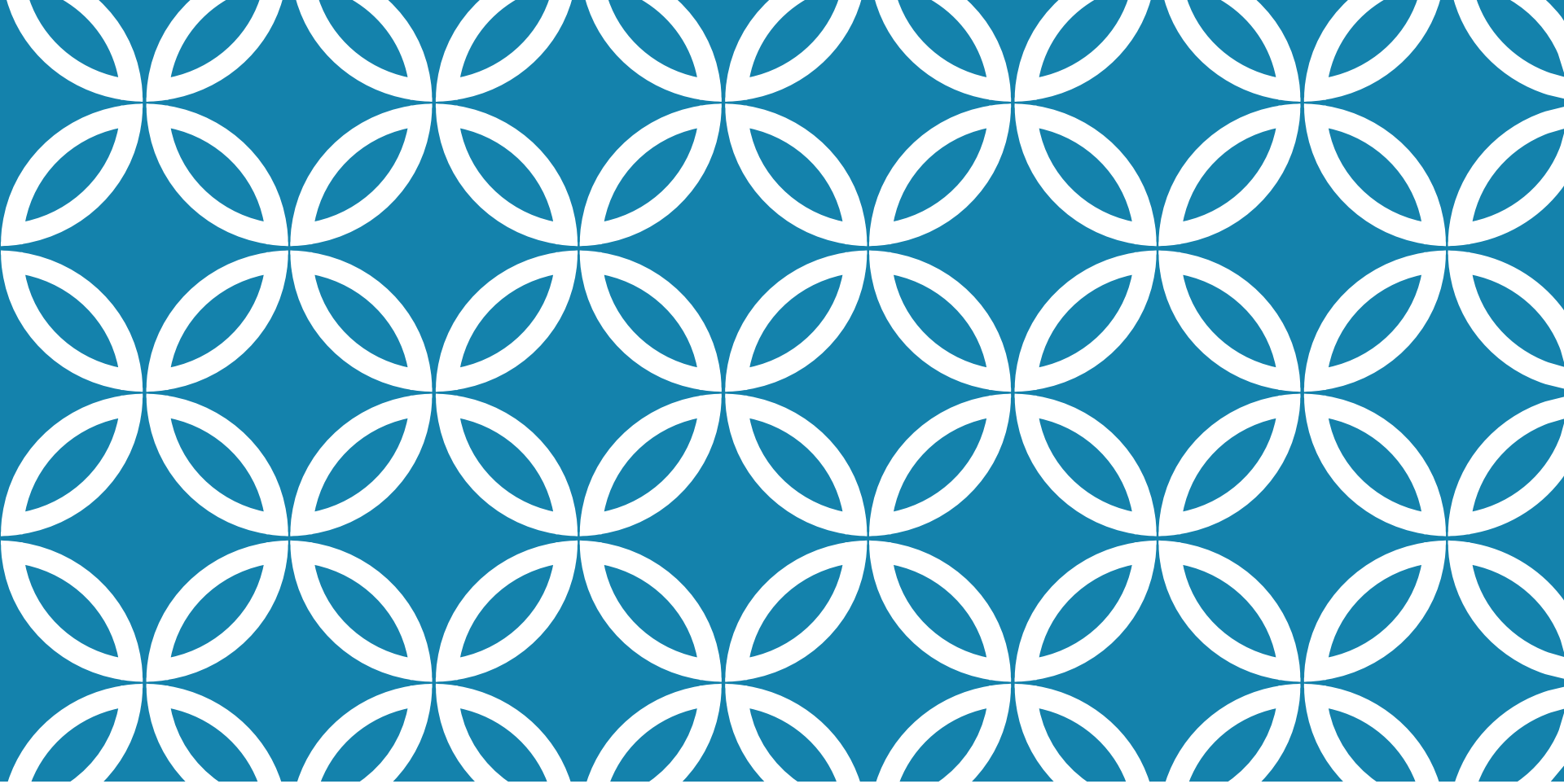


鏈結串列

資料結構
鍾宜玲

串列 (LIST)

有次序排列的資料，稱為串列 (list)，又稱為循序串列 (sequence list) 或線性串列 (linear list)。

串列中的資料可能會被

(1) 刪除

(2) 更改

(3) 插入

(4) 讀取

(5) 計算資料的個數

陣列製作串列之優缺點

優點：

- 容易製作：宣告即可。
- 容易隨機存取資料：利用索引對應。

缺點：

- 刪除、插入及更改資料會造成資料移動頻繁，減少系統效率。
- 必須事先宣告記憶體空間大小，如果無法預知記憶體的使用量，則可能因宣告的陣列不夠大，造成記憶體使用不足，或是因宣告太大而浪費了空間。

如何改進缺點？

動態記憶體配置 (DYNAMIC MEMORY ALLOCATION)

分配動態記憶體，語法如下：

```
指標變數=(型態 *) malloc( 記憶體大小 )；
```

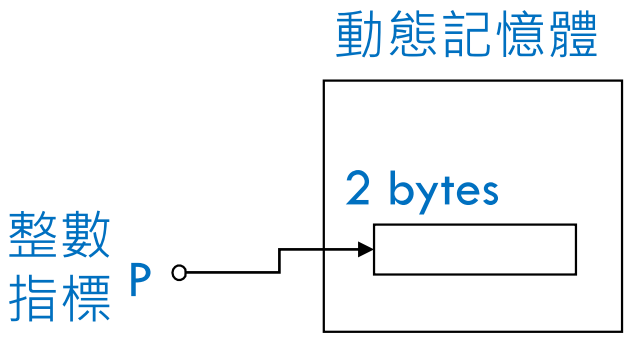
- malloc( ) 函數將會傳回分配的記憶體之起始位址，利用指標儲存此動態分配的記憶體位址，爾後即可透過此指標存取動態記憶體之資料。

利用函數free() 歸還記憶體給系統，語法如下：

```
free( 指標變數 );
```

範例

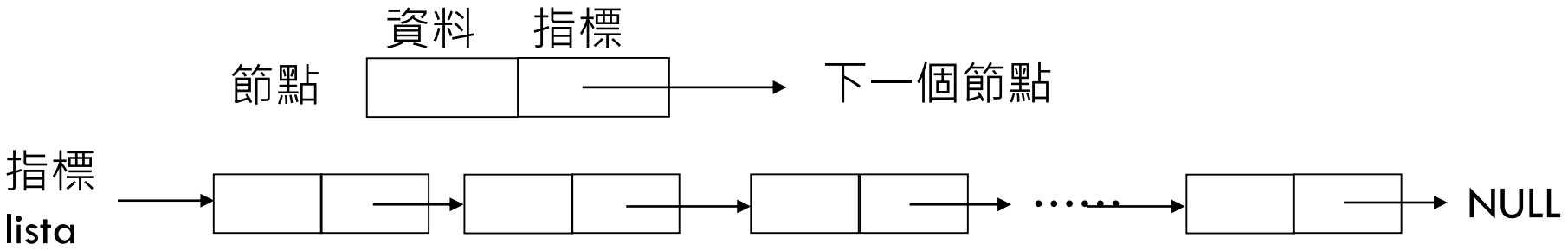
```
int *p;                                  /* 宣告了一個整數指標p */
p=(int *)malloc( sizeof(int) );  /*動態分配整數空間*/
*p=5;                                    /*將整數5 存入此動態記憶體內*/
free(p);                                 /*歸還記憶體*/
```

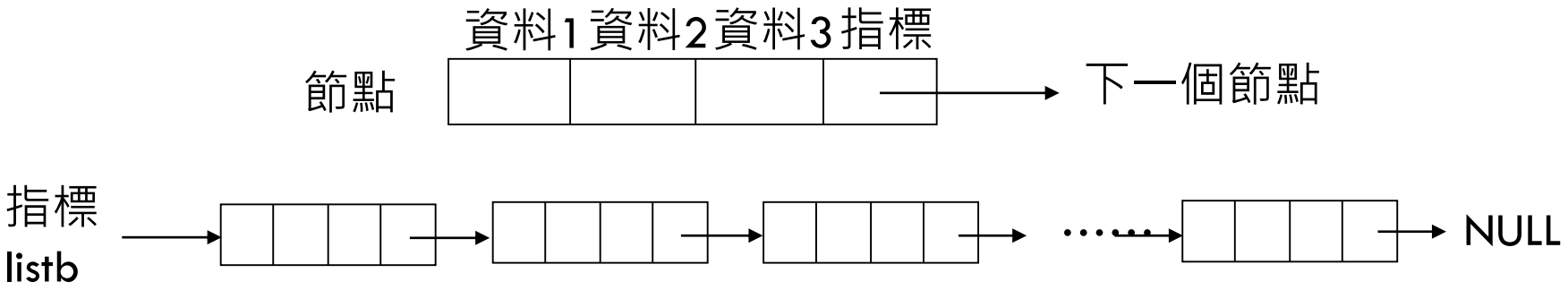


鏈結串列的定義與表示法

- 由動態記憶體分配的節點 (node) 串接而成。
- 由一個開頭指標指向第一個節點。
- 每個節點包含資料值與儲存下一個節點位址的指標，此指標稱為鏈結 (link)。
- 若某節點無下一節點，則此節點的鏈結為空指標，記為NULL。

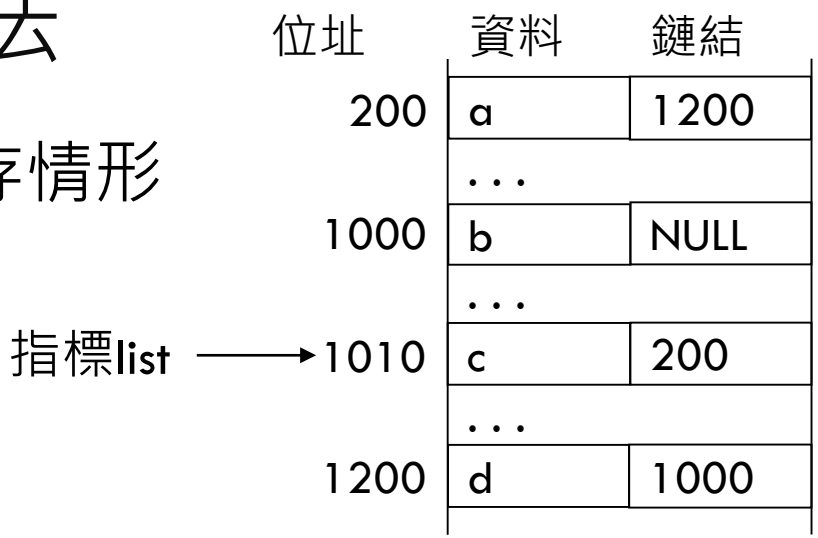
串列


資料
指標
節點
下一個節點
指標
lista
NULL

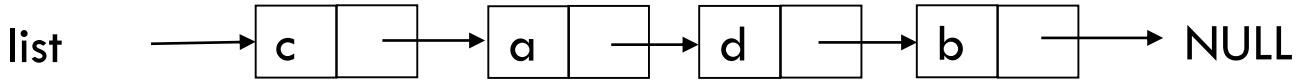

資料1 資料2 資料3 指標
節點
下一個節點
指標
listb
NULL

鏈結串列圖示法

資料在記憶體內的儲存情形



鏈結表示法



說明串列 list

- 第一個節點：位址1010，資料為 c
- 第二個節點：位址為200，資料為 a
- 第三個節點：位址1200，資料為 d
- 最後節點　：位址1000，資料為 b

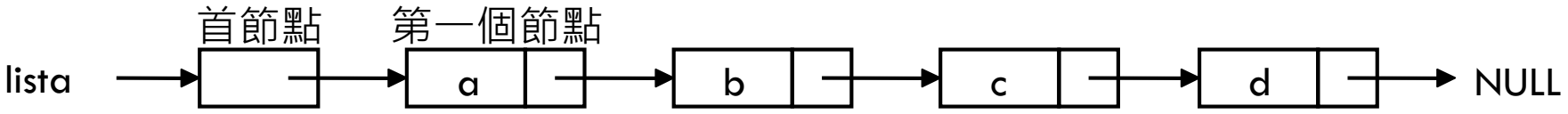
首節點(HEAD NODE)

鏈結串列在實際應用上常會在第一個節點之前另外增加一個首節點 (head node)，以方便程式的寫作。

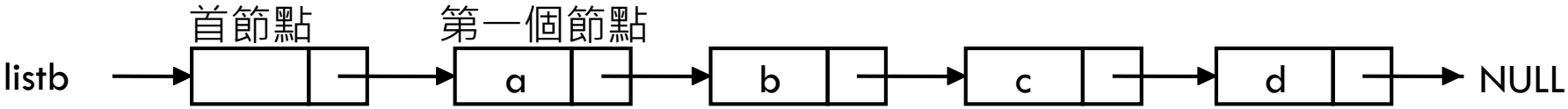
首節點可以與一般節點相同也可以不同，即可以有資料欄，也可以沒有。

若有資料欄，尚可用來儲存一些額外的資料。

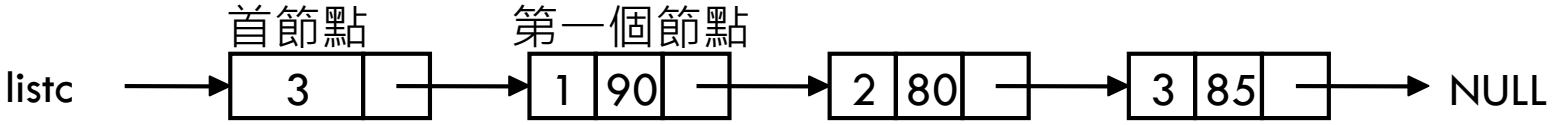
含首節點的各種鏈結串列



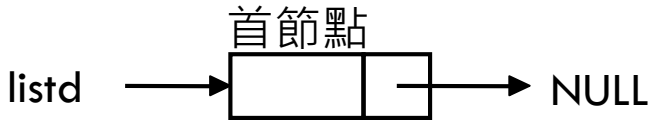
(a) 首節點只有一個鏈結欄



(b) 首節點之資料欄不存資料



(c) 首節點儲存人數



(d) 首節點的空鏈結串列