ordered map

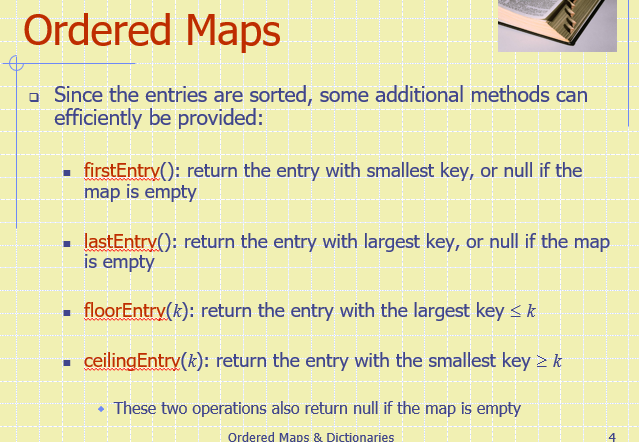
在某些应用中，仅仅通过entry的key就想找到他的值是不够的

entry也许下需要以total order来按顺序插入map

一个ordered map需要map本身具备的通常method，但同样有与key有关的 order

假设我们把key以一个总顺序被插入，comparator用来提供keys之间的order关系

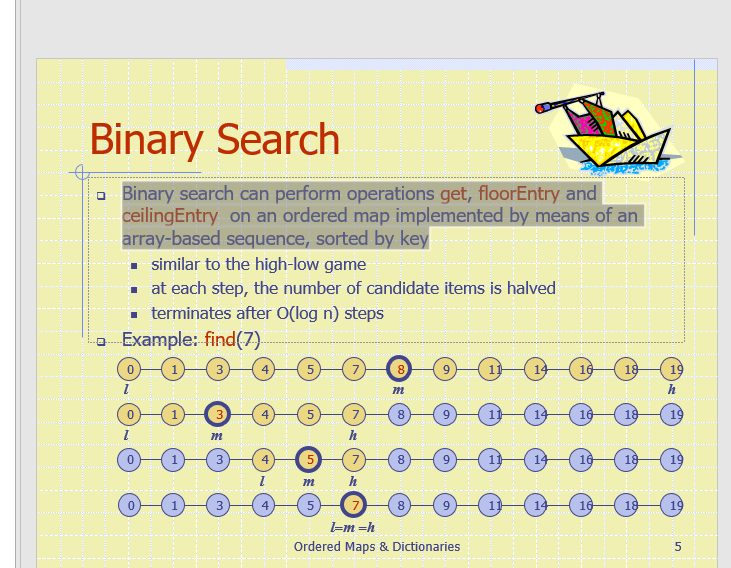
因为entry现在是sorted了，所以我们提供一些新method



binary search，

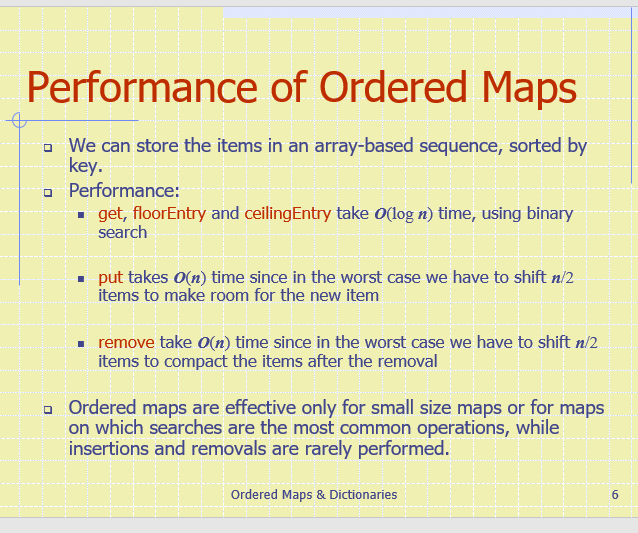
binary search 可以在有序map上进行get, floorentry,cellingEntry这几个操作，这个有序map是由基于array的sequence实现的，这个array将key按顺序排列

每次都对半分，Ologn steps后就会终结



ordered map的效率

一个array-based,用key sort的ordered map



get,floor celling都是logn

put需要On假设我们一开始就找到了但是我们需要n.2往后移，实际上这也不是最差的，在前半部分更差，但都是n

removeOn

适合用ordered map的map

1:size很小，put remove就算On也没啥影响

2.map的绝大部分操作都是get,floorentry与cellingentry，这样就不用Put和remove

Dictionary ADT,

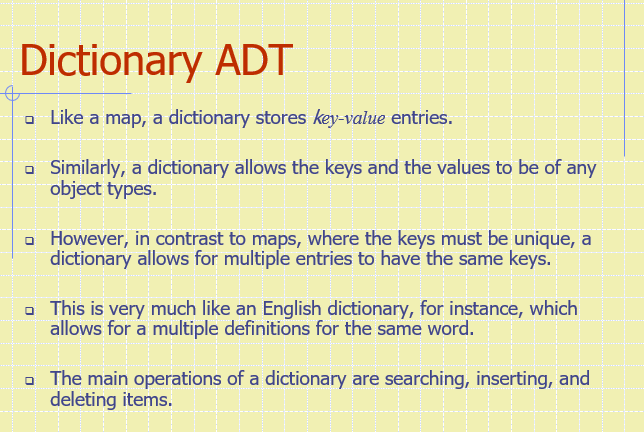
像map一样，一个dictionary存储着key-value entries

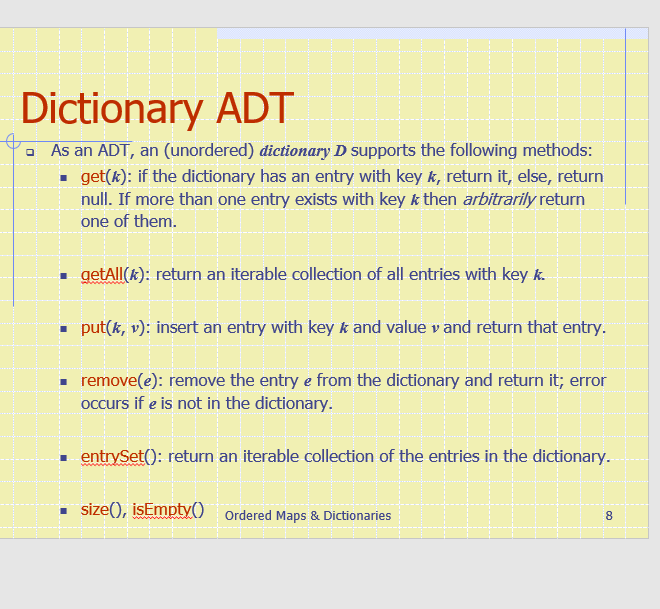
同样的，一个dictionary允许key和·value是任何object types

然而，与map相比，map需要key必须是不同的，而dictionary允许不同的entry有着同一个key

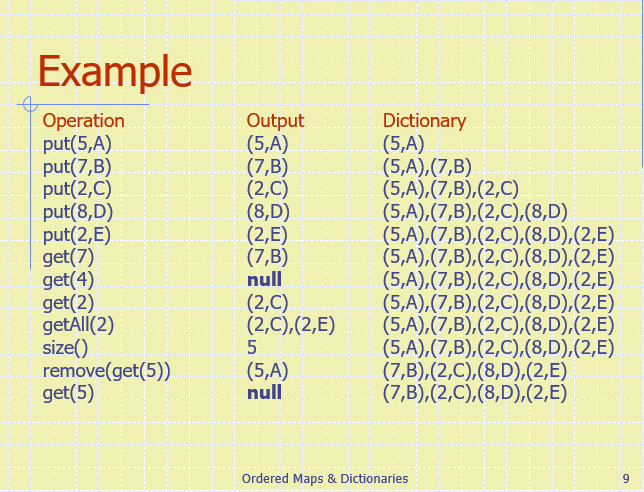
就有点像字典，一个词语可以有多个描述

dictionary的主要操作时searching ,inserting, deleting





如果有多个的时候，get可以任意返回其中一个



Put就不说，Put2E的时候，还是可以插入并且不会删除2C

get7只有一个，所以只return一个

get4没有，所以return null

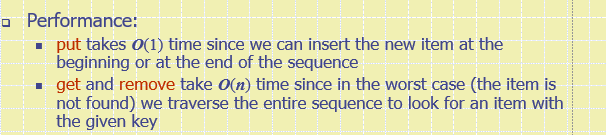
get2，随机return 了一个2C

getall，都return

List-based dictionary

log file\audit trail, 一个由unsorted sequence建立的dictionary

我们把dictionary的item存储在sequence里（doublelinked list 或者array）



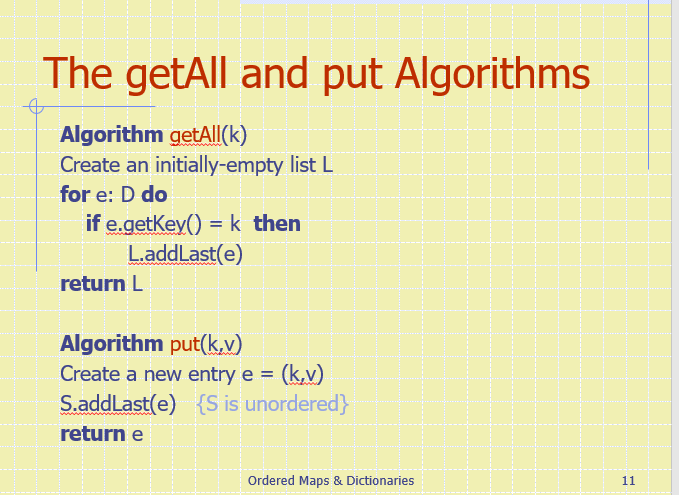
put就插在开头或者结尾，O1

因为是unsorted，所以是On，（没找到）

log file可以使用的情况

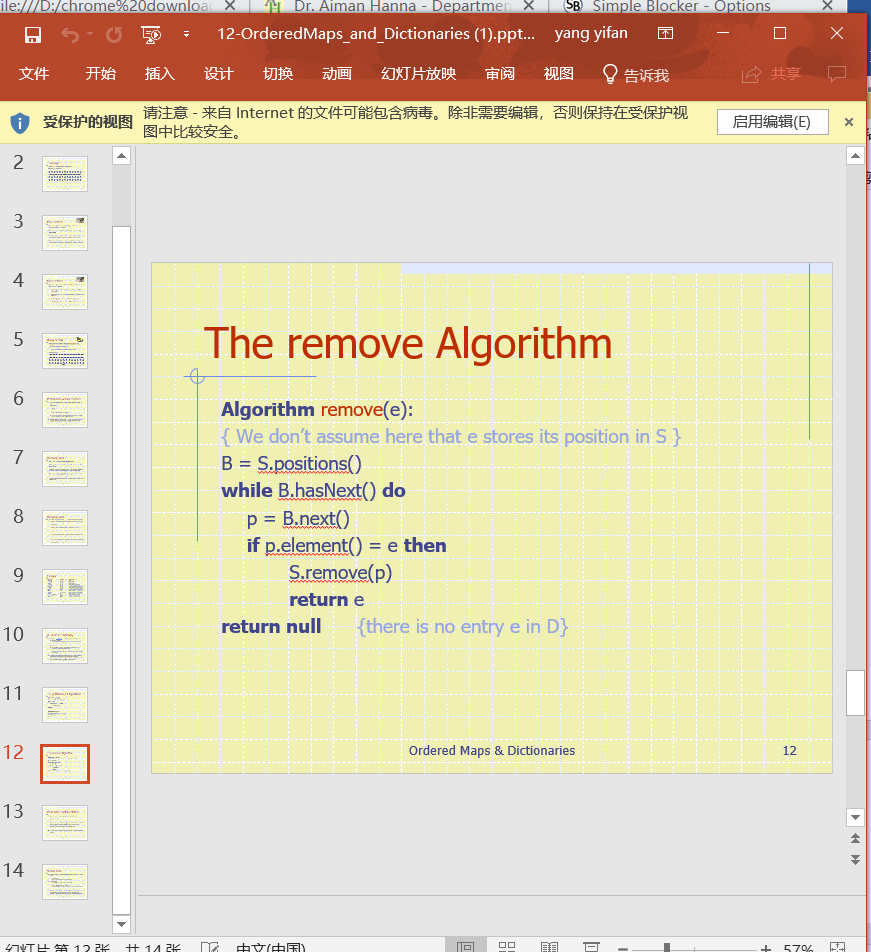
1.size小

最长够用的操作时insertion，而不是get或者remove

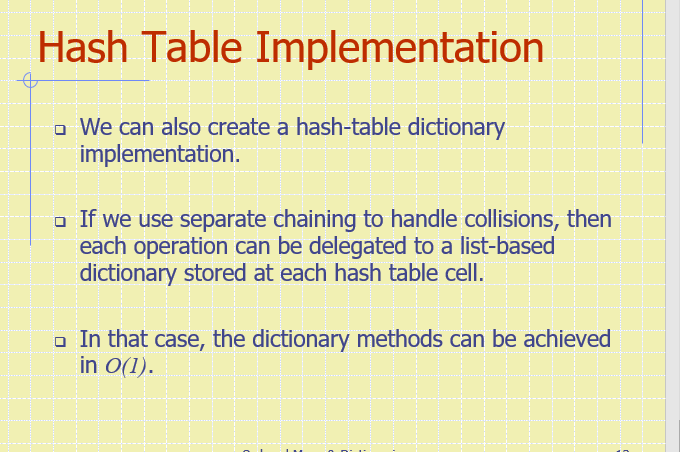


getall,创建一个新的emptylist L,如果e的key=k，加到新List里，return L

put就加到最后就行，因为是Unordered



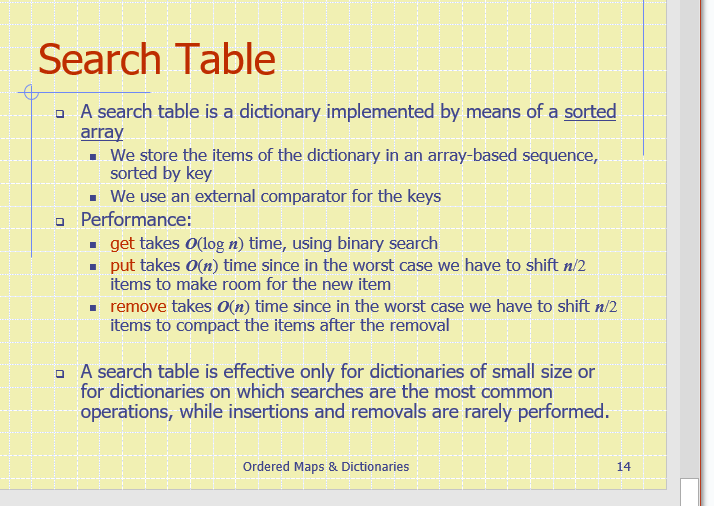
remov



我们可以创造一个hash-table dictionary

如果我们用separate chaining来处理 collision，那么每个操作都可以被分配到不同的list-based dictionary，这些dictionary存储在hash table中

这样dictionary method效率能到达O（1）



search table是一个sorted array组成的dictionary

需要额外comparator来比较key

get Ologn

put On

remove On

适用范围：

small size

只用get，不用Put remove