

6899. Maximum Number of Jumps to Reach the Last Index

My Submissions (/contest/weekly-contest-353/problems/maximum-number-of-jumps-to-reach-the-last-index/submissions/)

Back to Contest (/contest/weekly-contest-353/)

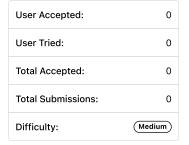
You are given a **0-indexed** array nums of n integers and an integer target.

You are initially positioned at index 0. In one step, you can jump from index i to any index j such that:

- $0 \le i < j < n$
- -target <= nums[j] nums[i] <= target

Return the **maximum number of jumps** you can make to reach index n - 1.

If there is no way to reach index n - 1, return -1.



Example 1:

```
Input: nums = [1,3,6,4,1,2], target = 2
Output: 3
Explanation: To go from index 0 to index n-1 with the maximum number of jumps, you can perform the following jumping sequence
- Jump from index 0 to index 1.
- Jump from index 1 to index 3.
- Jump from index 3 to index 5.
It can be proven that there is no other jumping sequence that goes from 0 to n - 1 with more than 3 jumps. Hence, the answer is
```

Example 2:

```
Input: nums = [1,3,6,4,1,2], target = 3
Output: 5
Explanation: To go from index 0 to index n-1 with the maximum number of jumps, you can perform the following jumping sequence
- Jump from index 0 to index 1.
- Jump from index 1 to index 2.
- Jump from index 2 to index 3.
- Jump from index 3 to index 4.
- Jump from index 4 to index 5.
It can be proven that there is no other jumping sequence that goes from 0 to n − 1 with more than 5 jumps. Hence, the answer is
```

Example 3:

```
Input: nums = [1,3,6,4,1,2], target = 0
Output: -1
Explanation: It can be proven that there is no jumping sequence that goes from 0 to n-1. Hence, the answer is -1.
```

Constraints:

```
• 2 <= nums.length == n <= 1000
```

- $-10^9 \le nums[i] \le 10^9$
- $0 \le target \le 2 * 10^9$

```
₫ C
JavaScript
1 ▼ function Deque() {
2
        let m = \{\}, first = 0, last = -1;
3
        return { unshift, shift, push, pop, front, back, size, show }
4
        function push(...args) {
5
            let i = 0;
6 •
            if (size() == 0) {
7
                first = last = 0;
8
                m[first] = args[i++];
9
            for (; i < args.length; i++) m[++last] = args[i];</pre>
```

```
11
12 •
         function unshift(...args) {
13
             let i = 0;
14 ▼
             if (size() == 0) {
15
                 first = last = 0;
16
                 m[first] = args[i++];
17
18
             for (; i < args.length; i++) m[--first] = args[i];</pre>
19
20 •
        function pop() {
             let res = m[last];
21
22
             delete m[last];
23
             last--;
             return res;
24
25
26 •
        function shift() {
27
             let res = m[first];
             delete m[first];
28
29
             first++;
30
             return res;
31
32 •
         function front() {
33
             return m[first];
34
35 ▼
         function back() {
36
             return m[last];
37
         function size() {
38 1
             if (first > last) return 0;
39
40
             return last - first + 1;
41
42 •
         function show() {
43
             let a = Object.keys(m), res = [];
44
             a.sort((x, y) \Rightarrow x - y);
45
             for (const k of a) res.push(m[k]);
46
             return res;
47
        }
    }
48
49
    const maximumJumps = (a, t) \Rightarrow \{
50 ▼
51
        let n = a.length, g = new Map();
52 ▼
         for (let i = 0; i < n; i++) {
             for (let j = i + 1; j < n; j++) {
53 ▼
                 if (Math.abs(a[i] - a[j]) <= t) {</pre>
54 ▼
55
                      if (!g.has(i)) g.set(i, new Set());
56
                      g.get(i).add(j);
57
                 }
58
             }
59
        let q = new Deque(), dis = Array(n).fill(Number.MIN_SAFE_INTEGER)
60
        dis[0] = 0;
61
        q.push(0);
62
        while (q.size()) {
63 ▼
             let cur = q.shift();
64
65
             let d = g.get(cur) || [];
66 •
             for (const next of d) {
67 ▼
                  if (next > cur) {
                      if (dis[next] < dis[cur] + 1) {</pre>
68 ▼
69
                          dis[next] = dis[cur] + 1;
70
                          q.push(next);
71
                      }
72
                 }
73
             }
74
        }
        return dis[n - 1] == Number.MIN_SAFE_INTEGER ? -1 : dis[n - 1];
75
76
    };
```

☐ Custom Testcase

Use Example Testcases

Submit
 Su

Run

Submission Result: Accepted (/submissions/detail/989802301/) ?

More Details > (/submissions/detail/989802301/)

Share your acceptance!