Given a singly linked list, return a random node's value from the linked list. Each node must have the **same probability** of being chosen.

Follow up:
What if the linked list is extremely large and its length is unknown to you? Could you solve this efficiently without using extra space?

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode(int x) { val = x; }
 * }
 */
class Solution {

    ListNode head;
    Random rand;
    /** @param head The linked list's head.
        Note that the head is guaranteed to be not null, so it contains at least one node. */
    public Solution(ListNode head) {
        this.head = head;
        this.rand = new Random();
    }

    /** Returns a random node's value. */
    public int getRandom() {
        ListNode node = head;
        ListNode cur = null;
        for (int i = 0; node != null; i++, node = node.next) {
            if (rand.nextInt() % (i+1) == 0) {
                cur = node;
            }
        }
        return cur.val;
    }
}

/**
 * Your Solution object will be instantiated and called as such:
 * Solution obj = new Solution(head);
 * int param_1 = obj.getRandom();
 */
```