

本节内容

# 散列函数 的构造

# 知识总览

核心目标：散列函数要尽可能地减少“冲突”

## 散列函数的构造

设计散列函数时应该注意什么？

★ 除留余数法

直接定址法

数字分析法

平方取中法

适用不同场景

# 设计散列函数时应该注意什么？

散列函数（哈希函数）： $\text{Addr} = H(\text{key})$  建立了“关键字”→“存储地址”的映射关系。

## 设计散列函数时应该注意什么：

1. 定义域必须涵盖所有可能出现的关键字。

反例： $H(\text{key}) = \sqrt{\text{key}} \% 13$ ，不支持关键字为负值

2. 值域不能超出散列表的地址范围。

反例： $H(\text{key}) = \text{key} \% 15$ ，可能被映射到非法地址13、14

3. 尽可能减少冲突。散列函数计算出来的地址应尽可能均匀分布在整個地址空间。

反例： $H(\text{key}) = (\text{key} \times 13) \% 13$   
任何关键字都会被映射到地址0

4. 散列函数应尽量简单，能够快速计算出任意一个关键字对应的散列地址。

反例： $H(\text{key}) = (\text{key}!) \% 13$ ，若关键字值较大，计算阶乘耗时高

例：散列表长度为13，关键字可能是任一“整数”

0	1	2	3	4	5	6	7	8	9	10	11	12

# 除留余数法



## 除留余数法 —— $H(\text{key}) = \text{key} \% p$

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 $p$

注：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

适用场景：较为通用，只要关键字是整数即可

例：散列表表长  $m=13$ 。13是质数，因此可令散列函数  $H(\text{key})=\text{key}\%13$

0	1	2	3	4	5	6	7	8	9	10	11	12

例：散列表表长  $m=15$ 。不大于15且最接近15的质数是13，因此可令散列函数  $H(\text{key})=\text{key}\%13$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
													^	^

# 拓展：为什么除留余数法要对质数取余？

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

注：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

原因：对质数取余，可以分布更均匀，从而减少冲突

设：可能出现的关键字={2,4,6,8,10,12,14}

散列表表长8，散列函数  $H(key)=key\%8$

0	1	2	3	4	5	6	7

8	2	4	6
	10	12	14

对合数取余，散列地址分布不均匀，易发生冲突

散列表表长8，散列函数  $H(key)=key\%7$

0	1	2	3	4	5	6	7
							^

14	8	2	10	4	12	6
----	---	---	----	---	----	---

对质数取余，散列地址分布均匀，不易发生冲突

Why? —— 取余运算会被“公因子”影响，参见《数论》



# 直接定址法



**直接定址法 ——  $H(\text{key}) = \text{key}$  或  $H(\text{key}) = a * \text{key} + b$**

其中，a和b是常数。这种方法计算最简单，且不会产生冲突。若关键字分布不连续，空位较多，则会造成存储空间的浪费。

适用场景：关键字分布基本连续

例：用散列表存储某班级的30个学生信息，班内学生学号为(1120112176~1120112205)

可令  $H(\text{key}) = \text{key} - 1120112176$

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

# 数字分析法

## 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**。

适用场景：关键字集合已知，且关键字的某几个数码位分布均匀

例：要求将手机用户的信息存入长度为 10000 的散列表。以“手机号码”作为关键字设计散列函数

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

以手机号后四位作为散列地址

0	1	2	3	...	...	9998	9999

# 平方取中法

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。

适用场景：关键字的每位取值都不够均匀。

例：某汽车品牌，以（底盘号、发动机号、外观号）作为一款汽车型号的标识。该品牌有100种车型，欲将这这些车型信息存入长度为100的散列表，如何设计散列函数？

底盘号有：28、34、39、60...

发动机号有：50、52、54...

外观号有：20、25、30、35...

车型号	平方	取中
605030	→	366061300900
345220	→	119176848400
395225	→	156202800625
605435	→	366551539225

	985
×	985
4925	
7880	
8865	
970225	

关键字的各个数码位都分布不均匀

中间几位的值受每个数码位影响



# 知识回顾与重要考点

