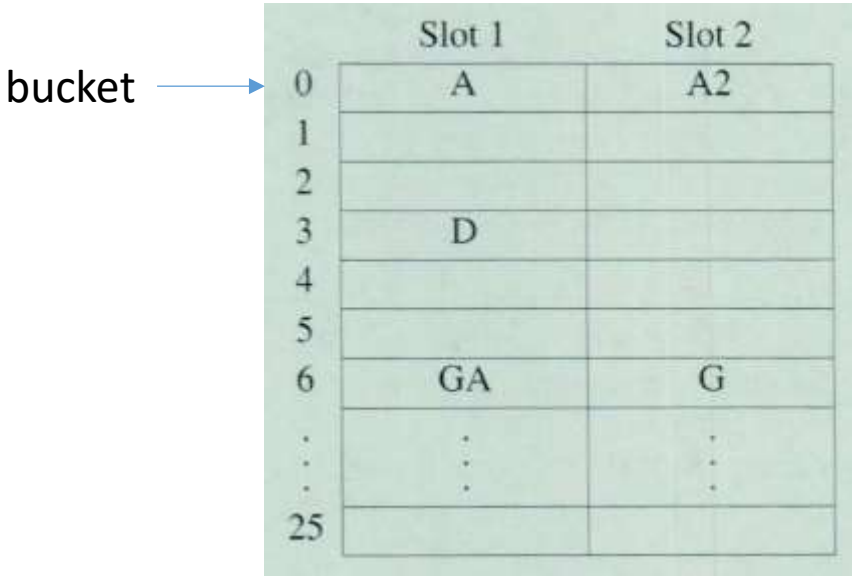
Chapter 08 搜尋 (Searching)(1)

雜湊(1)

- 字典是一種鍵-值對(**key-value pairs**)
  - {“Tom”: 100, “Alice": 200, “Mary": 300}
- 鍵-值對透過鍵將值儲存在一個表格中，此表格稱為**雜湊表(hash table)**。
- 雜湊表是一種類似資料表的索引表格，其中可分為b個**桶(bucket)** 每個桶又可分為s個**槽(slot)**，如下圖所示:
- 桶 :雜湊表中儲存值的位置，每一個位置對應到唯一的一個位址(bucket address)。

- 槽: 每一桶中可能包含好幾個值，而槽指的就是「桶」中可容納的值的個數。通常槽的個數為1，也就是每桶中只能包含一個值。
- 透過一個雜湊函數來計算(或轉換)一個鍵所對應的位址。如鍵為k，h為雜湊函數，則h(k)為對應的位址



雜湊(2)

- 雜湊表的鍵密度(***key density***)是指*n/T,* 其中n 是雜湊表中鍵-值對數目，而T 是所有可能的鍵-值對數目。
- 雜湊表的裝載密度(***Loading density)或***裝載因子***(loading factor)*** α = *n/(sb).*
- **同義字(Synonym)** : 當兩個鍵I1 及I2經雜湊函數運算後所得的位址相同，即h(I1)=h(I2)，則稱I1與I2對於h這個雜湊函數是同義字。
- **碰撞(Collision)**: 當一個位址同時被兩個或兩個以上的鍵所對應時，我們稱此現象為碰撞(Collision)
- **溢位(*overflow*)**:如果資料經過雜湊函數運算後，所對應到的bucket 已滿，則會使bucket發生溢位。

雜湊(3)

- 一個b=26，s=2 的雜湊表。假設有n=10 個不同的鍵-值對，其中每個鍵最多兩個字元，第一個字元必需是英文字母，之後是英文或數字，則
  - 這個表的鍵值密度是10/(26+26*36) 。
  - 這個表的裝載因子α 是10/52= 0.19 。
- 定義為h(x) = x的第一個字母 (如果字母A到Z對應到數字0 到25)
  - 鍵GA,D, A, G, L, A2, A1 , A3, A4 , E 會被對應
    到桶 6, 3, 0, 6, 11, 0, 0, 0, 0, 4
  - 其中A, A1 , A2, A3, A4 是同義字G 和GA 也是。
  - 當GA放入桶3中，之後若要放入G時發生碰撞
  - 若A和A2已在桶0中，此時桶子已滿，若A1要放入則發生溢位。

	Slot 1	Slot 2
0	A	A2
1		
2		
3	D	
4		
5		
6	GA	G
⋮	⋮	⋮
25		

雜湊函數及解決溢位的方法

- 雜湊函數
  - 除法(Division Method)
  - 平方取中法(Midsquare Method)
  - 疊合法(Folding Method)
  - 位數分析法(Digit Analysis)
- 解決溢位的方法
  - 線性探測(Linear Probing)
  - 重複雜湊法(Rehashing)
  - 串列(chaining)

雜湊函數

- 容易運算
- 盡量縮小碰撞發生
- 雜湊函式對於隨機的輸入也應該是沒有偏差的
  - 如果隨機從鍵空間(也就是所有可能的鍵之集合) 取出一個識別字x對所有的桶其h(x)=i 的機率都是1/b 。這樣一來隨機抽取的x會在b個桶裡的某個桶中的機會都是相同的，滿足這個特性的雜湊函式稱之為**均勻雜湊函式 (uniform hash function)** 。

除法(Division Method)

h(k)=k mod m

k：代表資料的鍵。

m：代表桶的數目。

【例】

鍵集合 k ={12,65,70,99,33,67,48}。

令m=13，則利用 h(k)=k mod 13。

索引	索引	資料
0	0	65
1	1	
2	2	67
3	3	
4	4	
5	5	70
6	6	
7	7	33
8	8	99
9	9	48
10	10	
11	11	
12	12	12

平方取中法(Midsquare Method)

- 中間平方法和除法相當類似，它是把資料乘以自己，之後再取中間的某段數字當作桶的索引

【例】將12,65,70,99,33,67,51平方後如下:

144,4225,4900,9801,1089,4489,2601

我們取百位數及十位數作為鍵， 分別為

14,22,90,80,08,48,60

- 上述這7 個數字的數列就是對應原先12,65,70,99,33,67,51等7 個數字存放在100個位址空間的桶索引

疊合法(Folding Method)

- 將鍵 k 切割成若干區塊，除了最後一個區塊，其他區塊長度都相同，之後將所有區塊相加，就是它的桶索引
- 相加有兩種方法：移動折疊法(shift folding)及邊界折疊法(folding at the boundaries)
  - 移動折疊法直接相加
  - 邊界折疊法：將奇數位區塊或偶數位區塊的數字反轉，再行相加

設鍵 k=187249653，且 m=1000。則首先將 k 切成三個區塊：187、249、653

移動折疊法：

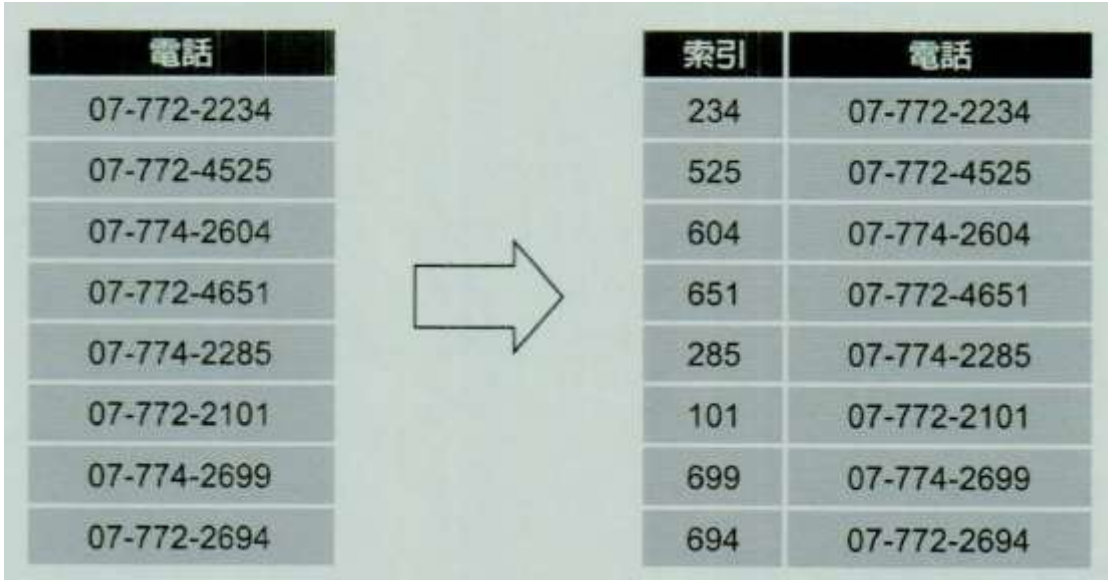
187 + 249 + 653 = 1089。

邊界折疊法：

781+249+356 =1386

位數分析法(Digit Analysis)

- 數位分析法適用於資料不會更改，且為數字型態的靜態表。 。在決定雜湊函數時先逐一檢查資料的相對位置及分佈情形，將重複性高的部份刪除。
- 例如下面這個電話表， 它是相當有規則性的， 除了區碼全部是07 外， 在中間三個數字的變化也不大，假設位址空間大小m=999，我們必須從下列數字擷取適當的數字， 即數字比較不集中，分佈範圍較為平均(或稱亂度高) ，最後決定取最後那四個數字的末三碼。故最後可得雜湊表為:



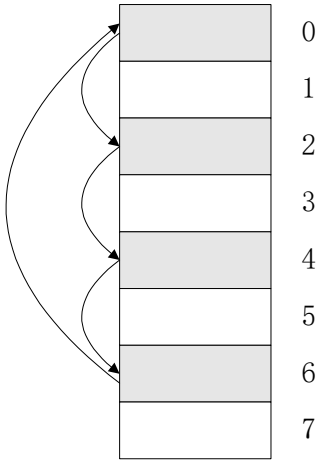
電話
07-772-2234
07-772-4525
07-774-2604
07-772-4651
07-774-2285
07-772-2101
07-774-2699
07-772-2694

索引	電話
234	07-772-2234
525	07-772-4525
604	07-774-2604
651	07-772-4651
285	07-774-2285
101	07-772-2101
699	07-774-2699
694	07-772-2694

解決碰撞的方法
Open Addressing (1)

- **線性探測(Linear Probing)**
  - 找到一個最近的沒有填滿的桶

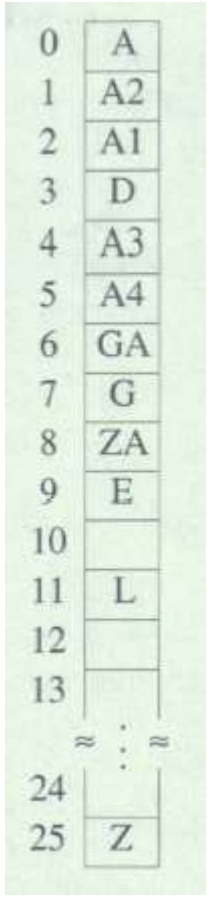
原理上，當 $h(k_i) = \alpha = h(k_j)$時，表示 k_j 與 k_i 撞在一起了。這時候，我們可以沿著位址 α 的下面，逐號地尋找一個空的位置來讓 k_j 棲身。如果一路找到位址空間的最後一號位置都沒有找到空位置的話，則可以再折回位址空間的第一號位置，再從上往下去搜尋，直到找到 α 的前一號位置為止。



練習08-01

- Assume we have a 26-bucket table with one slot per bucket and the following key: GA, D, A, G, L, A2, A1 , A3, A4, Z, ZA, E.

  the hash function *h(k)* = first character of *k.*



作業08-01

- 呈以上練習題，請寫一個程式，能依hash function的規則將key存在相對應的bucket(陣列)。

作業08-02

- 呈上題，請寫一個function，傳入一個key，能在以上hash table找到此key存在陣列的index，並回傳此key存在陣列的index，若沒找到則回傳-1。

解決碰撞的方法
Open Addressing (2)

- 二次探測 **(quadratic probing)**
  - 平方探測

  當溢位發生峙，下一次搜尋的位址是(h(k)+e) mod b 與(h(k)-e) mod b，即讓資料值加或減i的平方，如

  第一次尋找: h(k)

  第二次尋找: $(h(k)+ 1^2)\%b$

  第三次尋找: $(h(k)- 1^2)\%b$

  第四次尋找: $(h(k)+ 2^2)\%b$

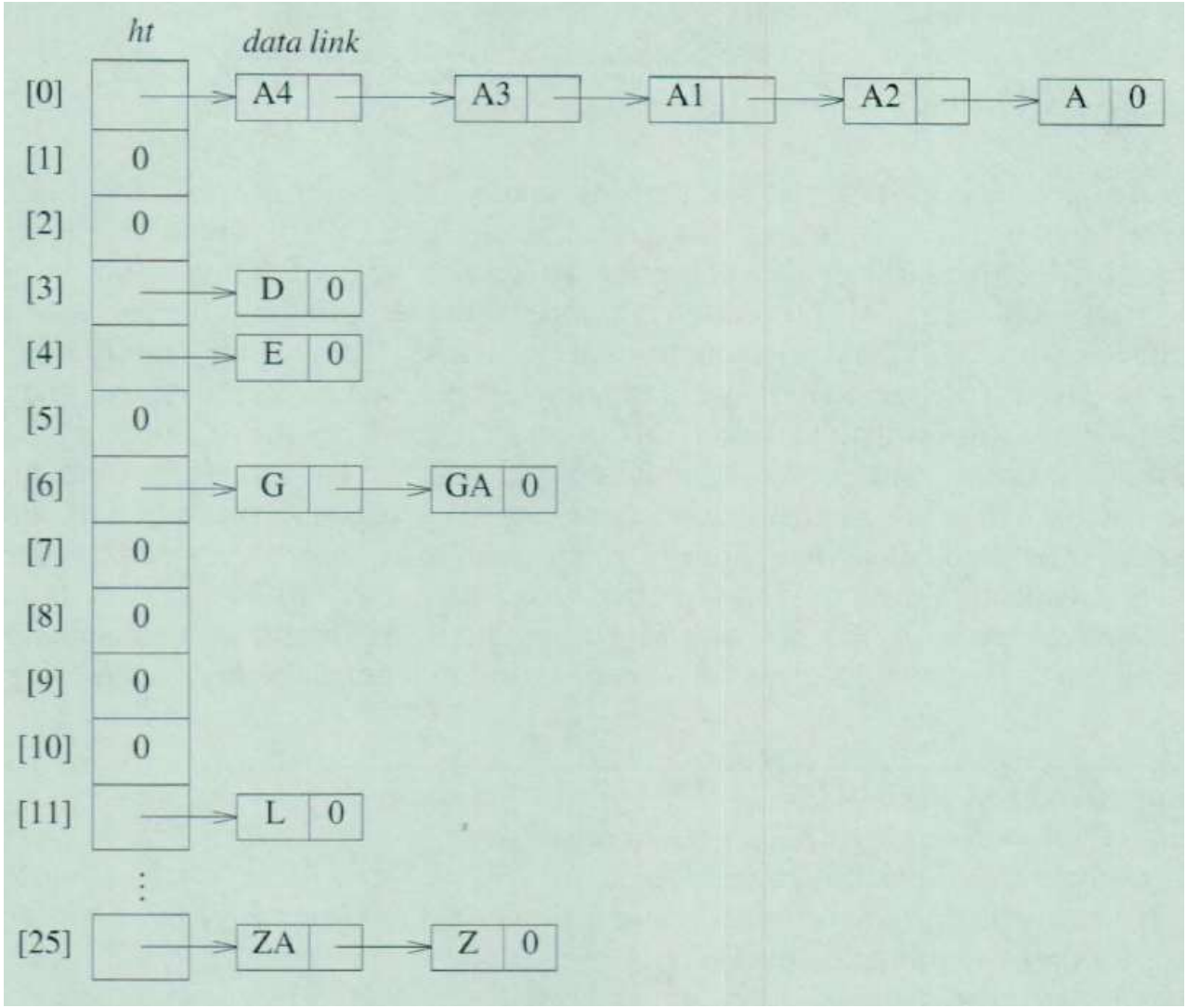
  第五次尋找: $(h(k)- 2^2)\%b$

  …
  - **重複雜湊法(Rehashing)**

    當有碰撞現象產生時，再改用事先準備好的其它種雜湊函數計算出不同的位址，若再碰撞，則再找出另一個雜湊函數來計算位址，依此類推。因此，重複赫序法需要事先準備一些雜湊函數 h_1，h_2，…，h_r，以備不時之需。

解決碰撞的方法
串列(chaining)

將雜湊表的所有空間建立n 個串列，最初的預設值只有n個串列首。如果發生碰撞就把相同位址之鍵值鏈結在串列首的後面，形成一個鏈結串列，直到所有的可用空間全部用完為止。



作業08-03

- Assume we have a 26-bucket table with one slot per bucket and the following key-value:

  {“GA”:11, “D”:12, “A”:13, “G”:14, “L”:15, “A2”:16, “A1”:7 “A3”:18, “A4”:19, “Z ”:20, “ZA ”:21, “E”:22}

  the hash function *h (k)* = first character of *k.*

  請寫一個程式，能依hash function的規則將key存在相對應的bucket。(使用串列法)

作業08-04

- 呈上題，請寫一個function，傳入一個key，能在以上hash table找到此key存的值，並回傳此key的值，若沒找到則回傳-1。

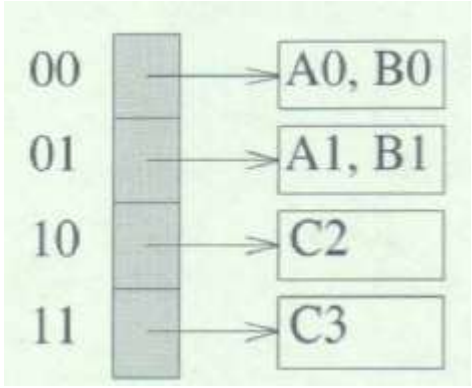
動態雜湊(dynamic hashing)(1)

- 當要存的資料量比hashing的bucket還多時，需要增加bucket的數量(增加一倍)，且須重新定址
  - 如用division hashing，原有b buckets, 增為2b個則， division hashing的divisor由b變為2b
- 動態雜湊又稱可伸展雜湊法(extensible hashing)

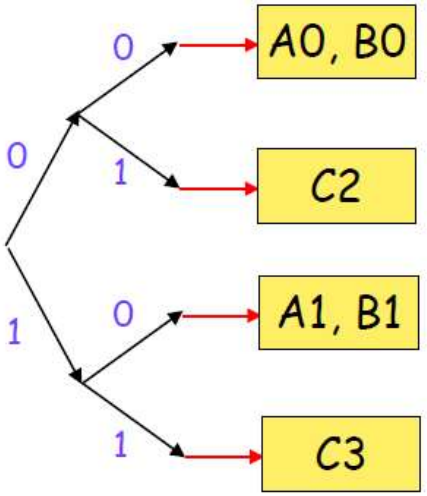
動態雜湊(dynamic hashing)(2)

k	$h(k)$
A0	100 000
A1	100 001
B0	101 000
B1	101 001
C1	110 001
C2	110 010
C3	110 011
C5	110 101

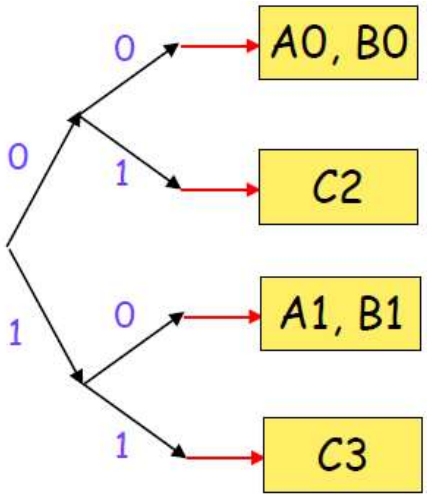
hash function *h(k)* that transforms keys into 6-bit，
Table size is 4.
Each bucket has 2 slot.

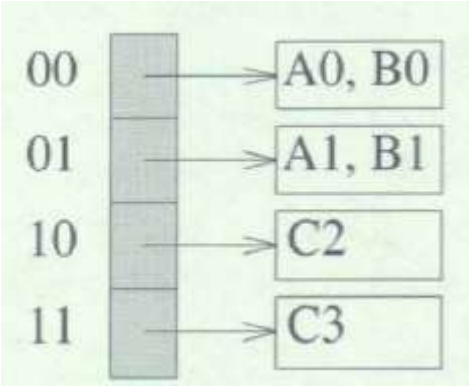


Insert A0, B0, A1, B1, C2, C3



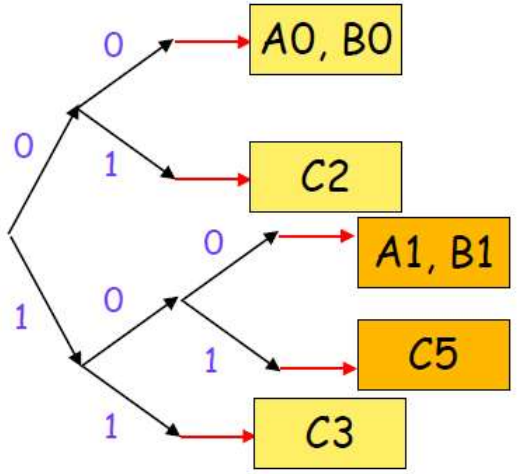
動態雜湊(dynamic hashing)(3)

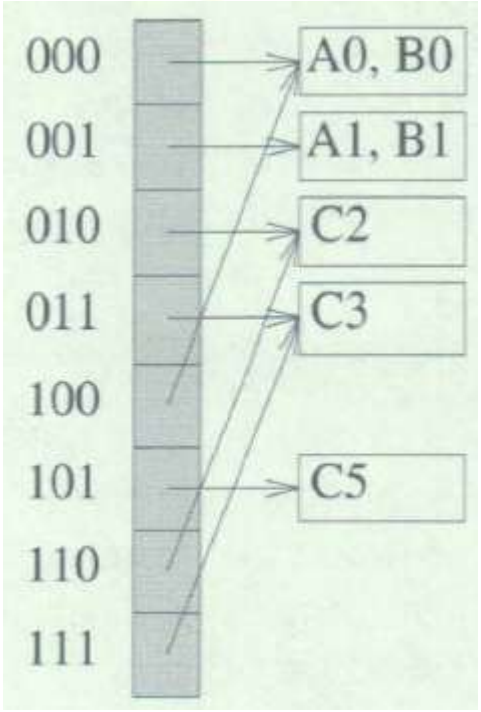




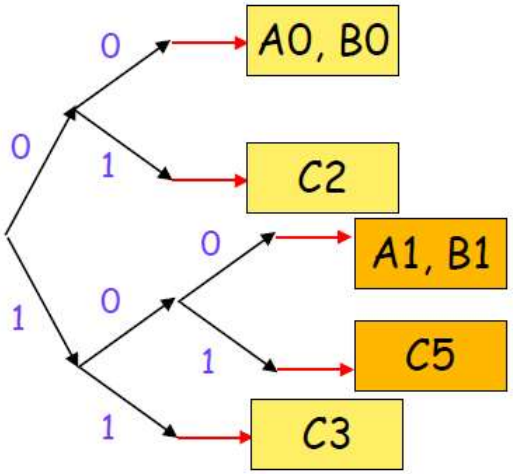
inserting C5

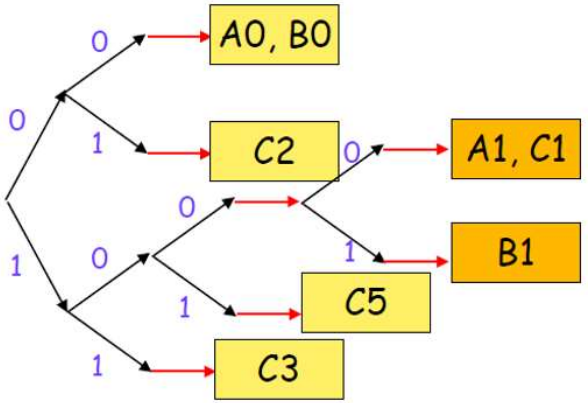
k	$h(k)$
A0	100 000
A1	100 001
B0	101 000
B1	101 001
C1	110 001
C2	110 010
C3	110 011
C5	110 101



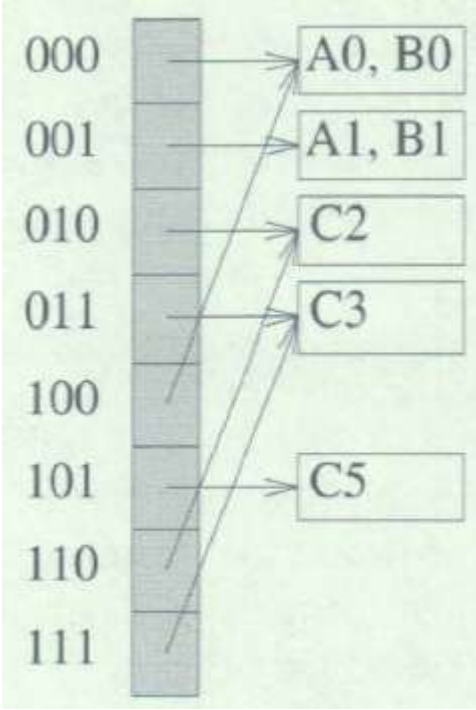


動態雜湊(dynamic hashing)(4)

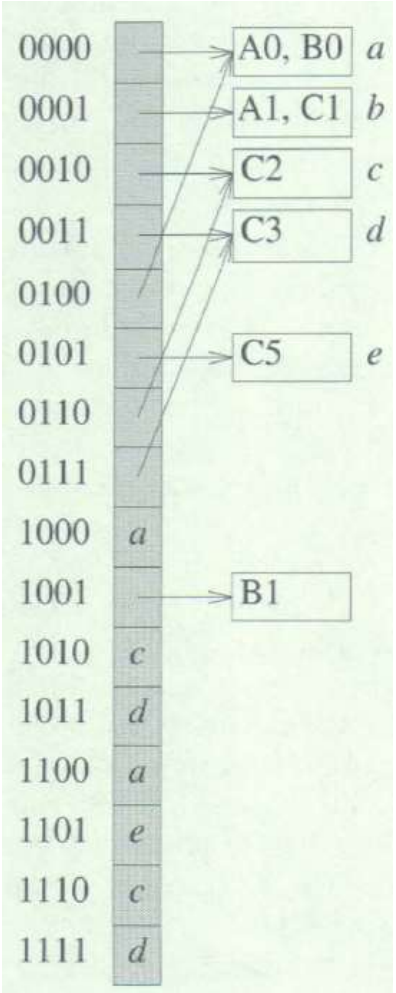




inserting C1



k	$h(k)$
A0	100 000
A1	100 001
B0	101 000
B1	101 001
C1	110 001
C2	110 010
C3	110 011
C5	110 101



循序搜尋法(Searching Method)
資料未排序

- 循序搜尋法(Searching Method)鍵(Key)

1. 從第一筆記錄開始一筆一筆地比較鍵直到檔案的最後一筆為止
2. 檔案若有N筆記錄，就須比對鍵N次
3. 又稱線性搜尋法(Linear Searching Method)

記錄編號	鍵值	搜尋條件	
		鍵值 = 153	鍵值 > 100
1	17	1(x)	1(x)
2	25	2(x)	2(x)
3	78	3(x)	3(x)
4	271	4(x)	4(o)
5	5	5(x)	5(x)
6	44	6(x)	6(x)
7	322	7(x)	7(o)
8	153	8(o)	8(o)
9	87	9(x)	9(x)
10	18	10(x)	10(x)
11	57	11(x)	11(x)
12	358	12(x)	12(o)
13	471	13(x)	13(o)
14	31	14(x)	14(x)
15	63	15(x)	15(x)
16	153+	16(o)	16(o)
比較次數		16	16
成功次數		2	6

循序搜尋法(Searching Method)
資料已排序

記錄編號	鍵值	搜尋條件	
		鍵值 = 153	鍵值 > 100
1	5	1(x)	1(x)
2	17	2(x)	2(x)
3	18	3(x)	3(x)
4	25	4(x)	4(x)
5	31	5(x)	5(x)
6	44	6(x)	6(x)
7	57	7(x)	7(x)
8	63	8(x)	8(x)
9	78	9(x)	9(x)
10	87	10(x)	10(x)
11	153	11(o)	11(o)
12	153^{+}	12(o)	
13	271	13(x)	
14	322		
15	358		
16	471		
比較次數		13	11
成功次數		2	16-11+1=6

二元搜尋法(Binary Searching Method)

- 從N筆記錄中尋找鍵值K之二元搜尋法可簡述如下：(沒有重複鍵值)

1. 將記錄依鍵值不遞減之順序排序。
2. 令K_i = 剩餘記錄中的第中間筆記錄之鍵值。

   若 $K_i > K$：(失敗)，剩餘檔案為前半段。

   若 $K_i = K$：(成功)，終止程式。

   若 $K_i < K$：(失敗)，剩餘檔案為後半段。
3. 繼續步驟 2，直到檔案無法再分割為止。

二元搜尋法(Binary Searching Method)

- 從N筆記錄中尋找鍵值K之二元搜尋法可簡述如下：(有重複鍵值)

1. 將記錄依鍵值不遞減之順序排序。
2. 令K_i = 剩餘記錄中的第中間筆記錄之鍵值。

   若 $K_i > K$：(失敗)，剩餘檔案為前半段。

   若 $K_i = K$：(成功)，繼續分別比對第i-1，i-2，...，以及第i+1，i+2，...，直到失敗為止。

   若 $K_i < K$：(失敗)，剩餘檔案為後半段。
3. 繼續步驟 2，直到檔案無法再分割為止。

二元搜尋法(Binary Searching Method)

記錄編號	鍵值	搜尋條件	
		鍵值 = 153	鍵值 > 100
1	5		
2	17		
3	18		
4	25		
5	31		
6	44		
7	57		
8	63	1(x)	1(x)
9	78		
10	87	4(x)	4(x)
11	153	3(o)	3(o)
12	153^{+}	2(o)	2(o)
13	271	5(x)	
14	322		
15	358		
16	471		
比較次數		5	4
成功之記錄編號		11, 12	11, 12, 13, 14, 15, 16

作業08-05

- 一個大小為20的整數陣列，陣列中的值已由小到大排序(有重複鍵值)，請寫一個function該function傳入一個整數值，使用二元搜尋法找出此值的開始index及最終index，並回傳開始index及最終index ，若沒找到則開始index=-1及最終index=-1