**Sprint Training**

**QUESTION DESCRIPTION**

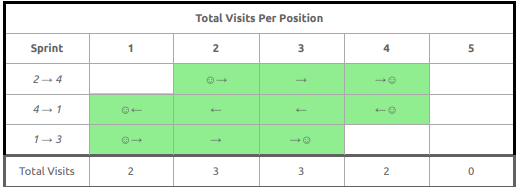
Pat is an ordinary kid who works hard to be a great runner. As part of training, Pat must run sprints of different intervals on a straight trail. The trail has numbered markers that the coach uses as goals. Pat's coach provides a list of goals to reach in order. Each time Pat starts at, stops at, or passes a marker it is considered a visit. Determine the lowest numbered marker that is visited the most times during Pat's day of training.

**Example**

n = 5

sprints = [2, 4, 1, 3]

if the number of markers on the trail, n = 5, and assigned sprints = [2, 4, 1, 3], Pat first sprints from position 2 → 4. The next sprint is from position 4 → 1, and then 1 → 3. A marker numbered position p is considered to be visited each time Pat either starts or ends a sprint there and each time it is passed while sprinting. The total number of visits to each position in the example is calculated like so:



Pat has visited markers 2 and 3 a total of 3 times each. Since 2 < 3, the lowest numbered marker that is Visited the most times during Pat's day of training is 2.

**Function Description**

Complete the function getMostVisited in the editor below.

getMostVisited has the following parameter(s):

* int n: an integer denoting the number of markers along the trail
* int sprints[m]: an array of integers denoting the sequence of markers to reach, beginning at the marker

shown in sprints[0].

**Returns:**

int: an integer denoting Pat's most visited position on the trail after performing all m − 1 sprints. If there are multiple such answers, return the smallest one.

**Constraints**

* 1 ≤ n ≤ 10
* 2 ≤ m ≤ 10
* 1 ≤ sprints[i] ≤ m (where 0 ≤ i < m)
* sprints[i-1] ≠ sprints[i] (where 0 < i < m)

**Input Format for Custom Testing**

* Input from stdin will be processed as follows and passed to the function.
* The first line contains an integer n, the number of markers along the path.
* The second line contains an integer m, the number of markers in the list of goals.
* The next m lines each contain an element sprints[i] where 0 ≤ i < m.

**Sample Case 0**

**Sample Input 0**

STDIN Function Parameters

10 → n = 10

4 → sprints[ ] size m = 4

1 → sprints = [1, 5, 10, 3]

5

10

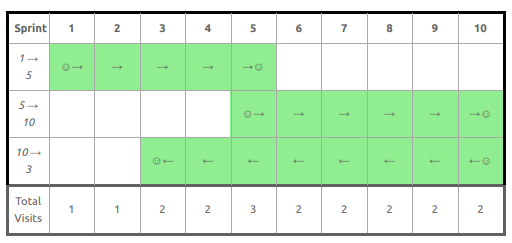
3

**Sample Output 0**

**5**

**Explanation 0**

**Given sprints = [1, 5, 10, 3], Pat performs the following sequence of sprints:**



In the table above, Pat visited marker 5 the most.

Sample Case 1

**Sample Input 1**

STDIN Function Parameters

5 → n = 5

2 → sprints[ ] size m = 2

1 → sprints = [1, 5]

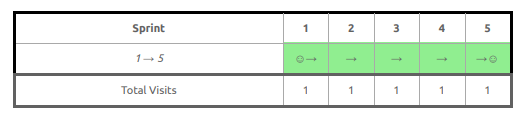
5

**Sample Output 1**

1

**Explanation 1**

Given sprints = [1, 5], Pat performs the following sprint:



In the table above, every marker is visited the same number of times. Return the smallest of these, which is 1.

**Sample Case 2**

**Sample Input 2**

STDIN Function Parameters

9 → n = 9

4 → sprints[ ] size m = 4

9 → sprints = [9, 7, 3, 1]

7

3

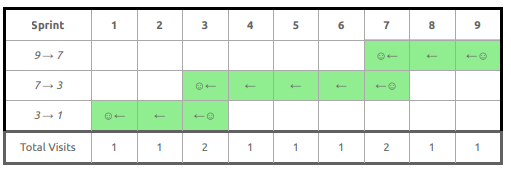
1

**Sample Output 2**

3

**Explanation 2**

Given sprints = [9, 7, 3, 1], Pat performs the following sequence of sprints:



In the table above, Pat visited positions 3 and 7 the most. Return the smallest of these, which is 3.

**CANDIDATE ANSWER**

Language used: Java 8

class Result {

/\*

\* Complete the 'getMostVisited' function below.

\*

\* The function is expected to return an INTEGER.

\* The function accepts following parameters:

\* 1. INTEGER n

\* 2. INTEGER\_ARRAY sprints

\*/

public static int getMostVisited(int n, List<Integer> sprints) {

// Write your code here

}

}