Recursion: Leap lists

Help Center

The solution is here.
For the leap list problem we are given a list where the last element is 0 (the goal element) and every other element is a positive integer. For example:

`[1, 2, 3, 3, 3, 1, 0]`

Given a starting index `n` for a player and a leap budget `k` we are asked to determine whether a player may "leap" from the starting index to the goal index in `k` or less leaps. By the rules of leap list however, if the player is currently at index `i` the player may only next leap to index `i+list[i]` (a right leap) or `i-list[i]` (a left leap) assuming that the leap would land on a valid index of the list. For example, if the player started at index 4 (which contains the value 3) their next leap could be to "leap left" to index 1 (since 4-3=1) but not to "leap right" to 7 (since 4+3=7) because this would be leaping beyond the boundary of the list. Likewise, after moving to index 1 (with value 2) the player then could then "leap right" to index 3 but not "leap left" to index -1. If the player found themselves on index 5 they could leap right to index 6 (to the goal!) or left to index 4.

Exercise 1 - Max Leaps Pseudocode

Write the pseudocode for a recursive algorithm **IsGoalReachableMax** that determines whether it is possible to leap from the starting position to the goal in a given number of leaps (or fewer). Once you are satisfied with your pseudo-code, you are welcome to examine our pseudo-code.

Exercise 2 - Max Leaps Python

Write the recursive function:

```
def is_goal_reachable_max(leap_list, start_index, max_leaps):
    """
    Determines whether goal can be reached in at most max_leaps leaps.

    Arguments:
    leap_list - the leap list game board.
    start_index - the starting index of the player.
    max_leaps - the most number of leaps allowed before the player loses.

    Returns:
    True if goal is reachable in max_leap or less leaps.  False if goal is not reachab
le in max_leap or fewer leaps.
    """
```

Assert your function produces the following output:

```
>>> is_goal_reachable_max([1, 2, 3, 3, 3, 1, 0], 0, 3)
True
```

```
>>> is_goal_reachable_max([1, 2, 3, 3, 3, 1, 0], 0, 2)
False
>>> is_goal_reachable_max([1, 2, 3, 3, 3, 1, 0], 4, 3)
True
>>> is_goal_reachable_max([1, 2, 3, 3, 3, 1, 0], 4, 2)
False
>> is_goal_reachable_max([2, 1, 2, 2, 2, 0], 1, 5)
False
>>> is_goal_reachable([2, 1, 2, 2, 2, 0], 3, 1)
True
```

Exercise 3 - Unlimited Leaps Pseudocode

Write the pseudocode for a recursive algorithm **IsGoalReachable** that determines whether it is possible to leap from the starting position to the goal in any number of leaps. Once you are satisfied with your pseudo-code, you are welcome to examine our pseudo-code.

Exercise 4 - Unlimited Leaps Python

Write the recursive function:

```
def is_goal_reachable(leap_list, start_index):
    """
    Determines whether goal can be reached in any number of leaps.

    Arguments:
    leap_list - the leap list game board.
    start_index - the starting index of the player.

    Returns:
    True if goal is reachable.  False if goal is not reachable.
    """
```

Assert your function produces the following output:

```
>>> is_goal_reachable([1, 2, 3, 3, 3, 1, 0], 0)
True
>>> is_goal_reachable([1, 2, 3, 3, 3, 1, 0], 4)
True
>> is_goal_reachable([2, 1, 2, 2, 2, 0], 1)
False
>>> is_goal_reachable([2, 1, 2, 2, 2, 0], 3)
True
>>> is_goal_reachable([3, 6, 4, 1, 3, 4, 2, 5, 3, 0], 0)
True
```

Created Sun 24 Aug 2014 12:42 AM BST

Last Modified Wed 10 Sep 2014 9:25 PM BST