极客学院
jikexueyuan.com

名企数据结构面试题之链表（上）

名企数据结构面试题之链表（上） — **课程概要**

- 链表原理
- 逆序打印链表
- 链表的最大元素
- 链表反转

链表原理

链表原理

- 链表的内存模型
- 创建和打印
- 插入和删除
- 查询和修改
- 测试链表
- 其它知识点

链表原理 — **链表的内存模型**

单链表有两个属性：

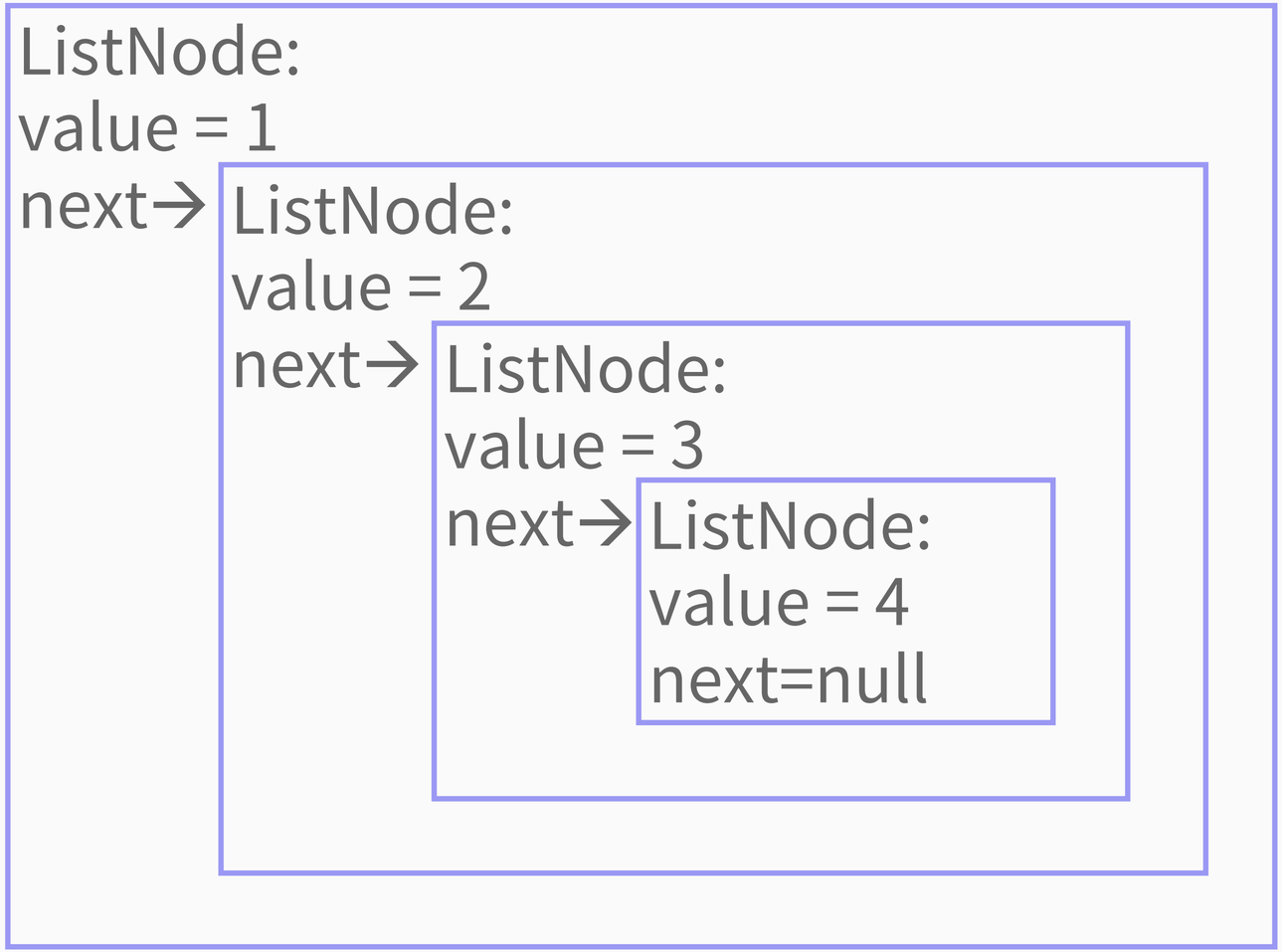
- value，值
- next，指向下一个节点的指针（引用）

双链表**附加**一个属性：

- pre，指向上一个节点的指针（引用）

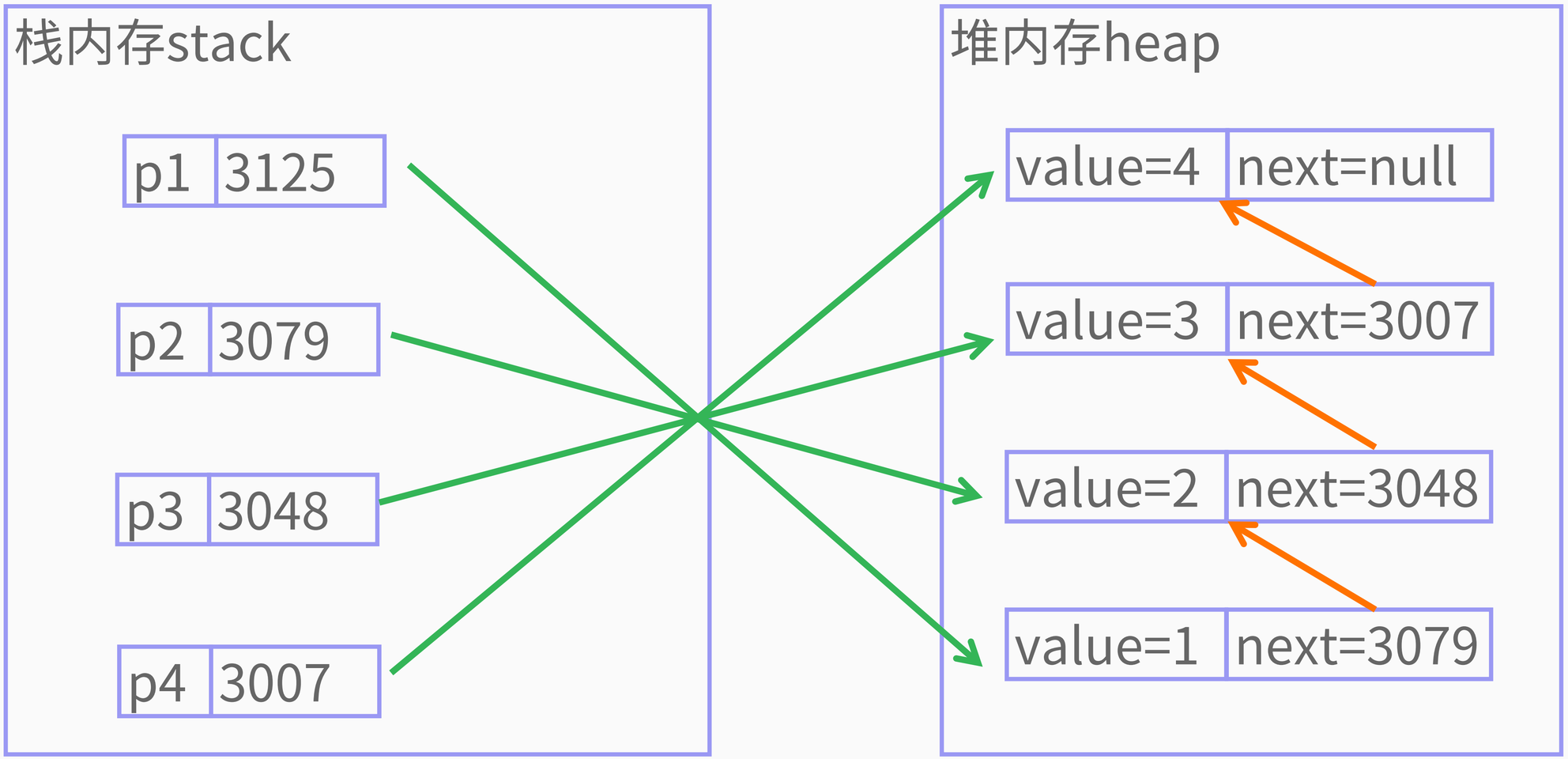
链表原理 — **链表的内存模型**

似乎是这样：



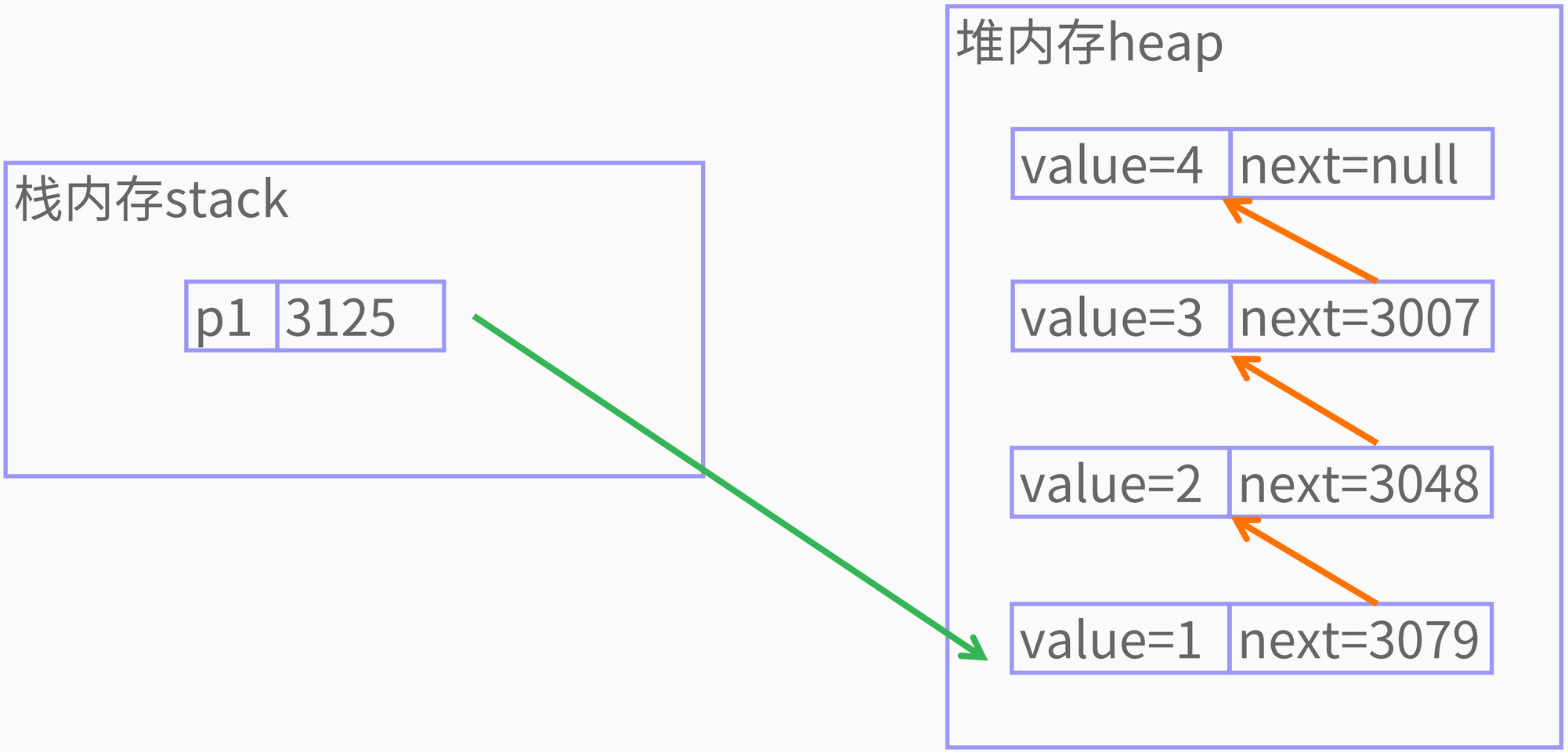
链表原理 — 链表的内存模型

其实是这样：



链表原理 — 链表的内存模型

只有一个引用的情形：



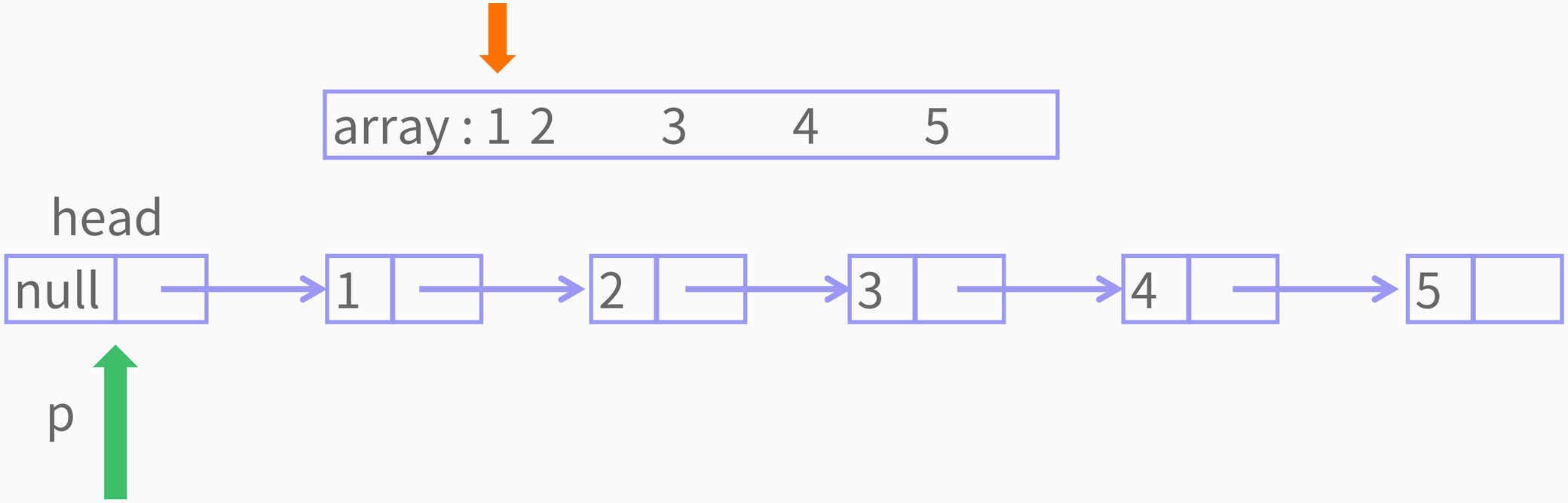
链表原理 — 创建和打印

单链表API：MiniList.java

方法名/属性名	作用
head	头结点，固定
void arrayToList(T[] array)	根据数组array创建链表
void printList()	打印链表
void insert(int index, T value)	在第index个节点后面插入value
T remove(int index)	删除第index个节点，并返回节点的值
T get(int index)	返回第index个节点的值
void set(int index,T value)	将第index个节点的值设置为value

链表原理 — 创建和打印

根据数组array创建链表：

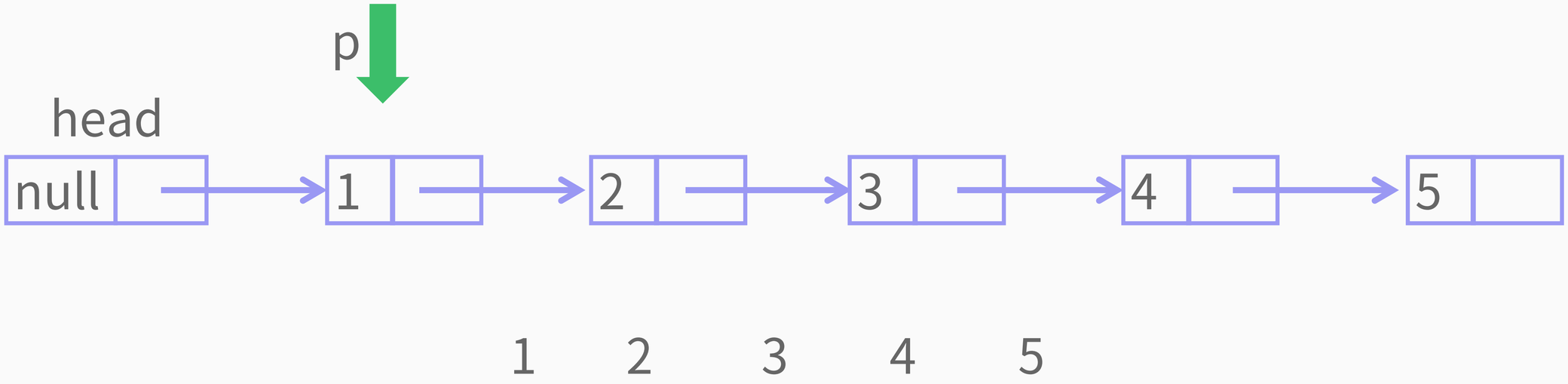


链表原理 — 创建和打印

依次打印链表，其实就是遍历

遍历的方法：

p=p.next

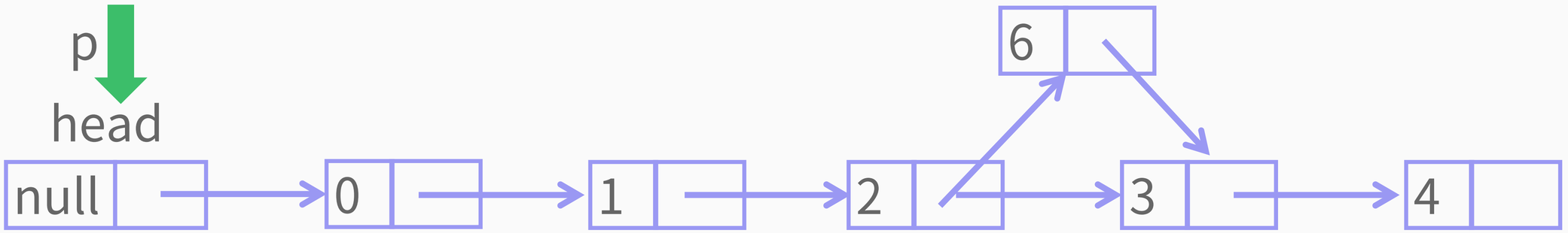


1　　2　　3　　4　　5

链表原理 — 插入和删除

void insert(int index, T value)，在第index个节点后面插入value，index从0开始。

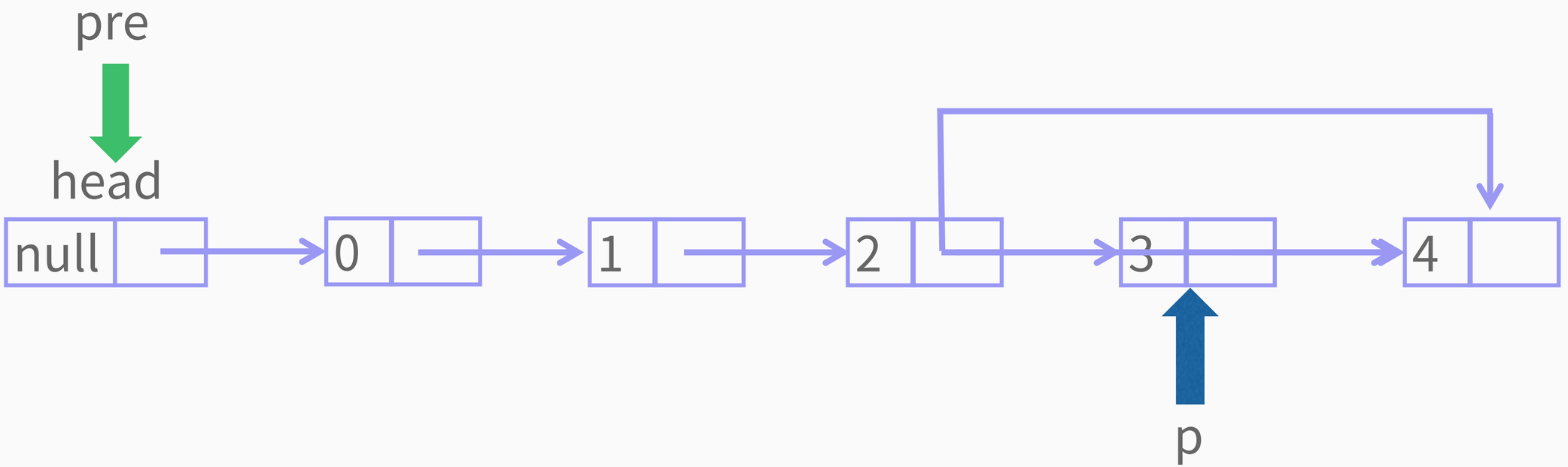
- 遍历链表，找到第index个节点p
- 新建节点node
- node.next=p.next
- p.next=node



链表原理 — 插入和删除

T remove(int index)，删除第index个节点，并返回节点的值

- 遍历链表，找到前趋节点pre，以及节点p
- pre.next=p.next
- return p.value



链表原理 — **查询和修改**

T get(int index)，返回第index个节点的值，index从0开始。

void set(int index,T value)，将第index个节点的值设置为value，index从0开始。

遍历！

链表原理 — 测试链表

Key	Value
类名	TestList
方法名	testMiniList

操作	输出
根据数组{0,1,2,3,4}创建链表；输出链表	0 1 2 3 4
在第2个节点后面插入10；输出链表	0 1 2 10 3 4
删除第4个节点；输出链表	0 1 2 10 4
将第3个节点的值设置为13；输出链表	0 1 2 13 4
查询第4个节点	4

链表原理 — **其它知识点**

参考LinkedList源码

- 容器大小：size()
- 是否为空：isEmpty()
- for循环的封装
- 边界检查
- 迭代器

ArrayList、LinkedList、Stack、Queue等线性结构的源码，以后再探讨。

逆序打印链表

逆序打印链表

- 问题描述
- 非递归算法的思路
- 非递归算法的实现
- 递归算法的思路
- 递归算法的实现

逆序打印链表 — 问题描述

给定单链表，从尾到头打印每个节点的值，不同的值之间用空格隔开。

比如：1→2→3→4→5

输出：5 4 3 2 1

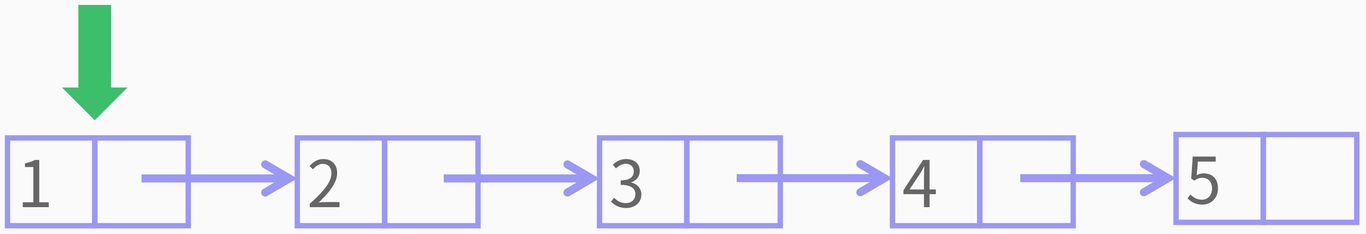
用非递归以及递归两种算法实现。

逆序打印链表 — **非递归算法的思路**

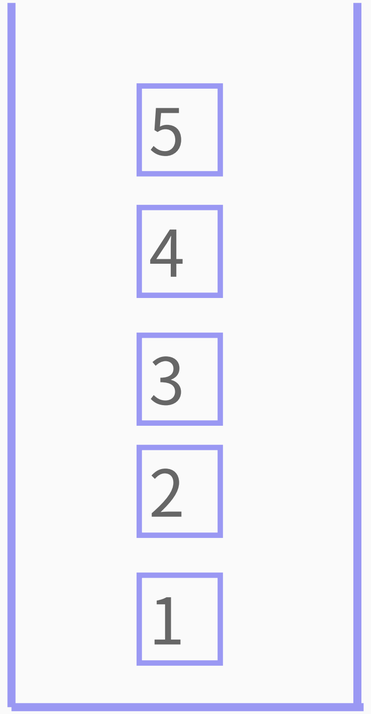
先打印**尾**部、**后**打印**头**部。

自然而然联想到：先进后出的**栈**。

- 遍历链表，将所有的节点（值）依次压栈
- 依次弹栈、打印，直到栈为空



输出：　5　4　3　2　1

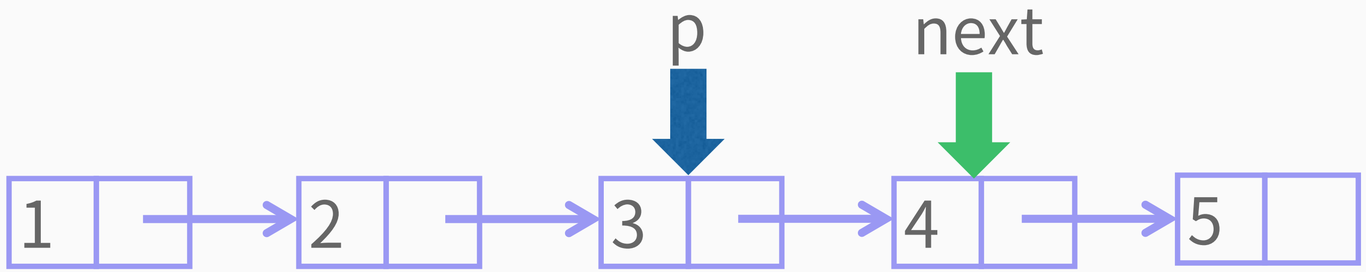


逆序打印链表 — **非递归算法的实现**

Key	Value
类名	MiniList
方法名	printInverse
时间复杂度	O(N)
空间复杂度	O(N)
测试输入	1→2→3→4→5
测试输出	5 4 3 2 1

逆序打印链表 — 递归算法的思路

```
void recursive(ListNode p){//对于某个节点p
    if(p!=null){
        recursive(p.next);//递归p的下一个节点
        print(p);//打印本节点p
    }
}
```



输出：　5　4　3

逆序打印链表 — 递归算法的实现

Key	Value
类名	MiniList
方法名	printInverseRecursive
时间复杂度	O(N)，N表示链表长度
空间复杂度	O(N)
测试输入	1→2→3→4→5
测试输出	5 4 3 2 1

链表的最大元素

链表的最大元素

- 如何比较大小
- 打擂台算法
- 代码实现
- 测试用例
- 其它知识点

链表的最大元素 — **如何比较大小**

对于T类型的a和b，如何比较大小?

- Comparable接口
- 自定义Comparator

Key	Value
类名	MiniList
方法名	compare
作用	比较a和b的大小

链表的最大元素 — **打擂台算法**

争夺**武林盟主**

令狐冲，武力95

韦小宝，武力15

张三丰，武力98

木婉清，武力60

链表的最大元素 — **打擂台算法**

如何获取链表的最大元素?

```
T getMax(){
    p=head.next;
    max=p.value;//临时变量max，记录最大值
    p=p.next;
    while(p!=null){
    //遍历链表，依次比较max与当前值的大小
        if(compare(p.value, max)>0){//或者≥
            max=p.value;
        }
        p=p.next;
    }
    return max;
}
```

链表的最大元素 — 代码实现

Key	Value
类名	MiniList
方法名	getMax
时间复杂度	O(N)，N表示链表长度
空间复杂度	O(1)

链表的最大元素 — **测试用例**

Key	Value
类名	TestList
方法名	testMaxInteger
测试输入	3→4→1→2
测试输出	4

链表的最大元素 — 其它知识点

Comparable、Comparator广泛应用于其它数据结构，以及排序、查找等算法。

类名	方法名	作用
TreeMap	put	二叉排序树的插入
PriorityQueue	siftUp	优先队列的上滤
Collections	sort	排序
Collections	binarySearch	二分查找

以后再探讨！

链表反转

链表反转

- 问题描述
- 非递归算法的思路
- 非递归算法的实现
- 递归算法的思路
- 递归算法的实现
- 思考题

链表反转 — 问题描述

给定单链表，反转这个单链表，并返回新的头节点。

例如，给定单链表：1→2→3→4→5→6

经过反转之后，变成：6→5→4→3→2→1

leetCode 206：Reverse Linked List

用递归、非递归两种算法实现。

假设链表的长度为n，要求两种算法的时间复杂度必须为O(N)；

非递归算法的空间复杂度必须为O(1)。

链表反转 — **非递归算法的思路**

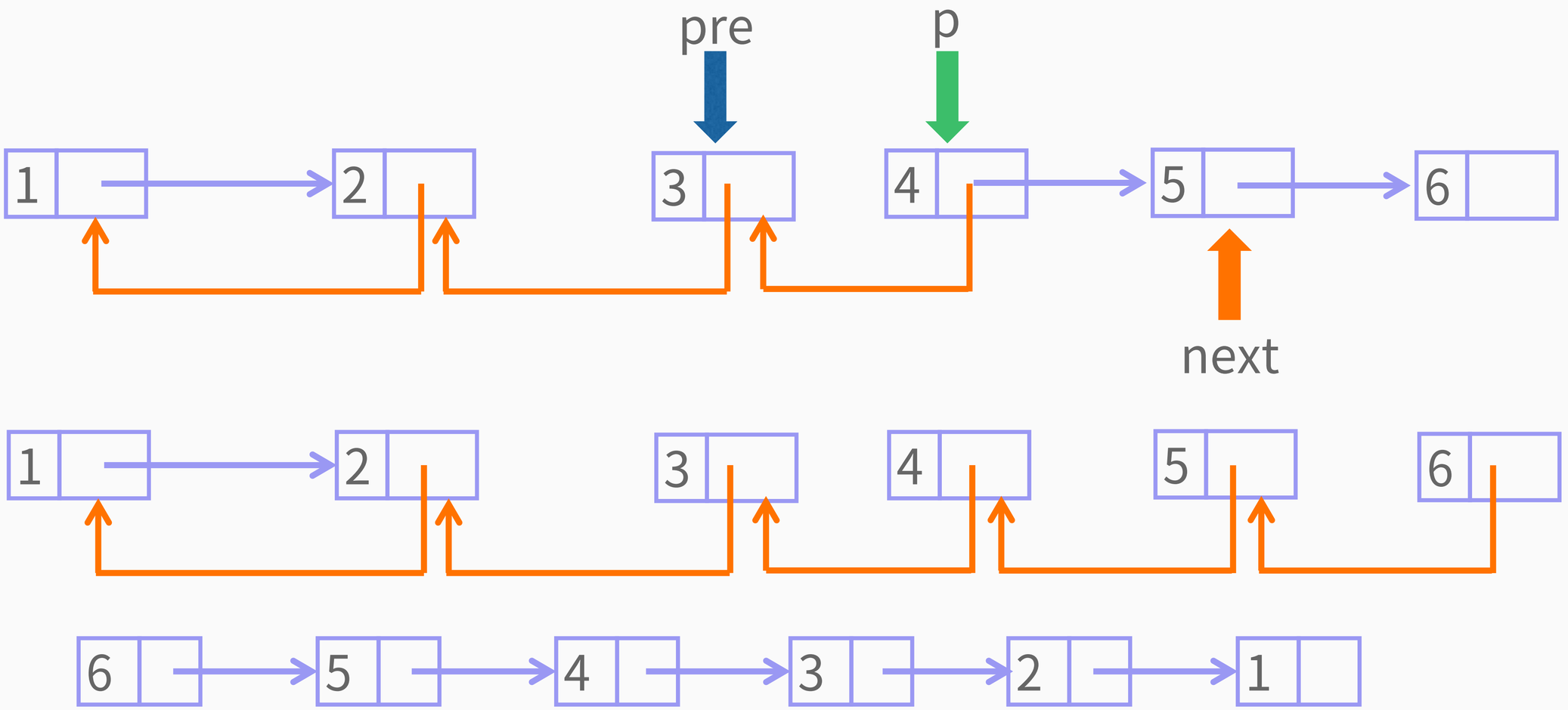
三个指针：

- pre，前趋节点
- p，当前节点
- next，下一个节点

关键的四个步骤：

```
next=p.next;
p.next=pre;
pre=p;
p=next;
```

链表反转 — 非递归算法的思路



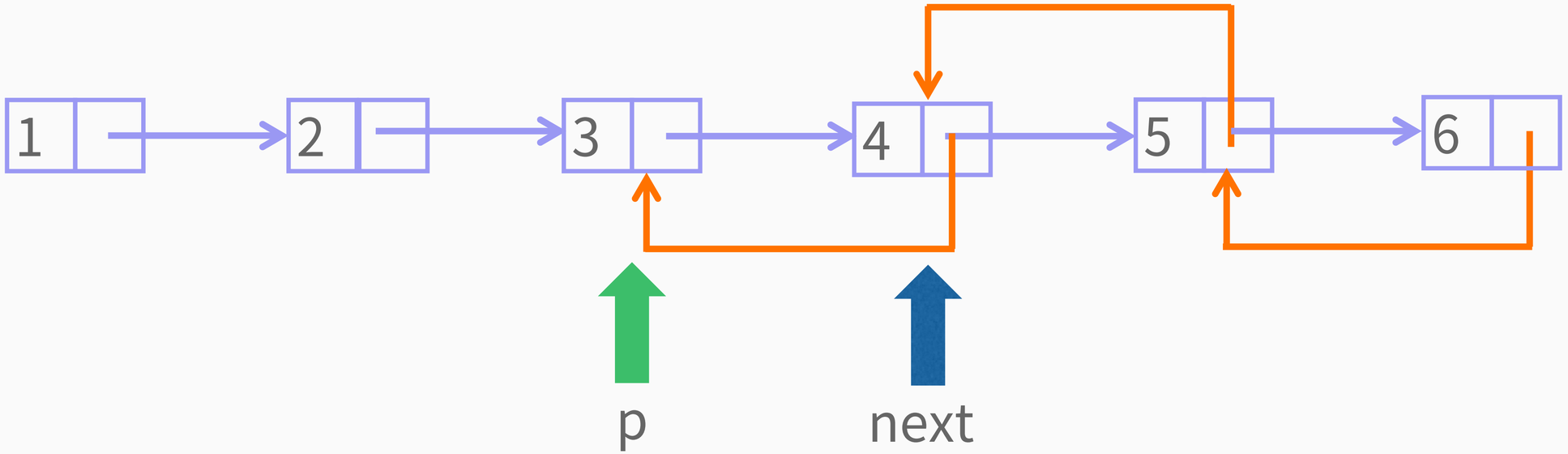
链表反转 — 非递归算法的实现

Key	Value
类名	_206ReverseLinkedList
方法名	reverseList
时间复杂度	O(N)，N表示链表长度
空间复杂度	O(1)
测试输入	1→2→3→4→5→6
测试输出	6→5→4→3→2→1

链表反转 — 递归算法的思路

对于某个节点p：

- 递归处理next
- next.next=p
- 返回尾部节点



链表反转 — 递归算法的实现

Key	Value
类名	_206ReverseLinkedList
方法名	reverseListRecursive
时间复杂度	O(N)，N表示链表长度
空间复杂度	O(N)
测试输入	1→2→3→4→5→6
测试输出	6→5→4→3→2→1

链表反转 — **思考题**

leetCode 25：Reverse Nodes in k-Group

依次反转链表的k个节点。

1→2→3→4→5→6→7，k=2

反转之后：2→1→4→3→6→5→7

链表反转 — **思考题**

leetCode 92：Reverse Linked List II

反转链表的某一部分。

1→2→3→4→5，m=2，n=4

反转之后：1→4→3→2→5

小结：

- 指针（引用），pre、p、next
- p=p.next
- xxx.next=yyy.next

名企数据结构面试题之链表（上）

本套课程中我们学习了名企数据结构面试之链表（上）。你应当掌握了以下知识：

- 链表原理
- 逆序打印链表
- 链表的最大元素
- 链表反转

你可以使用leetCode验证程序是否正确，还可以在白纸上书写代码；如果想进一步提高，你可以继续在极客学院学习**名企数据结构面试题之链表（中）**课程。

极客学院

jikexueyuan.com

中国最大的IT职业在线教育平台