Python入门与基础
第四讲 线性数据结构 Part II
Queue & Stack

林平之老师

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: ninechapter
知乎专栏: http://zhuanlan.zhihu.com/jiuzhang
微博: http://www.weibo.com/ninechapter
官网: www.jiuzhang.com

版权声明

九章课程不提供视频，也严禁录制视频的侵权行为

否则将追求法律责任和经济赔偿

请不要缺课

本节重点

栈原理及实现

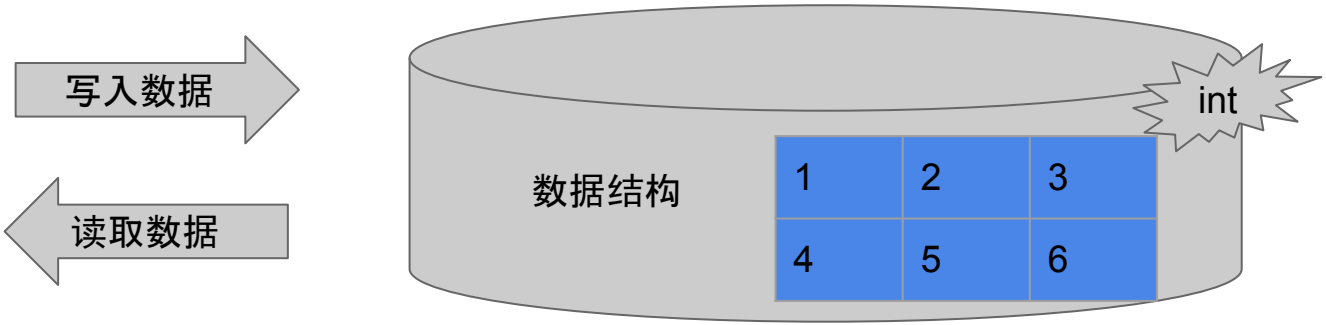
队列原理实现及使用Python的Queue模块

算法的时间、空间复杂度

数据结构Recap

数据结构 = 数据 + 存储方式 + 操作

- 数据 - 存储什么数据？如int，string类型
- 存储方式 - 如何组织数据，数据之间的关系？
- 操作 - 如何快速的读取查询数据，写入数据到数据结构中？



数据结构Recap

不要求数据的顺序，维护成本低

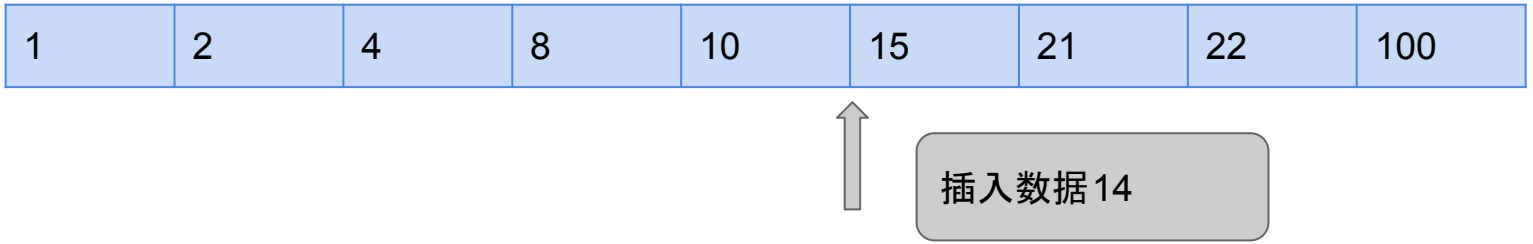
比如：List 列表

1	2	5	-1	3	10	2	2	2

数据结构Recap

要求数据的顺序，维护成本高，使用成本低

比如：排序的List



栈 Stack

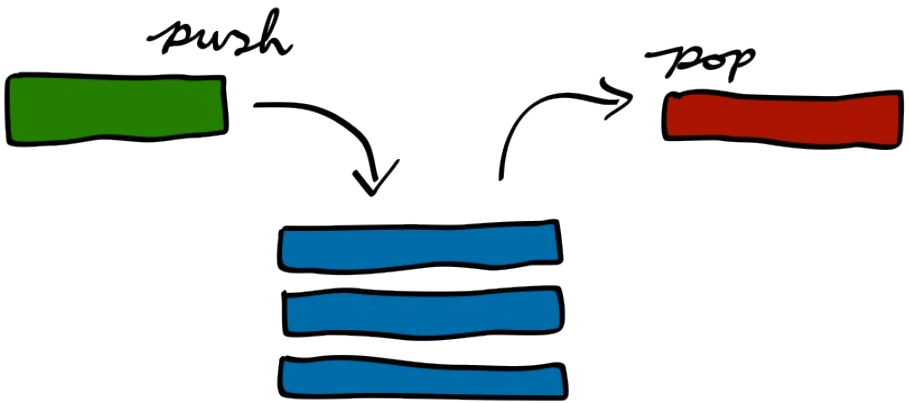
什么是栈(stack)

栈是一种后进先出(last in first out, LIFO)的线性数据结构

栈 Stack

栈的操作

- push 放入一个元素
- pop 弹出一个元素
- top 获取栈顶元素
- empty 判断栈是否为空



栈的操作示例

Stack Operation	Stack Contents	Return Value
s.isEmpty()	[]	True
s.push(100)	[100]	
s.push('linpz')	[100,'linpz']	
s.peek()	[100,'linpz']	'linpz'
s.push(True)	[100,'linpz',True]	
s.size()	[100,'linpz',True]	3
s.isEmpty()	[100,'linpz',True]	False
s.push(8.100)	[100,'linpz',True,8.1]	
s.pop()	[100,'linpz',True]	8.1
s.pop()	[100,'linpz']	True
s.size()	[100,'linpz']	2

栈 Stack

栈的实现

- 使用基于List实现Stack

Implement Stack

https://www.lintcode.com/en/problem/implement-stack/

https://www.jiuzhang.com/solutions/implement-stack/

Valid Parentheses

https://www.lintcode.com/en/problem/valid-parentheses/

https://www.jiuzhang.com/solutions/valid-parentheses/

Queue 队列

队列

什么是队列(queue)

- 食堂里排队打饭
- 过安检的时候排队

队列是一种**先进先出**(first in first out, FIFO)的**线性**数据结构

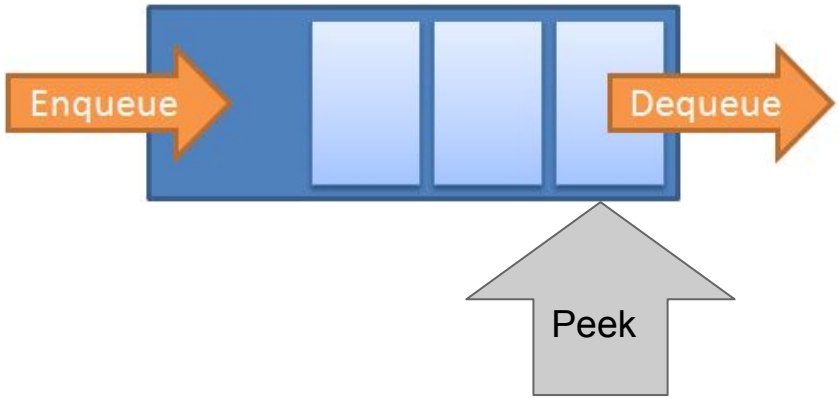
Queue提供的接口方法

1. Queue() 定义一个空队列，无参数，返回值是空队列
2. enqueue(item) 在队列尾部加入一个数据项，参数是数据项，无返回值
3. dequeue() 删除队列头部的数据项，不需要参数，返回值是被删除的数据，队列本身有变化
4. isEmpty() 检测队列是否为空。无参数，返回布尔值
5. size() 返回队列数据项的数量。无参数，返回一个整数

队列提供的接口方法

Queue模块中队列的提供的结构

- put (enqueue) 元素进队列
- get (dequeue) 元素出对列
- empty 判断队列是否
- 为空

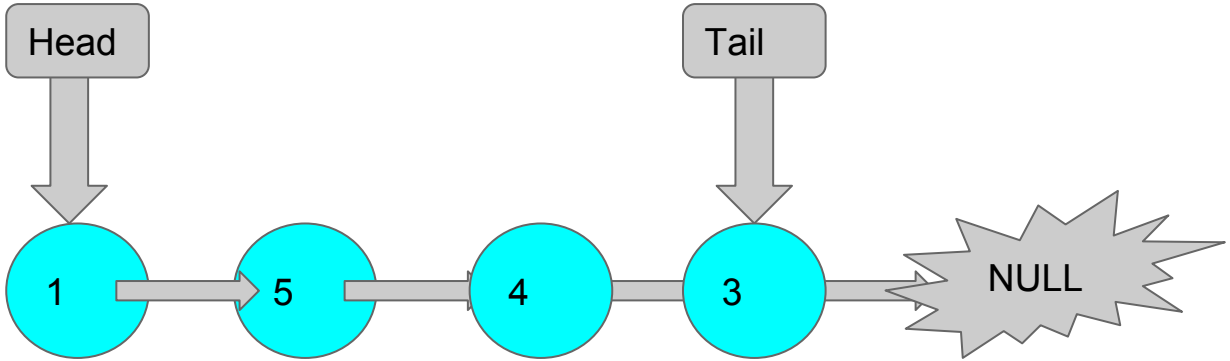


队列 Queue

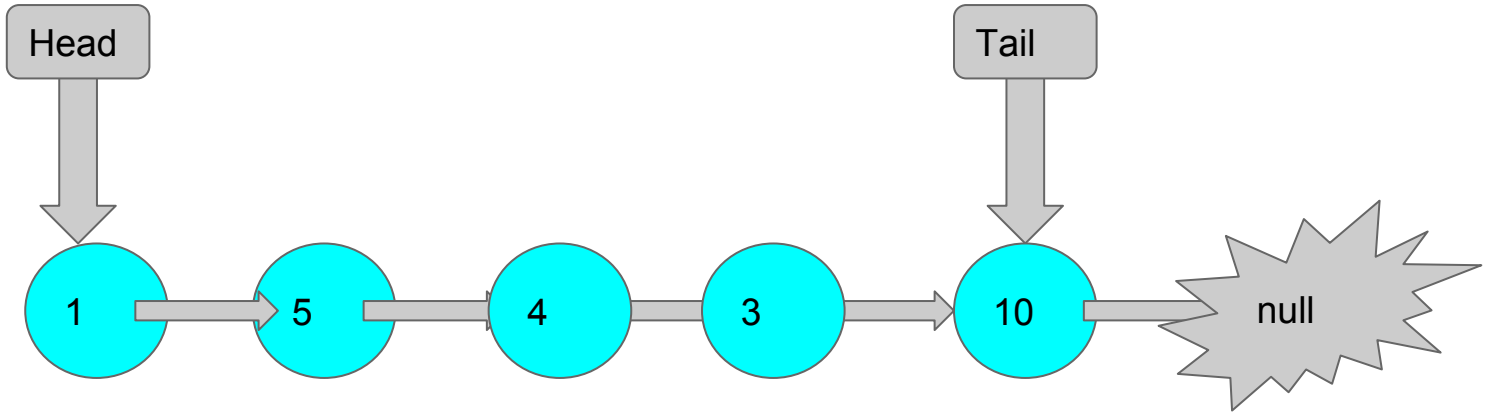
队列的实现

- 使用ListNode实现Queue
- 使用Queue模块

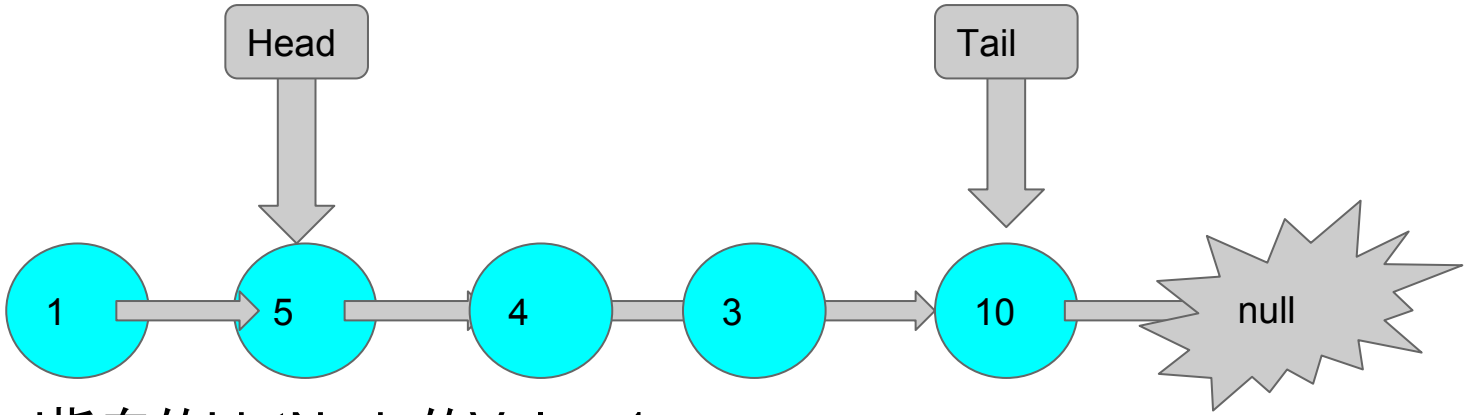
基于ListNode的queue



put插入一个Value-10

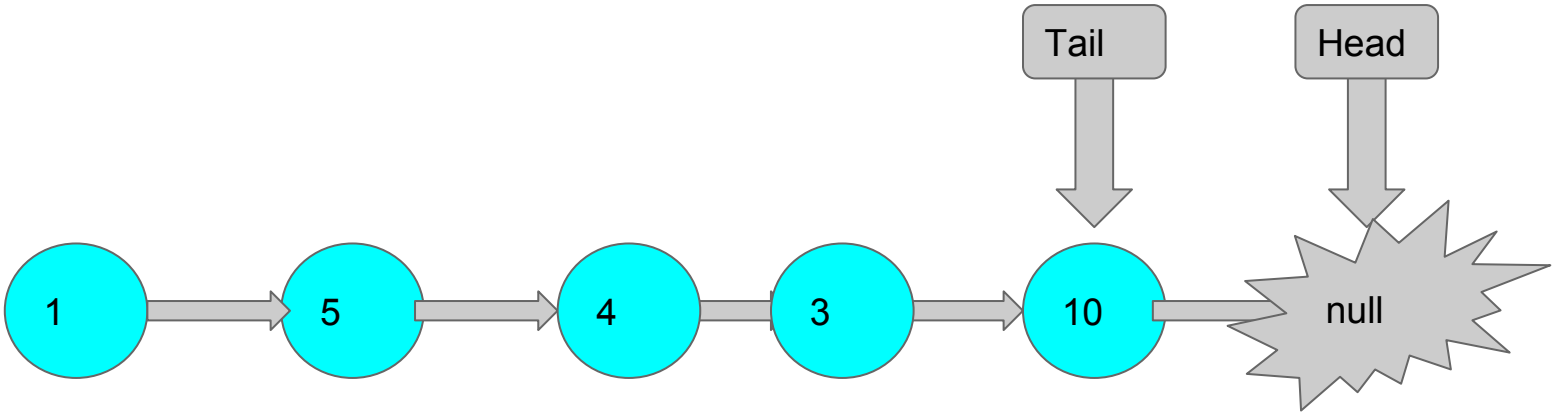


get 取出队列头部元素的值



- 得到Head指向的ListNode的Value 1
- Head往后移动来到下一个ListNode
- 返回1这个值

empty 判断对列为空



面试真题Implement Queue by Linked List

Implement Queue by Linked List

http://www.lintcode.com/en/problem/implement-queue-by-linked-list/

http://www.jiuzhang.com/solution/implement-queue-by-linked-list/

Python提供的Queue模块

Python中队列是线程间最常用的交换数据的方式(在爬虫中我们会使用它)

```
3  import Queue
4  queue = Queue.Queue(maxsize = 10)
5
```

定队列长度

- 如果maxsize小于1就表示队列长度无限

PUT 元素放入队列


```
7
8   queue.put(10)
9
```

put()方法在队尾插入一个元素，如果队列是满的，该线程会阻塞（等待），直到空出一个位置来，然后在队尾放入元素

GET 获取队列的元素


```
12   queue.get()
```

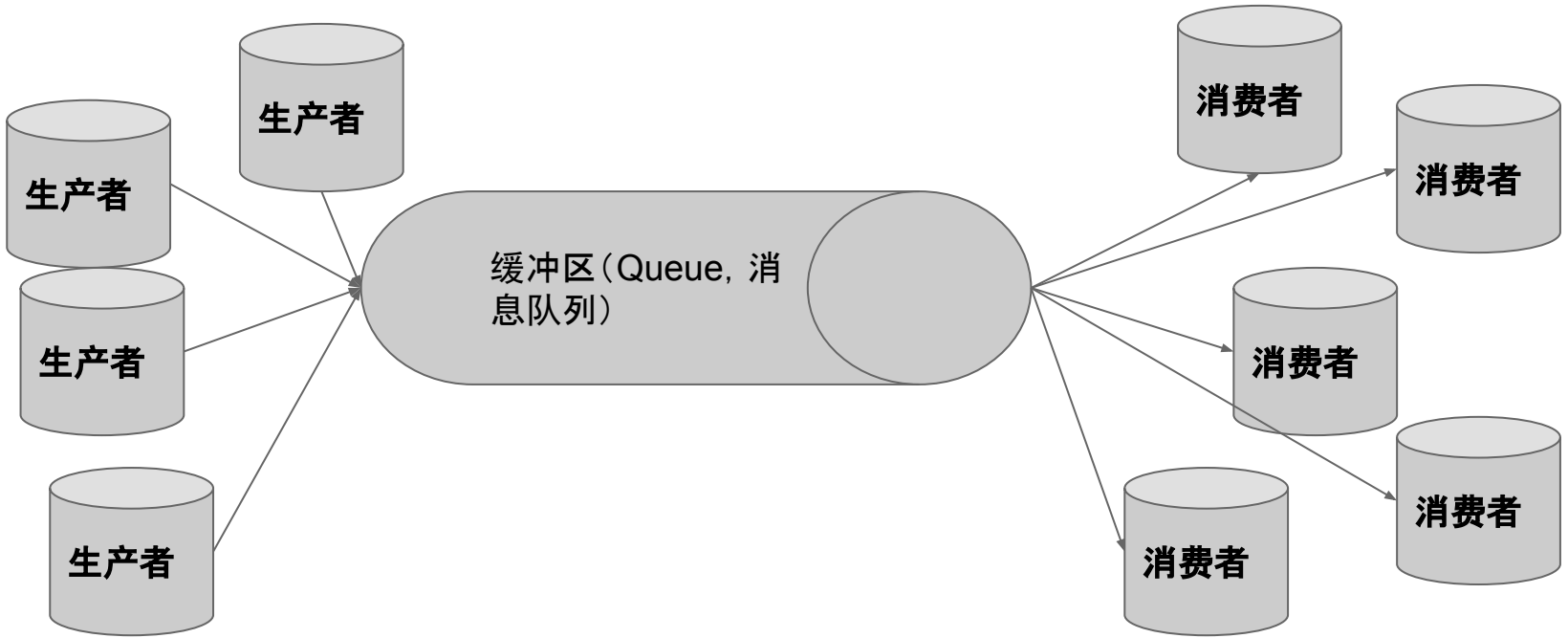
get()方法从队头删除并返回一个项目，队列为空默认为阻塞线程（等待），直到有元素放入到队列之后，将其取走。

Python使用Queue实现生产者消费者模型

什么是生产者消费者模式：

- 当某模块或者组件负责生产数据，然后这些数据由其他模块或者组件来负责处理（此处的模块可能是：函数、线程、进程等）
- 产生数据的模块或者组件称为生产者，而处理数据的模块称为消费者。
- 在生产者与消费者之间有缓冲区，生产者负责往缓冲区放数据，消费者负责从缓冲区读取数据，这就形成了生产者消费者模式。

生产者消费者模式



生产者消费者模式的优点

- 解耦

  如果消费者直接调用生产者的代码，那么代码会相互产生依赖(耦合)，通过中间的缓冲区可以解耦两者

- 并发

  由于生产者和消费者是两个不同的模块，对于生产者而言它负责生产数据，只要往缓存区中丢数据完成，就可以继续生产下一个数据，不会因为数据是否被消费而阻塞

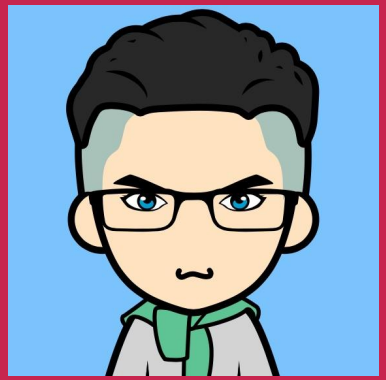
队列 Queue

队列的应用

Message queue 消息队列

- https://github.com/apache/kafka
- https://github.com/rabbitmq/rabbitmq-server

BFS 广度优先搜索



谢谢大家