; otherwise, return false.

Sample Input

```
4
5 3
0 0 0 0 0
6 5
0 0 0 1 1 1
6 3
0 0 1 1 1 0
3 1
0 1 0
```

Sample Output

```
YES
YES
NO
NO
```

Explanation

We perform the following $q = 4$ queries:

1. For $game = [0, 0, 0, 0, 0]$ and $leap = 3$, we can walk and/or jump to the end of the array because every cell contains a 0. Because we can win, we return true.
2. For $game = [0, 0, 0, 1, 1, 1]$ and $leap = 5$, we can walk to index 1 and then jump $i + leap = 1 + 5 = 6$ units to the end of the array. Because we can win, we return true.
3. For $game = [0, 0, 1, 1, 1, 0]$ and $leap = 3$, there is no way for us to get past the three consecutive ones. Because we cannot win, we return false.
4. For $game = [0, 1, 0]$ and $leap = 1$, there is no way for us to get past the one at index 1. Because we cannot win, we return false.

Author Shafaet
Difficulty Medium
Max Score 25
Submitted By 35733

NEED HELP?

View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

MORE DETAILS

Download problem statement
Download sample test cases
Suggest Edits

Change Theme Java 7

```java
import java.util.*;

public class Solution {

    public static boolean canWin(int leap, int[] game) {
        // Return true if you can win the game; otherwise, return false.
    }

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int q = scan.nextInt();
        while (q-- > 0) {
            int n = scan.nextInt();
            int leap = scan.nextInt();

            int[] game = new int[n];
            for (int i = 0; i < n; i++) {
```

```
            game[i] = scan.nextInt();
        }

        System.out.println( (canWin(leap, game)) ? "YES" : "NO" );
    }
    scan.close();
  }
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code