本节内容

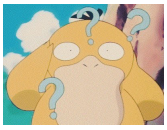
散列查找

散列表（Hash Table）

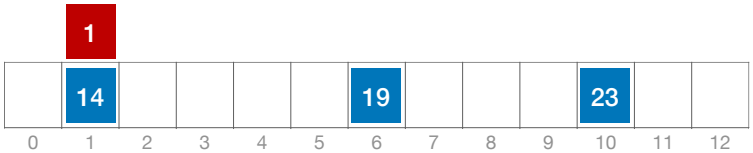
散列表（Hash Table），又称哈希表。是一种数据结构，特点是：数据元素的关键字与其存储地址直接相关

如何建立“关键字”与“存储地址”的联系？

通过“散列函数（哈希函数）”：Addr=H(key)



例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13

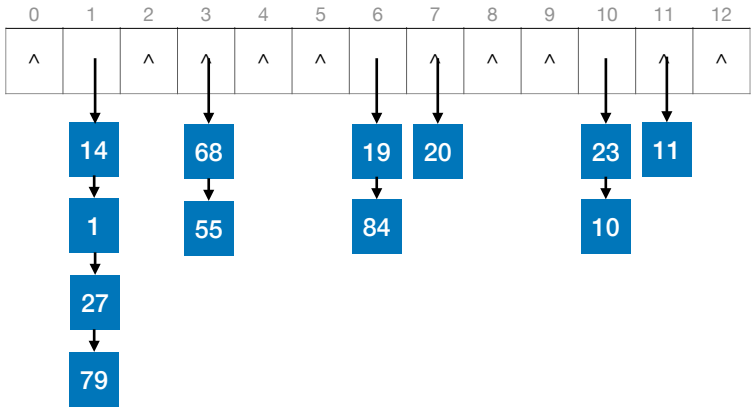


19%13=6
14%13=1
23%13=10
1%13=1

若不同的关键字通过散列函数映射到同一个值，则称它们为“同义词”
通过散列函数确定的位置已经存放了其他元素，则称这种情况为“冲突”

处理冲突的方法——拉链法

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13

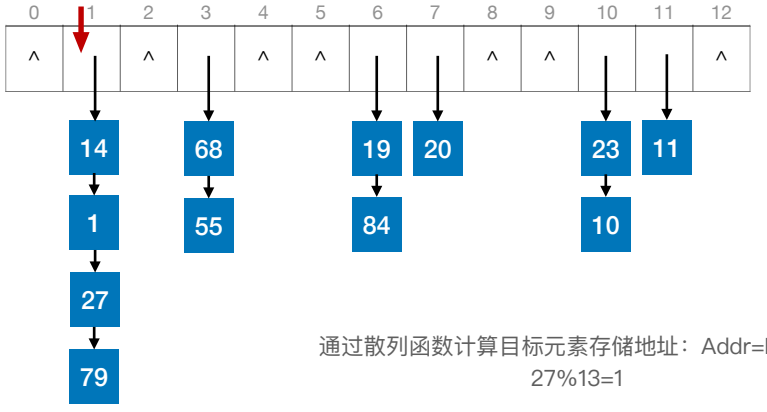


19%13=6　27%13=1
14%13=1　55%13=3
23%13=10　11%13=11
1%13=1　10%13=10
68%13=3　79%13=1
20%13=7
84%13=6

用拉链法（又称链接法、链地址法）处理“冲突”：把所有“同义词”存储在一个链表中

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



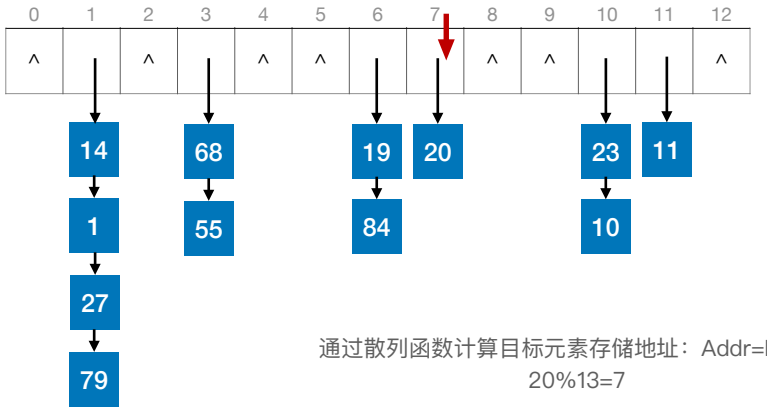
查找目标：27

通过散列函数计算目标元素存储地址：Addr=H(Key)
27%13=1

27的查找长度=3

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



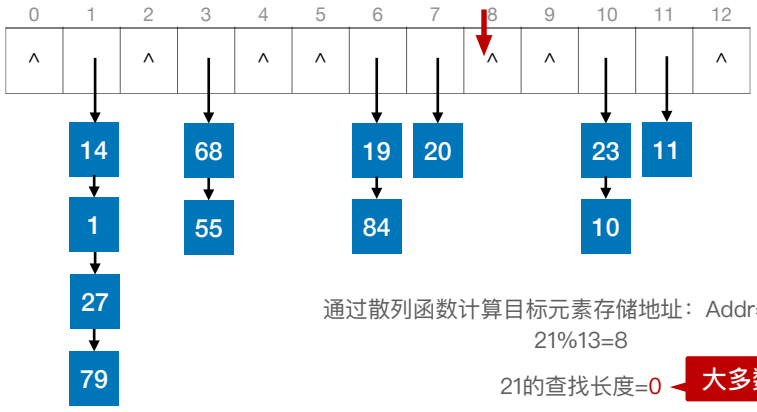
查找目标：20

通过散列函数计算目标元素存储地址：Addr=H(Key)
20%13=7

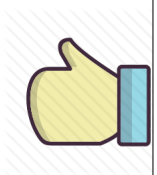
20的查找长度=1

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



查找目标：21



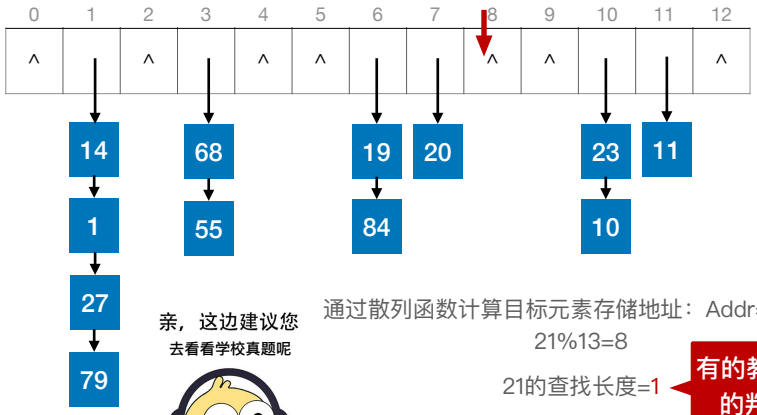
通过散列函数计算目标元素存储地址：Addr=H(Key)
21%13=8

21的查找长度=0 大多数学校的计算方法

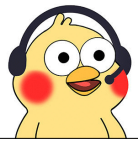
查找长度——在查找运算中，需要对比关键字的次数称为查找长度

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



查找目标：21



通过散列函数计算目标元素存储地址：Addr=H(Key)

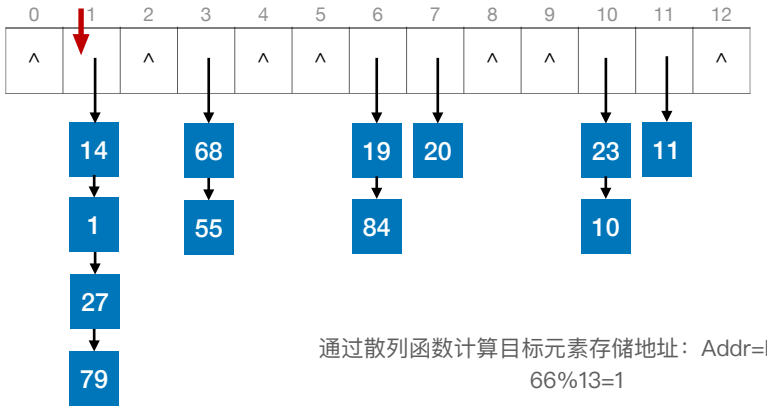
21%13=8

21的查找长度=1

有的教材也会把“空指针”的判定算作一次比较

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



查找目标：66

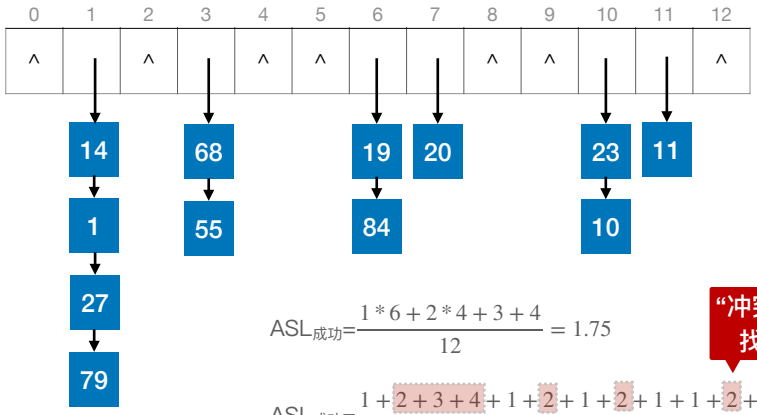
通过散列函数计算目标元素存储地址：Addr=H(Key)

66%13=1

66的查找长度=4

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



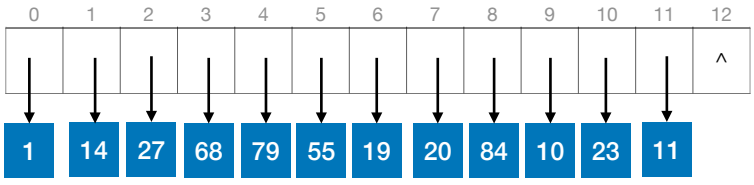
$$ASL_{成功}=\frac{1*6+2*4+3+4}{12}=1.75$$

“冲突”越多，查找效率越低

$$ASL_{成功}=\frac{1+2+3+4+1+2+1+2+1+1+2+1}{12}=1.75$$

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 ~~H(key)=key%13~~



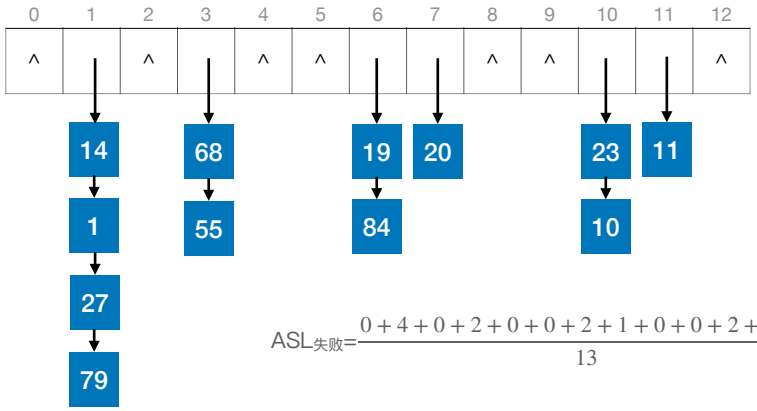
最理想情况：散列查找时间复杂度可到达O(1)

$$ASL_{成功}=\frac{1+1+1+1+1+1+1+1+1+1+1+1}{12}=1$$

如何设计冲突更少的散列函数?

散列查找

例：有一堆数据元素，关键字分别为 {19, 14, 23, 1, 68, 20, 84, 27, 55, 11, 10, 79}，散列函数 H(key)=key%13



$$ASL_{失败}=\frac{0+4+0+2+0+0+2+1+0+0+2+1+0}{13}=0.92$$

装填因子

装填因子会直接影响散列表的查找效率

装填因子α=表中记录数/散列表长度

常见的散列函数

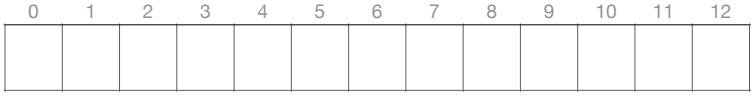
设计目标——让不同关键字的冲突尽可能地少

除留余数法 —— H(key) = key % p

散列表表长为m，取一个不大于m但最接近或等于m的质数p

质数又称素数。指除了1和此整数自身外,不能被其他自然数整除的数

例：散列表表长13，散列函数 H(key)=key%13



例：散列表表长15，散列函数 H(key)=key%13

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

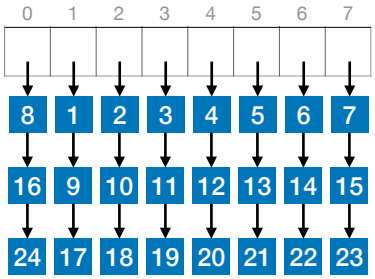
除留余数法 —— H(key) = key % p

散列表表长为m，取一个不大于m但最接近或等于m的质数p

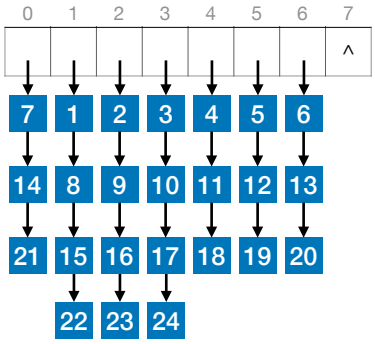
质数又称素数。指除了1和此整数自身外,不能被其他自然数整除的数

设：可能出现的关键字={1,2,3,4,5,6,7,8,9,10……}

散列表表长8，散列函数 H(key)=key%8



散列表表长8，散列函数 H(key)=key%7



常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

除留余数法 —— H(key) = key % p

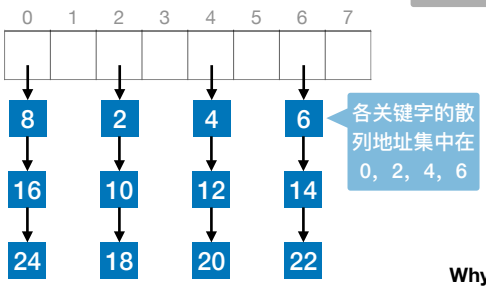
散列表表长为m，取一个不大于m但最接近或等于m的质数p

质数又称素数。指除了1和此整数自身外,不能被其他自然数整除的数

设：可能出现的关键字={2,4,6,8,10,12……}

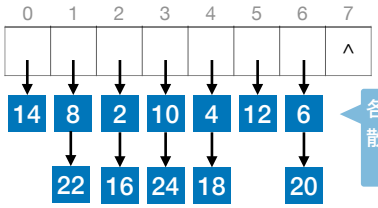
如：中国人更喜欢双数，车牌号双数更多

散列表表长8，散列函数 H(key)=key%8



各关键字的散列地址集中在0，2，4，6

散列表表长8，散列函数 H(key)=key%7



各关键字的散列地址分布均匀

Why? ——用质数取模，分布更均匀，冲突更少。参见《数论》

Tips：散列函数的设计要结合实际的关键字分布特点来考虑，不要教条化

王道考研/CSKAOYAN.COM

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

直接定址法 —— H(key) = key 或 H(key) = a*key + b

其中，a和b是常数。这种方法计算最简单，且不会产生冲突。它适合关键字的分布基本连续的情况，若关键字分布不连续，空位较多，则会造成存储空间的浪费。

例：存储同一个班级的学生信息，班内学生学号为(1120112176~1120112205)
H(key) = key – 1120112176

0	1	2	3	4	5	…	…	…	26	27	28	29
…176	…177	…178	…179	…180	…181	…	…	…	…202	…203	…204	…205

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

数字分析法 —— 选取数码分布较为均匀的若干位作为散列地址

设关键字是r进制数（如十进制数），而r个数码在各位上出现的频率不一定相同，可能在某些位上分布均匀一些，每种数码出现的机会均等；而在某些位上分布不均匀，只有某几种数码经常出现，此时可选取数码分布较为均匀的若干位作为散列地址。这种方法适合于已知的关键字集合，若更换了关键字，则需要重新构造新的散列函数。

例：以“手机号码”作为关键字设计散列函数

138XXXX2875
138XXXX1682
138XXXX9125
…..
199XXXX1684
199XXXX1236

设计长度为10000的散列表，以手机号后四位作为散列地址

0	1	2	3	…	…	9998	9999

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

平方取中法——取关键字的平方值的中间几位作为散列地址。

具体取多少位要视实际情况而定。这种方法得到的散列地址与关键字的每位都有关系，因此使得散列地址分布比较均匀，适用于关键字的每位取值都不够均匀或均小于散列地址所需的位数。

1310^2=1,716,100
1110^2=1,232,100
1300^2=1,690,000
1210^2= 1,464,100
1200^2=1,440,000

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

平方取中法——取关键字的平方值的中间几位作为散列地址。

具体取多少位要视实际情况而定。这种方法得到的散列地址与关键字的每位都有关系，因此使得散列地址分布比较均匀，适用于关键字的每位取值都不够均匀或均小于散列地址所需的位数。

例：要存储整个学校的学生信息，以“身份证号”作为关键字设计散列函数

身份证号码规则：
前1、2位数字表示：所在省份的代码；
第3、4位数字表示：所在城市的代码；
第5、6位数字表示：所在区县的代码；
第7-14位数字表示：出生年、月、日；
第15、16位数字表示：所在地的派出所的代码；
第17位数字表示性别：奇数表示男性，偶数表示女性；
第18位数字是校检码。

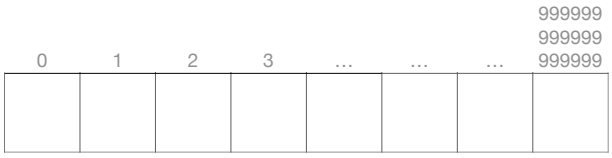
假设学生不超过十万人，可身份证号平方取中间5位

王道考研/CSKAOYAN.COM

常见的散列函数

设计目标——让不同关键字的冲突尽可能地少

例：要存储整个学校的学生信息，以“身份证号”作为关键字设计散列函数



若散列表的长度为10000000000000000000（别数了，有18个0 🐶）
则可以直接用身份证号作为散列地址，且不可能有冲突，查找时间复杂度为O(1)

散列查找是典型的“用空间换时间”的算法，只要散列函数设计的合理，则散列表越长，冲突的概率越低。

知识回顾与重要考点

下次一定