11.2 散列函数的构造方法

❖ 一个“好”的散列函数一般应考虑下列两个因素：

1. 计算简单，以便提高转换速度；
2. 关键词对应的地址空间分布均匀，以尽量减少冲突。

❖ 数字关键词的散列函数构造

1. 直接定址法

取关键词的某个线性函数值为散列地址，即

$h(key) = a \times key + b$　　(a、b为常数)

h(key)=key-1990

地址h(key)	出生年份(key)	人数(attribute)
0	1990	1285万
1	1991	1281万
2	1992	1280万
…	……	……
10	2000	1250万
…	……	……
21	2011	1180万

2. 除留余数法

散列函数为：h(key) = key mod p

例： h(key) = key % 17

地址 h(key)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
关键词 key	34	18	2	20			23	7	42		27	11		30		15	

- 这里：p = Tablesize = 17
- 一般，p 取素数

3．数字分析法

分析数字关键字在各位上的变化情况，取比较随机的位作为散列地址

- 比如：取11位手机号码key的后4位作为地址：

  散列函数为：h(key) = atoi(key+7)　(char *key)

如果关键词 key 是18位的身份证号码：

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
3	3	0	1	0	6	1	9	9	0	1	0	0	8	0	4	1	9
省		市		区（县）下属辖区编号		（出生）年份				月份		日期		该辖区中的序号			校验

$h_1(key) = (key[6]-'0')\times10^4 + (key[10]-'0')\times10^3 + (key[14]-'0')\times10^2 + (key[16]-'0')\times10 + (key[17]-'0')$

$h(key) = h_1(key)\times10 + 10$　（当 key[18] = 'x'时）

或　$= h_1(key)\times10 + key[18]-'0'$　（当 key[18] 为'0'~'9'时）

4. 折叠法

把关键词分割成位数相同的几个部分，然后叠加

如： 56793542

```
  542
  793
+ 056
------
 1391
```

h(56793542) = 391

5. 平方取中法

如： 56793542

```
          56793542
        x 56793542
--------------------
322550641 2905764
```

h(56793542) = 641

❖字符关键词的散列函数构造

1．一个简单的散列函数——ASCII码加和法

对字符型关键词key定义散列函数如下：

$h(key) = (\Sigma key[i]) \bmod TableSize$

冲突严重：a3、b2、c1；
eat、 tea；

2．简单的改进——前3个字符移位法

$h(key)=(key[0]\times 27^2 + key[1]\times 27 + key[2]) \bmod TableSize$

3．好的散列函数——移位法

涉及关键词所有n个字符，并且分布得很好：

$$h(key)=\left(\sum_{i=0}^{n-1} key[n-i-1]\times 32^i\right) \bmod TableSize$$

仍然冲突：string、 street、strong、structure等等；
空间浪费：$3000/26^3 \approx 30\%$

❖ 如何快速计算：

$h(\text{“abcde”}) = \text{‘a’}*32^4 + \text{‘b’}*32^3 + \text{‘c’}*32^2 + \text{‘d’}*32 + \text{‘e’}$

```
Index Hash ( const char *Key, int TableSize )
{
    unsigned int h = 0;      /* 散列函数值，初始化为0 */
    while ( *Key != '\0' )   /* 位移映射 */
        h = ( h << 5 ) + *Key++;
    return  h % TableSize;
}
```