Problem 1265: Print Immutable Linked List in Reverse

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

You are given an immutable linked list, print out all values of each node in reverse with the help of the following interface:

ImmutableListNode

: An interface of immutable linked list, you are given the head of the list.

You need to use the following functions to access the linked list (you

can't

access the

ImmutableListNode

directly):

ImmutableListNode.printValue()

: Print value of the current node.

ImmutableListNode.getNext()

: Return the next node.

The input is only given to initialize the linked list internally. You must solve this problem without modifying the linked list. In other words, you must operate the linked list using only the mentioned APIs.

Example 1:

Input:

head = [1,2,3,4]

Output:

[4,3,2,1]

Example 2:

Input:

head = [0,-4,-1,3,-5]

Output:

[-5,3,-1,-4,0]

Example 3:

Input:

head = [-2,0,6,4,4,-6]

Output:

[-6,4,4,6,0,-2]

Constraints:

The length of the linked list is between

[1, 1000]

.

The value of each node in the linked list is between

[-1000, 1000]

.

Follow up:

Could you solve this problem in:

Constant space complexity?

Linear time complexity and less than linear space complexity?

Code Snippets

C++:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public:
* void printValue(); // print the value of the node.
* ImmutableListNode* getNext(); // return the next node.
* };
*/

class Solution {
public:
void printLinkedListInReverse(ImmutableListNode* head) {

}
};
```

Java:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* interface ImmutableListNode {
* public void printValue(); // print the value of this node.
* public ImmutableListNode getNext(); // return the next node.
* };
*/

class Solution {
public void printLinkedListInReverse(ImmutableListNode head) {

}
}
```

Python3:

```
# """
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
# """
# class ImmutableListNode:
# def printValue(self) -> None: # print the value of this node.
# def getNext(self) -> 'ImmutableListNode': # return the next node.

class Solution:
def printLinkedListInReverse(self, head: 'ImmutableListNode') -> None:
```

Python:

```
# """
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
# """
# class ImmutableListNode(object):
# def printValue(self): # print the value of this node.
# . """
# :rtype None
# """
#
# def getNext(self): # return the next node.
# . """
# :rtype ImmutableListNode
```

```
# """

class Solution(object):
def printLinkedListInReverse(self, head):
"""
:type head: ImmutableListNode
:rtype: None
"""
```

JavaScript:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* function ImmutableListNode() {
* @ return {void}
* this.printValue = function() { // print the value of this node.
* ...
* };
*
* @return {ImmutableListNode}
* this.getNext = function() { // return the next node.
* ...
* };
* };
*/

/**
* @param {ImmutableListNode} head
* @return {void}
*/
var printLinkedListInReverse = function(head) {

};
```

TypeScript:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation
* class ImmutableListNode {
* printValue() {}
```

```
*
* getNext(): ImmutableListNode {}
* }
*/

function printLinkedListInReverse(head: ImmutableListNode) {

};
```

C#:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public void PrintValue(); // print the value of this node.
* public ImmutableListNode GetNext(); // return the next node.
* }
*/

public class Solution {
public void PrintLinkedListInReverse(ImmutableListNode head) {

}
}
```

C:

```
/**
* Definition for ImmutableListNode.
* struct ImmutableListNode {
* struct ImmutableListNode* (*getNext)(struct ImmutableListNode*); // return
the next node.
* void (*printValue)(struct ImmutableListNode*); // print the value of the
node.
* };
*/

void printLinkedListInReverse(struct ImmutableListNode* head) {

}
```

Go:

```
/* Below is the interface for ImmutableListNode, which is already defined for
you.
*
* type ImmutableListNode struct {
*
* }
*
* func (this *ImmutableListNode) getNext() ImmutableListNode {
* // return the next node.
* }
*
* func (this *ImmutableListNode) printValue() {
* // print the value of this node.
* }
*/

func printLinkedListInReverse(head ImmutableListNode) {

}
```

Kotlin:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* fun getNext(): ImmutableListNode? {} // return the next node.
* fun printValue() {} // print the value of this node.
* };
*/

class Solution {
fun printLinkedListInReverse(head:ImmutableListNode?) {

}
}
```

Swift:

```
/**
* Definition for ImmutableListNode.
```

```
* public class ImmutableListNode {
* public func printValue() {}
* public func getNext() -> ImmutableListNode? {}
* }
*/

class Solution {
func printLinkedListInReverse(_ head: ImmutableListNode?) {

}
}
```

Ruby:

```
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
#
# class ImmutableListNode
# def printValue()
# . print the value of this node.
# def end
# """
#
# def getNext()
# . return the next node.
# end
# end

def printLinkedListInReverse(head)

end
```

PHP:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public function printValue() {} // print the value of this node.
* public function getNext() {} // return the next node.
* };
*/
```

```
class Solution {
/**
* @param ImmutableListNode $head
* @return void
*/
function printLinkedListInReverse($head) {

}
}
```

Scala:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode{
* def printValue(): Unit = {} // print the value of this node.
* def getNext(): ImmutableListNode = {} // return the next node.
* };
*/

object Solution {
def printLinkedListInReverse(head: ImmutableListNode): Unit = {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
```

```
*/

/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public:
* void printValue(); // print the value of the node.
* ImmutableListNode* getNext(); // return the next node.
* };
*/

class Solution {
public:
void printLinkedListInReverse(ImmutableListNode* head) {

}
};
```

Java Solution:

```
/**
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* interface ImmutableListNode {
* public void printValue(); // print the value of this node.
* public ImmutableListNode getNext(); // return the next node.
* };
*/

class Solution {
```

```
public void printLinkedListInReverse(ImmutableListNode head) {

}
}
```

Python3 Solution:

```
"""
Problem: Print Immutable Linked List in Reverse
Difficulty: Medium
Tags: array, linked_list, stack

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

# """
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
# """
# class ImmutableListNode:
# def printValue(self) -> None: # print the value of this node.
# def getNext(self) -> 'ImmutableListNode': # return the next node.

class Solution:
def printLinkedListInReverse(self, head: 'ImmutableListNode') -> None:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
# """
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
# """
# class ImmutableListNode(object):
# def printValue(self): # print the value of this node.
# . """
# :rtype None
# """
```

```
#
# def getNext(self): # return the next node.
# . """
# :rtype ImmutableListNode
# """

class Solution(object):
def printLinkedListInReverse(self, head):
"""
:type head: ImmutableListNode
:rtype: None
"""
```

JavaScript Solution:

```
/**
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* function ImmutableListNode() {
* @ return {void}
* this.printValue = function() { // print the value of this node.
* ...
* };
*
* @return {ImmutableListNode}
* this.getNext = function() { // return the next node.
* ...
* };
* };
*/
```

```
/**
* @param {ImmutableListNode} head
* @return {void}
*/
var printLinkedListInReverse = function(head) {

};
```

TypeScript Solution:

```
/**
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation
* class ImmutableListNode {
* printValue() {}
*
* getNext(): ImmutableListNode {}
* }
*/

function printLinkedListInReverse(head: ImmutableListNode) {

};
```

C# Solution:

```
/*
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
```

```
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public void PrintValue(); // print the value of this node.
* public ImmutableListNode GetNext(); // return the next node.
* }
*/

public class Solution {
public void PrintLinkedListInReverse(ImmutableListNode head) {

}
}
```

C Solution:

```
/*
* Problem: Print Immutable Linked List in Reverse
* Difficulty: Medium
* Tags: array, linked_list, stack
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition for ImmutableListNode.
* struct ImmutableListNode {
* struct ImmutableListNode* (*getNext)(struct ImmutableListNode*); // return
the next node.
* void (*printValue)(struct ImmutableListNode*); // print the value of the
node.
* };
*/
```

```
void printLinkedListInReverse(struct ImmutableListNode* head) {

}
```

Go Solution:

```
// Problem: Print Immutable Linked List in Reverse
// Difficulty: Medium
// Tags: array, linked_list, stack
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

/* Below is the interface for ImmutableListNode, which is already defined for
you.
*
* type ImmutableListNode struct {
*
* }
*
* func (this *ImmutableListNode) getNext() ImmutableListNode {
* // return the next node.
* }
*
* func (this *ImmutableListNode) printValue() {
* // print the value of this node.
* }
*/

func printLinkedListInReverse(head ImmutableListNode) {

}
```

Kotlin Solution:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
```

```
* fun getNext(): ImmutableListNode? {} // return the next node.
* fun printValue() {} // print the value of this node.
* };
*/

class Solution {
fun printLinkedListInReverse(head:ImmutableListNode?) {

}
}
```

Swift Solution:

```
/**
* Definition for ImmutableListNode.
* public class ImmutableListNode {
* public func printValue() {}
* public func getNext() -> ImmutableListNode? {}
* }
*/

class Solution {
func printLinkedListInReverse(_ head: ImmutableListNode?) {

}
}
```

Ruby Solution:

```
# This is the ImmutableListNode's API interface.
# You should not implement it, or speculate about its implementation.
#
# class ImmutableListNode
# def printValue()
# . print the value of this node.
# def end
# """
#
# def getNext()
# . return the next node.
# end
```

```
# end

def printLinkedListInReverse(head)

end
```

PHP Solution:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode {
* public function printValue() {} // print the value of this node.
* public function getNext() {} // return the next node.
* };
*/

class Solution {
/**
* @param ImmutableListNode $head
* @return void
*/
function printLinkedListInReverse($head) {

}
}
```

Scala Solution:

```
/**
* // This is the ImmutableListNode's API interface.
* // You should not implement it, or speculate about its implementation.
* class ImmutableListNode{
* def printValue(): Unit = {} // print the value of this node.
* def getNext(): ImmutableListNode = {} // return the next node.
* };
*/

object Solution {
def printLinkedListInReverse(head: ImmutableListNode): Unit = {
```

```
}
}
```