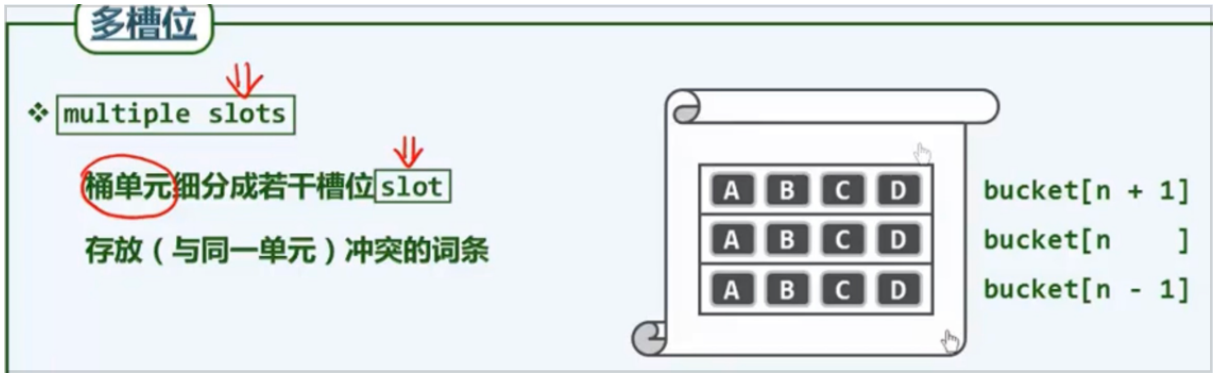
11C 排解冲突(1)

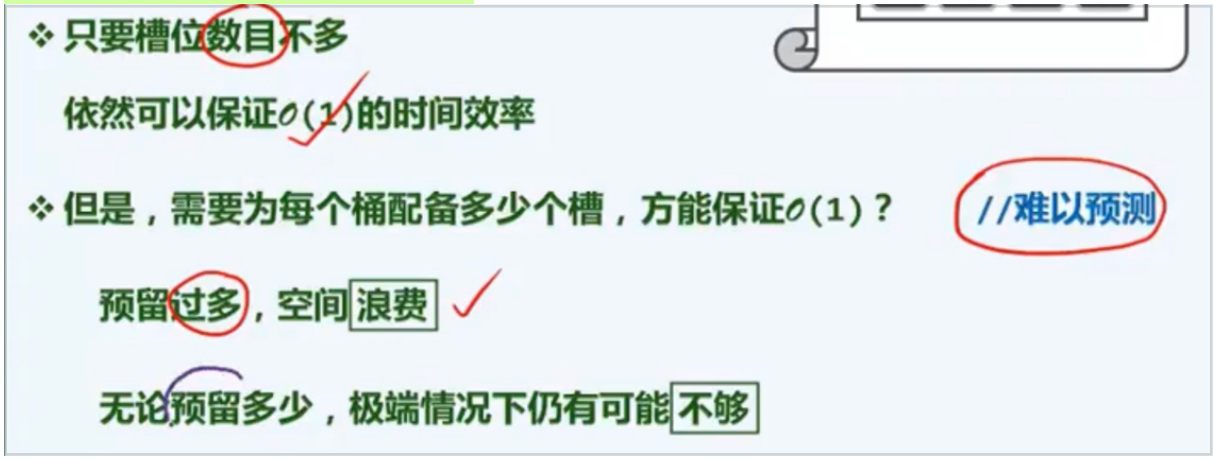
#数据结构邓神

一山二虎 排解冲突 | 预案

multiple slots



但是会不会一个槽位溢出了怎么办?



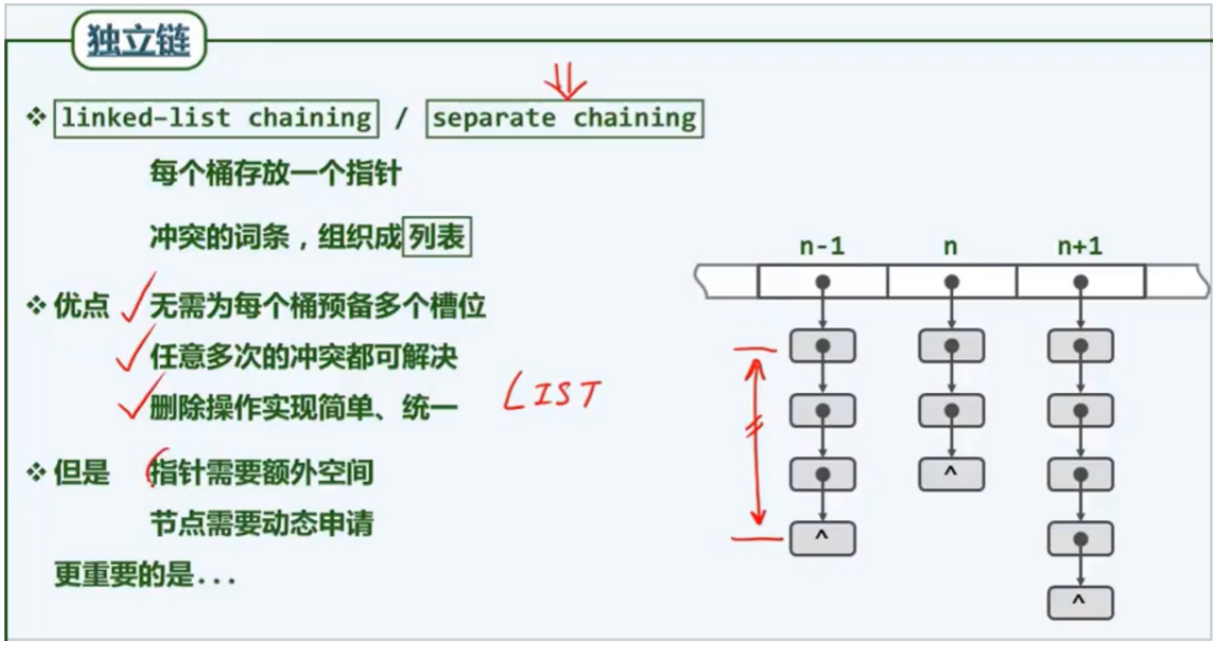
泾渭分明 | 独立链条

改用链表

```
vector<list<elemType>> hash
```

但是这些都是没有办法优化散列函数的解决办法

最好的办法是竭尽全力优化散列函数，经可能避免重复，提高均匀度，避免重复



动态申请内存空间的时间效率要高出两个数量级

最大缺陷： 空间未必连续分布，系统缓存失去效果

(系统无法预测你的访问方向，无法通过缓存加速你的访问)

我们如何充分利用缓存呢

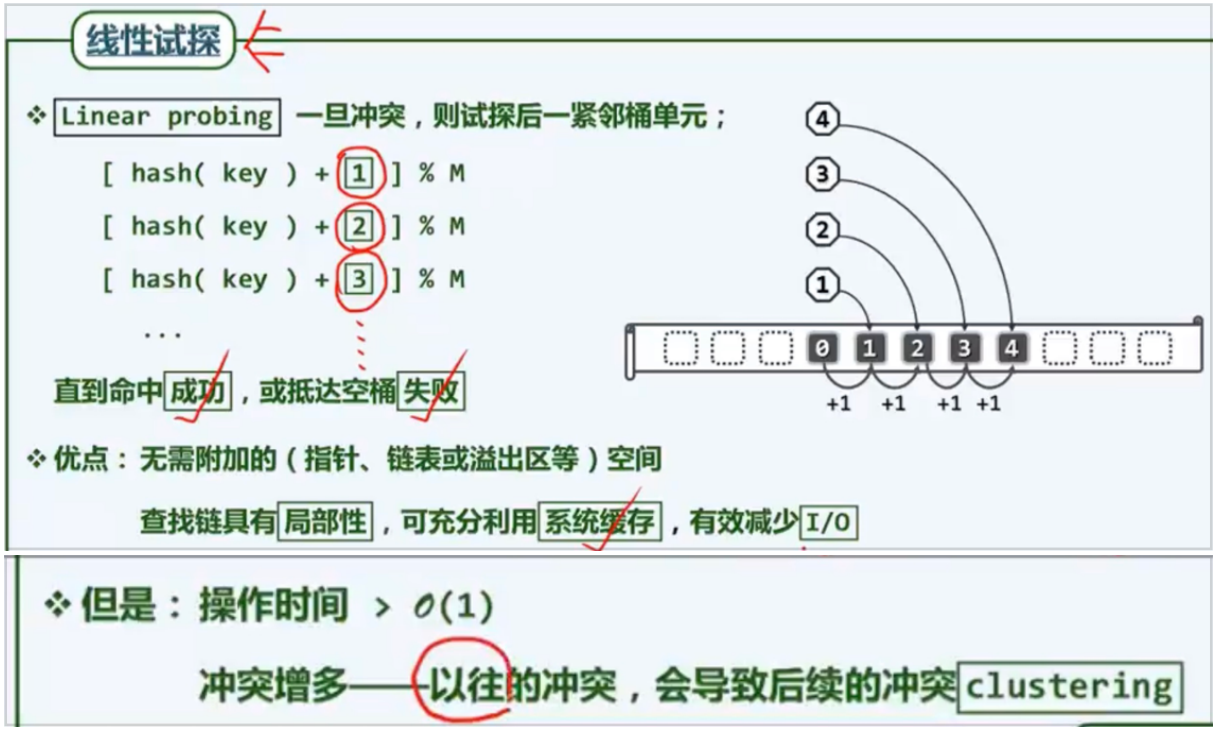
开放定址 open addressing

独立链表法： closed addressing

不用申请额外的空间，尝试使用系统缓存加速

线性试探

这种方法看起来慢，但是线性访问可以充分利用系统缓存加速！



有可能存在一大片连续而都被使用的单元，导致插入单元被阻挡半天

反过来查找也可能导致这种情况，而且这种情况会愈演愈烈，情况会不断劣化。

懒惰删除

因为一旦查找到空桶就会立刻停止，所以其中不能有任何空隙

那么如何删除词条