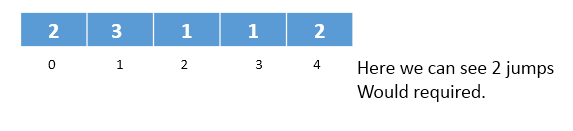
# Array Jumping

## Problem

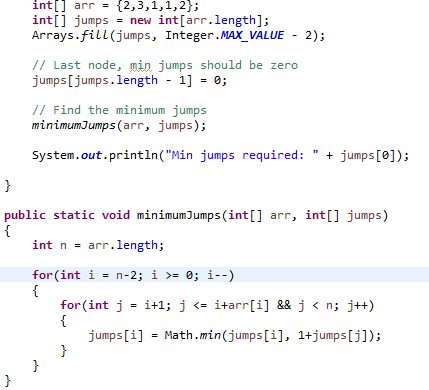
There is an array with values represents number of maximum positions from there can jump. We need to identify the minimum jumps we can make to reach last node in optimized way.



## Dynamic programming

Find the minimum jumps required to reach last node, and start from last node itself, so any before node the minimum jumps required should be

Jump[i] = From (I -> pos to -> i+arr[i]) Min(arr[i], 1+jumps[j])



### Time complexity with dynamic programming

Here the worst case time complexity is **O (n2)**

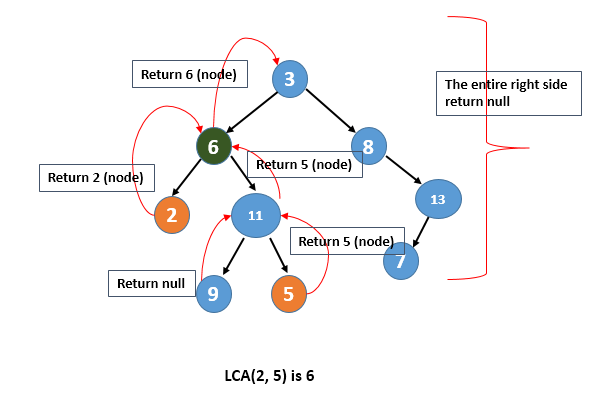
### Segment tree

Here the inner for loop does is just to find the minimum jumps from i+1 to last node. Which can be solved in Log n time using segment tree.

Hence the total time complexity will be **O (n Log n)**

# Least Common Ancestor

Here to find out the least common node between two pair of nodes.



### Program

