資料結構 HW2 檢討

2014/11/17

Basic

題目內容

- 要寫一個雙向鏈結串列 (doubly linked list)，並且支援以下功能
- 基本功能
  - "N"：移至下一個
  - "P"：移至上一個
  - "R"：回報該節點之數值，不影響當前指標

題目內容 (Cont.)

- 刪除
  - "DN"：刪除下一個，不影響當前指標
  - "DP"：刪除上一個，不影響當前指標

題目內容 (Cont.)

- 插入
  - "IN"：插入下一個(該節點的內容數值+0.5作為下一個節點的值)，不影響當前指標
  - "IP"：插入上一個(該節點的內容數值-0.5作為上一個節點的值)，不影響當前指標

實作要求

- 使用雙向鏈結串列

```
struct dnode{
    float num;
    struct dnode* prev;
    struct dnode* next;
};
```

實作要求 (Cont.)

- newDnode()函數

```
struct dnode* newDnode(
    float num,
    struct dnode* prev,
    struct dnode* next){
    // your implementation
}
```

- 改自 06_list.c newRoom 函數

輸入

- data.txt: 每行有一個浮點數
  - float num;
  - fscanf(fp, “%f”, num);
- inst.txt: 每行有一個指令
  - char cmd[3];
  - fscanf(fp, “%s”, cmd);

需要的功能

- 基本功能的部分
  - N & P: 指標需要向前或向後動
    - 我們需要一個可以動的指標
    - struct dnode *ptr;
  - R: 印出指標指到的 num
    - printf(“%f” , ptr->num);

需要的功能 (Cont.)

- 刪除的功能
  - DN & DP: 刪掉前面或後面的 node
    - 要 free 要刪掉的 node
    - 刪掉之後要把相關的 prev 和 next 接好
    - 要做的事情滿多滿複雜的
    - 我們需要兩個函數
    - delPrev(struct dnode* node){…}
    - delNext(struct dnode* node){…}

需要的功能 (Cont.)

- 插入的功能
  - IN & IP: 在前面或後面插入新 node
    - 我們有 newDnode 可以用!!
    - 要把新 node 的 prev 和 next 接好
    - 當然也要把其他相關的 node 接好
    - 我們需要兩個函數
    - insertPrev(float num, struct dnode* node){…}
    - insertNext(float num, struct dnode* node){…}

開始實作囉

先來複習一下需要的函數

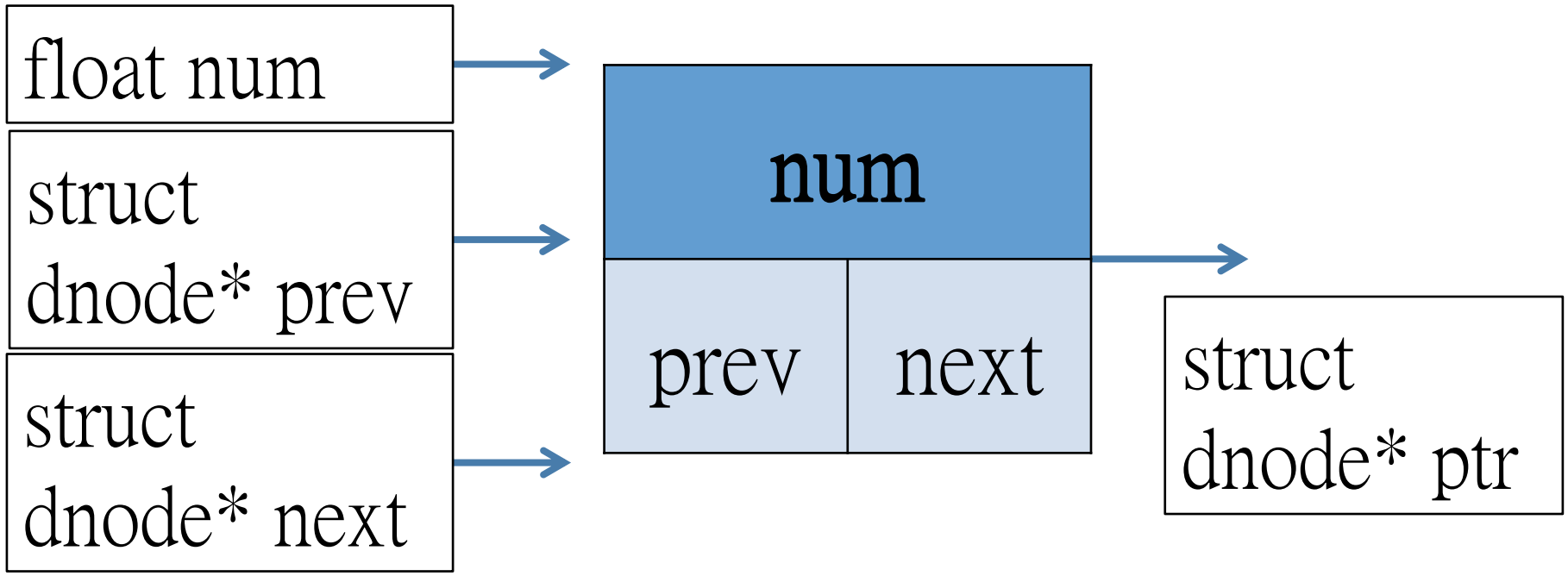
- newDnode: 新增一個 node
- delPrev: 刪掉前一個 node
- delNext: 刪掉後一個 node
- insertPrev: 插入到目前 node 之前
- insertNext: 插入到目前 node 之後

先來複習一下需要的函數

- newDnode: 新增一個 node
- delPrev: 刪掉前一個 node
- delNext: 刪掉後一個 node
- insertPrev: 插入到目前 node 之前
- insertNext: 插入到目前 node 之後

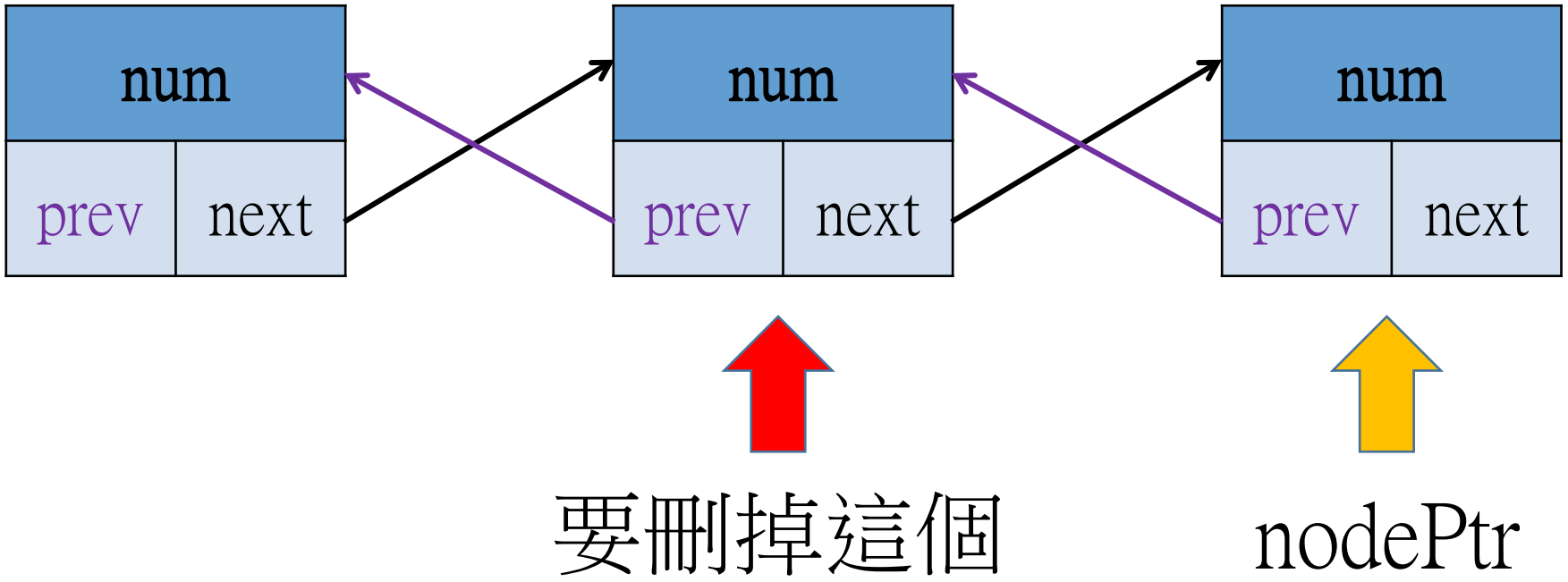
newDnode

- 傳入 num, prev 指標, next 指標
- 回傳新創 node 的指標

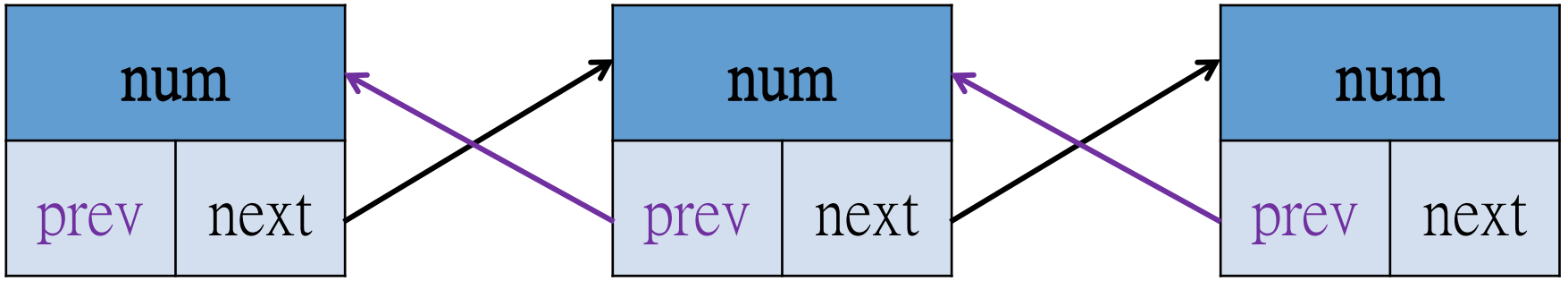


delPrev

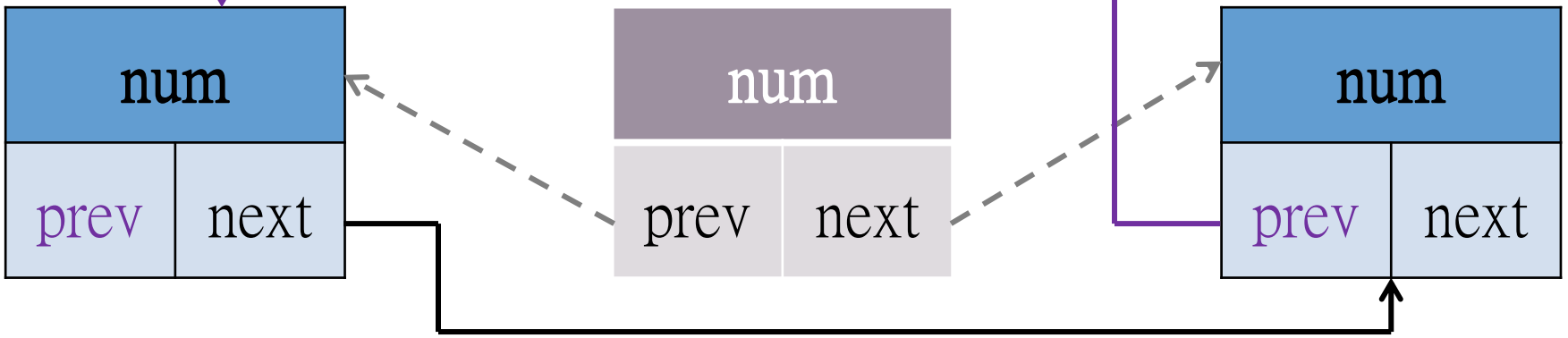
- 傳入 node 指標
- 不用回傳值



delPrev (Cont.)

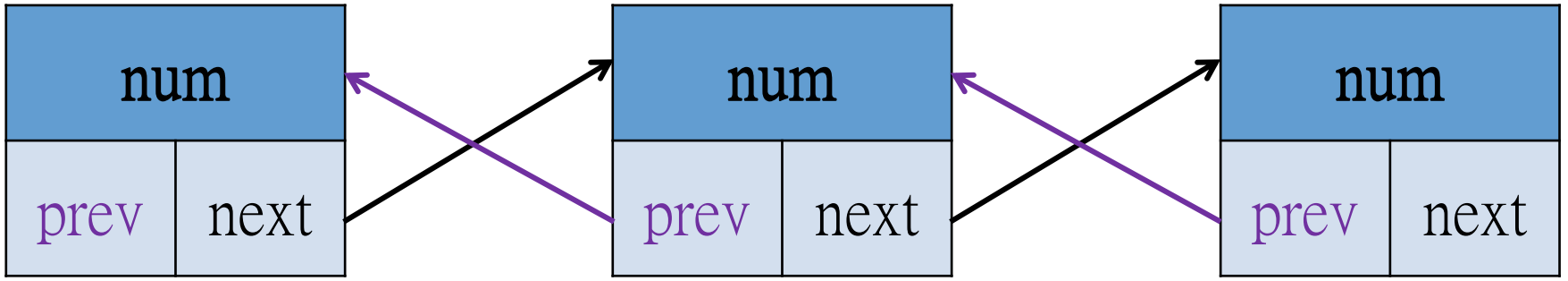


step 1: node->prev = node->prev->prev



step 2: node->prev->next = node

delPrev (Cont.)

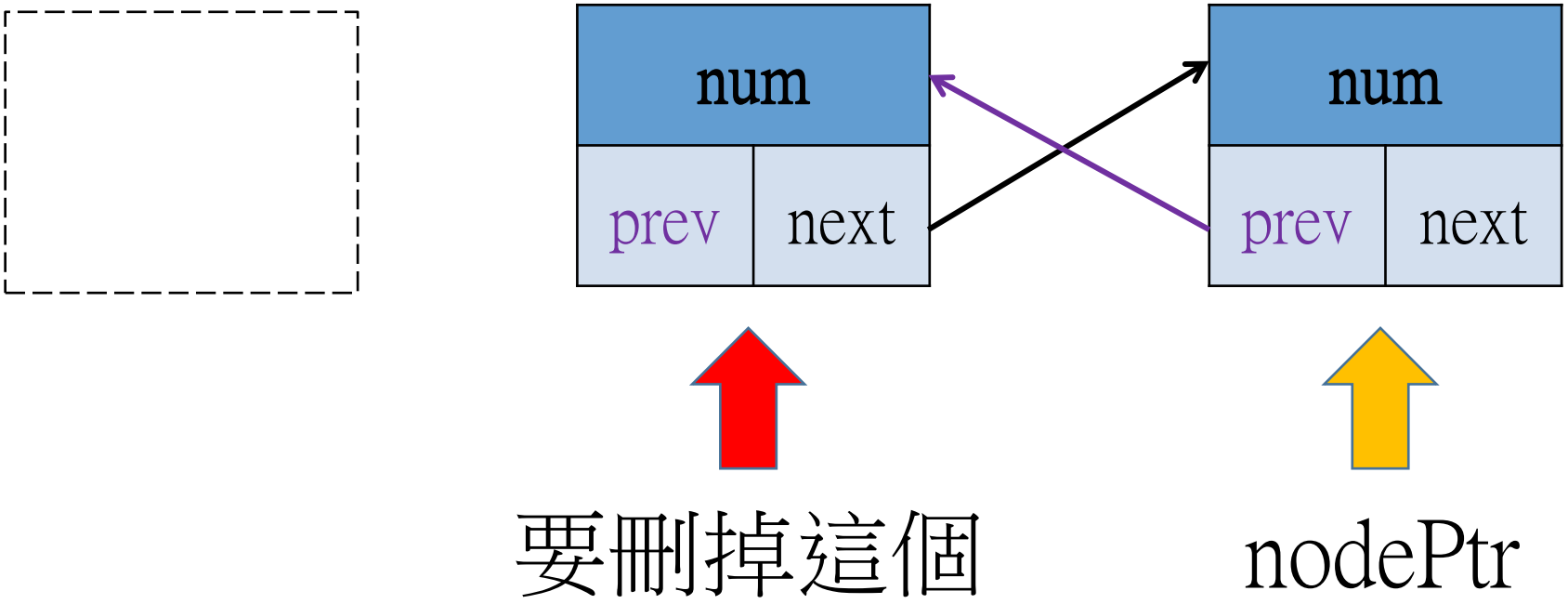


step 1: nodePtr->prev = nodePtr->prev->prev

step 2: nodePtr->prev->next = nodePtr

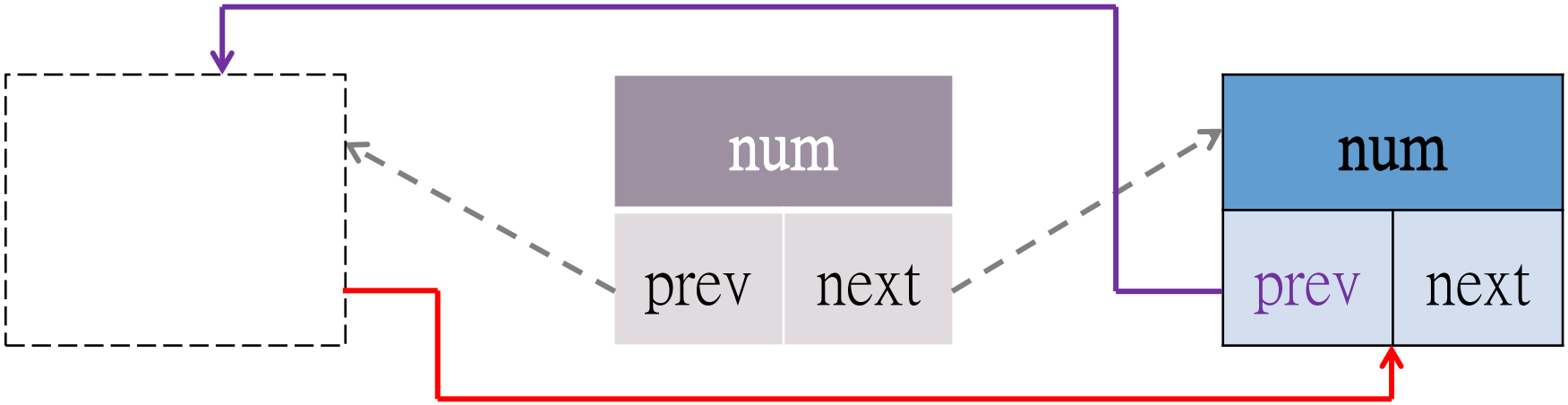
delPrev (Cont.)

- 還有一點小問題



delPrev (Cont.)

Step 1: nodePtr->prev = nodePtr->prev->prev



Step 2: nodePtr->prev->next = nodePtr

這時候就不需要 step 2 了

先來複習一下需要的函數

- newDnode: 新增一個 node
- delPrev: 刪掉前一個 node
- delNext: 刪掉後一個 node
- insertPrev: 插入到目前 node 之前
- insertNext: 插入到目前 node 之後

delNext

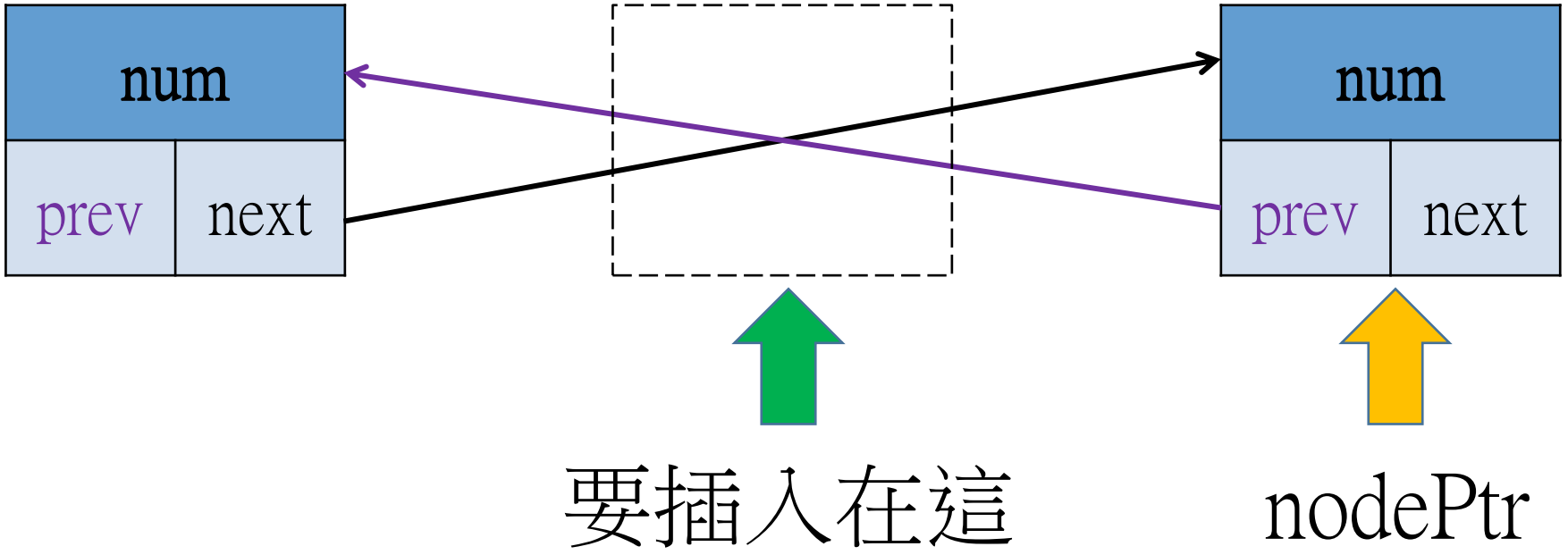
- 基本上和 delPrev 一樣
- 方向反過來就好了

先來複習一下需要的函數

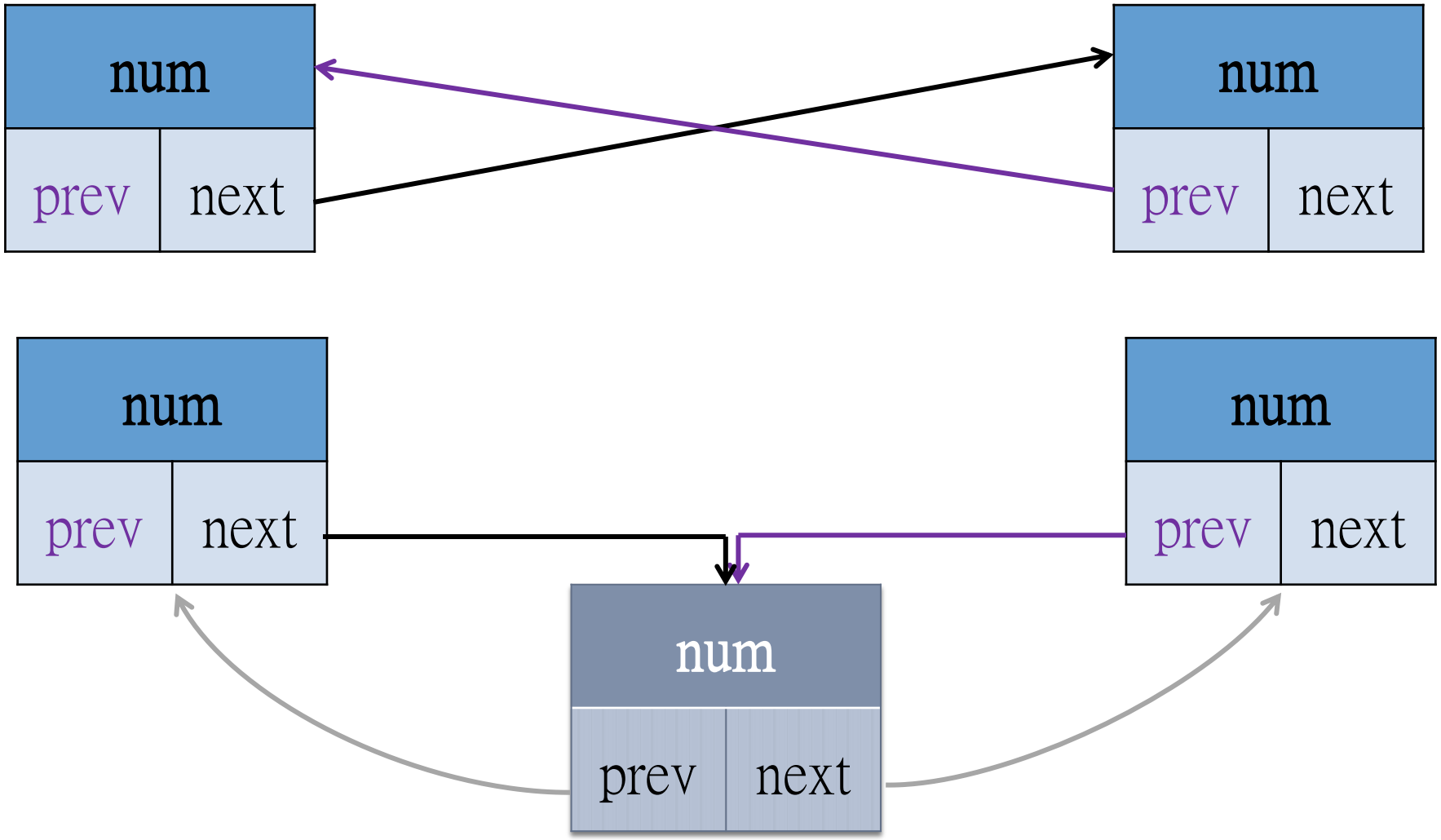
- newDnode: 新增一個 node
- delPrev: 刪掉前一個 node
- delNext: 刪掉後一個 node
- insertPrev: 插入到目前 node 之前
- insertNext: 插入到目前 node 之後

insertPrev

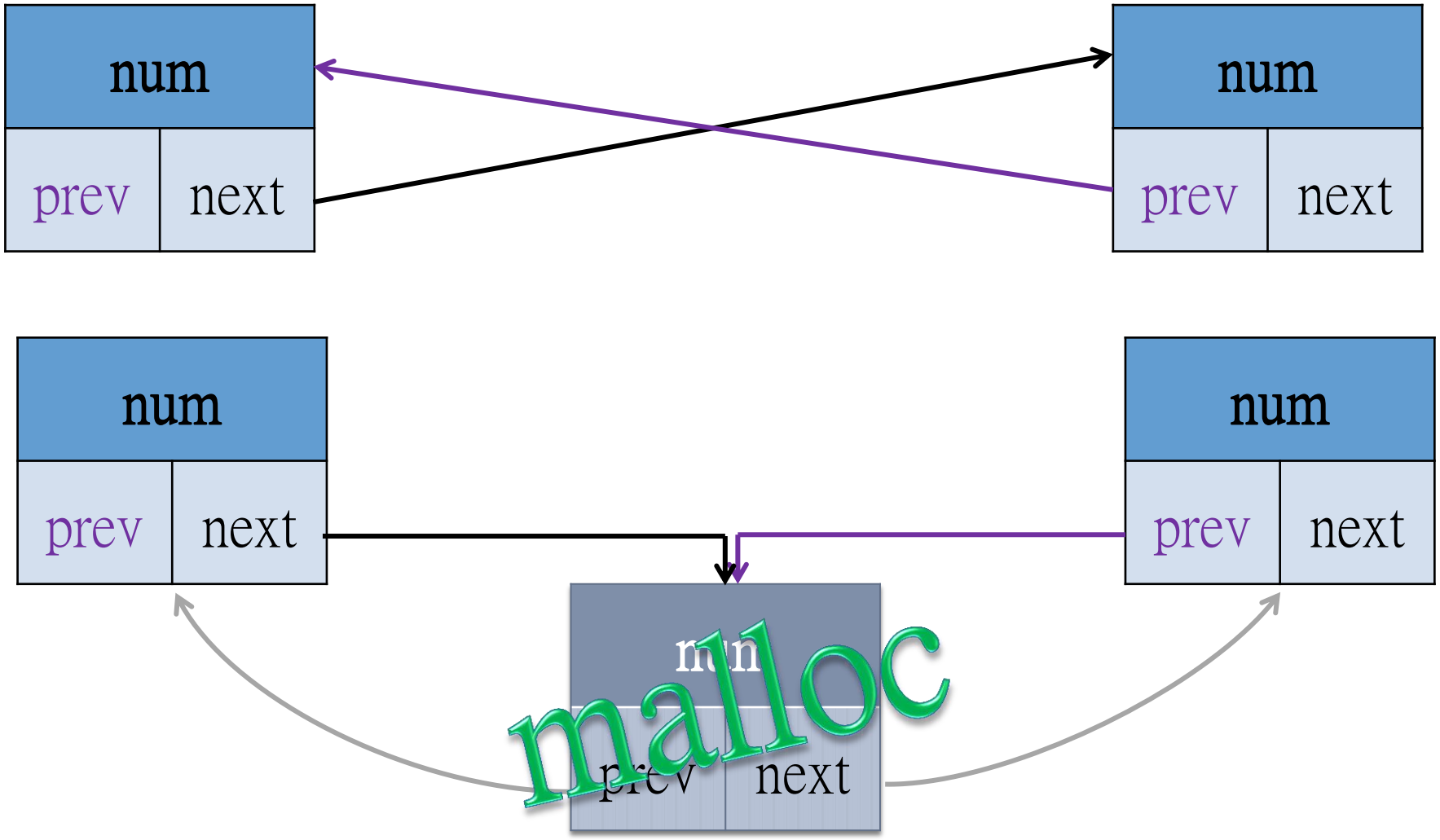
- 傳入要插入的 num, node 指標
- 回傳插入 node 的指標



insertPrev (Cont.)



insertPrev (Cont.)



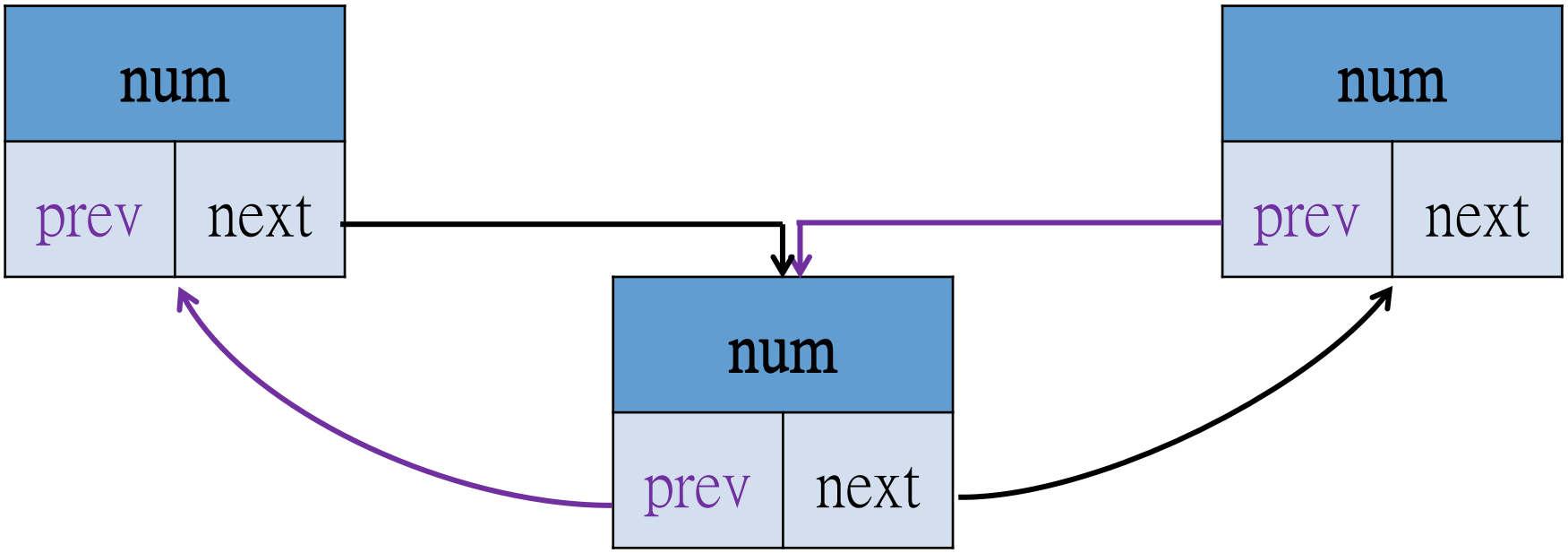
insertPrev (Cont.)

- Step 1: 新增一個 node
  - prev 是 nodePtr->prev
  - next 是 nodePtr
  - newDnode(num, nodePtr->prev, nodePtr)
  - 要把 newDnode 回傳值記起來
    - newPtr = newDnode(…)

insertPrev (Cont.)

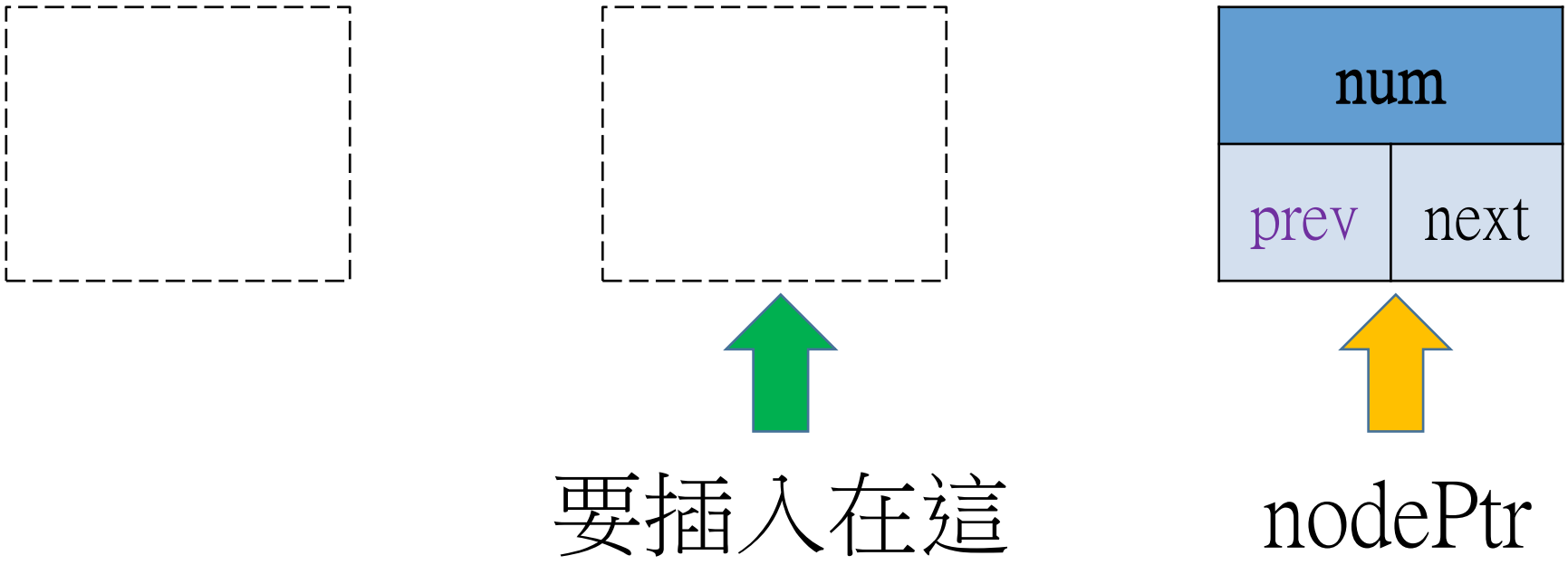
Step 2: nodePtr->prev = newPtr

Step 3: nodePtr->prev->prev->next = newPtr



insertPrev (Cont.)

- 還是有點小問題

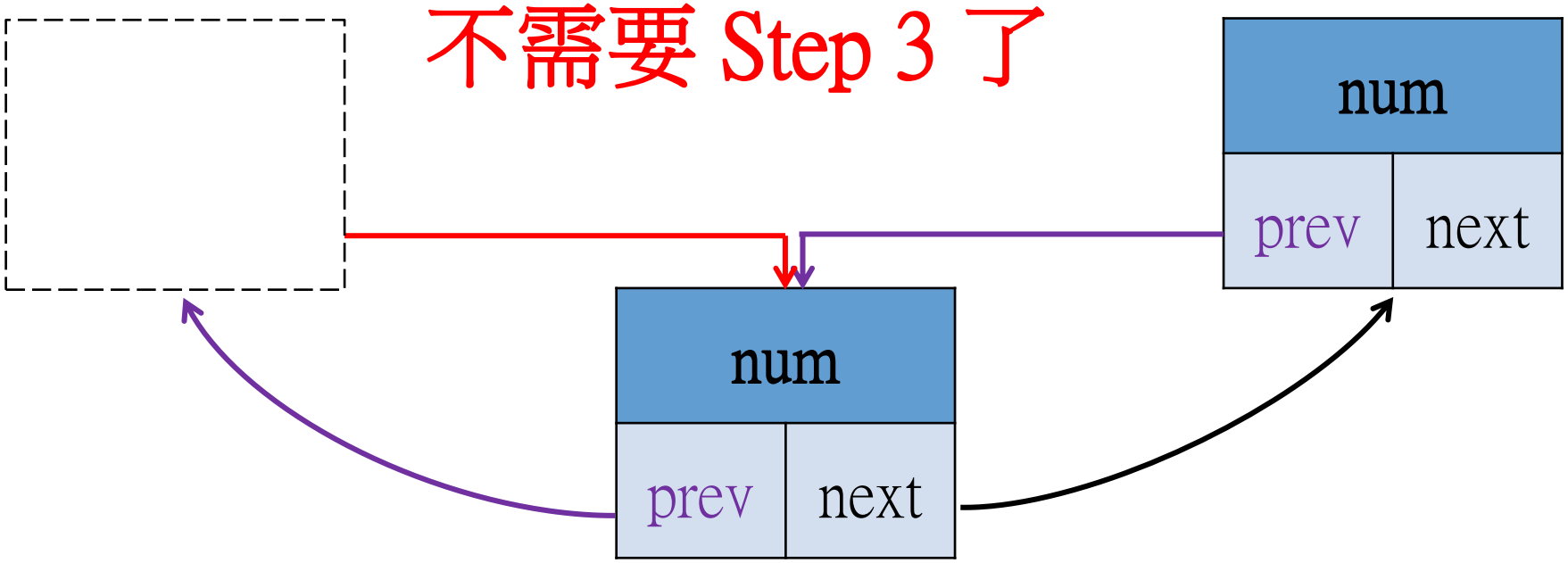


insertPrev (Cont.)

Step 2: nodePtr->prev = newPtr

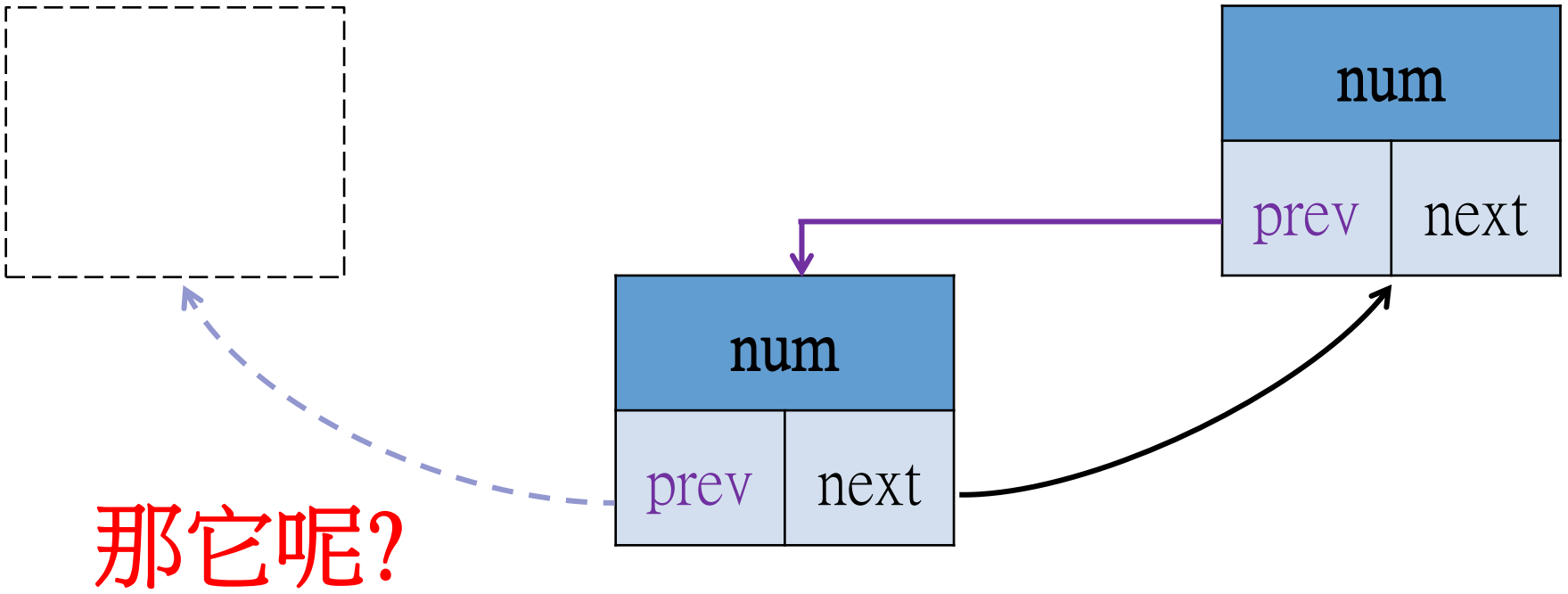
Step 3: nodePtr->prev->prev->next = newPtr

不需要 Step 3 了



insertPrev (Cont.)

Step 2: nodePtr->prev = newPtr



insertPrev (Cont.)

- 其實不需要修改
  - prev 是 nodePtr->prev
  - nodePtr->prev 是 NULL!!
  - newDnode(num, nodePtr->prev, nodePtr)

先來複習一下需要的函數

- newDnode: 新增一個 node
- delPrev: 刪掉前一個 node
- delNext: 刪掉後一個 node
- insertPrev: 插入到目前 node 之前
- insertNext: 插入到目前 node 之後

insertNext

- 基本上和 insertPrev 一樣
- 方向反過來就好了

Bonus

需要的功能

- Bonus 的部分
  - H & T: 指標直接跳到頭或尾
    - 額外記 head 和 tail 兩個指標
    - struct dnode *head, *tail;
    - 或是每次都跑到 NULL
  - RE: 把鏈結串列前後顛倒
    - 把每個 node 的 prev 跟 next 交換

開始實作囉

需要的函數

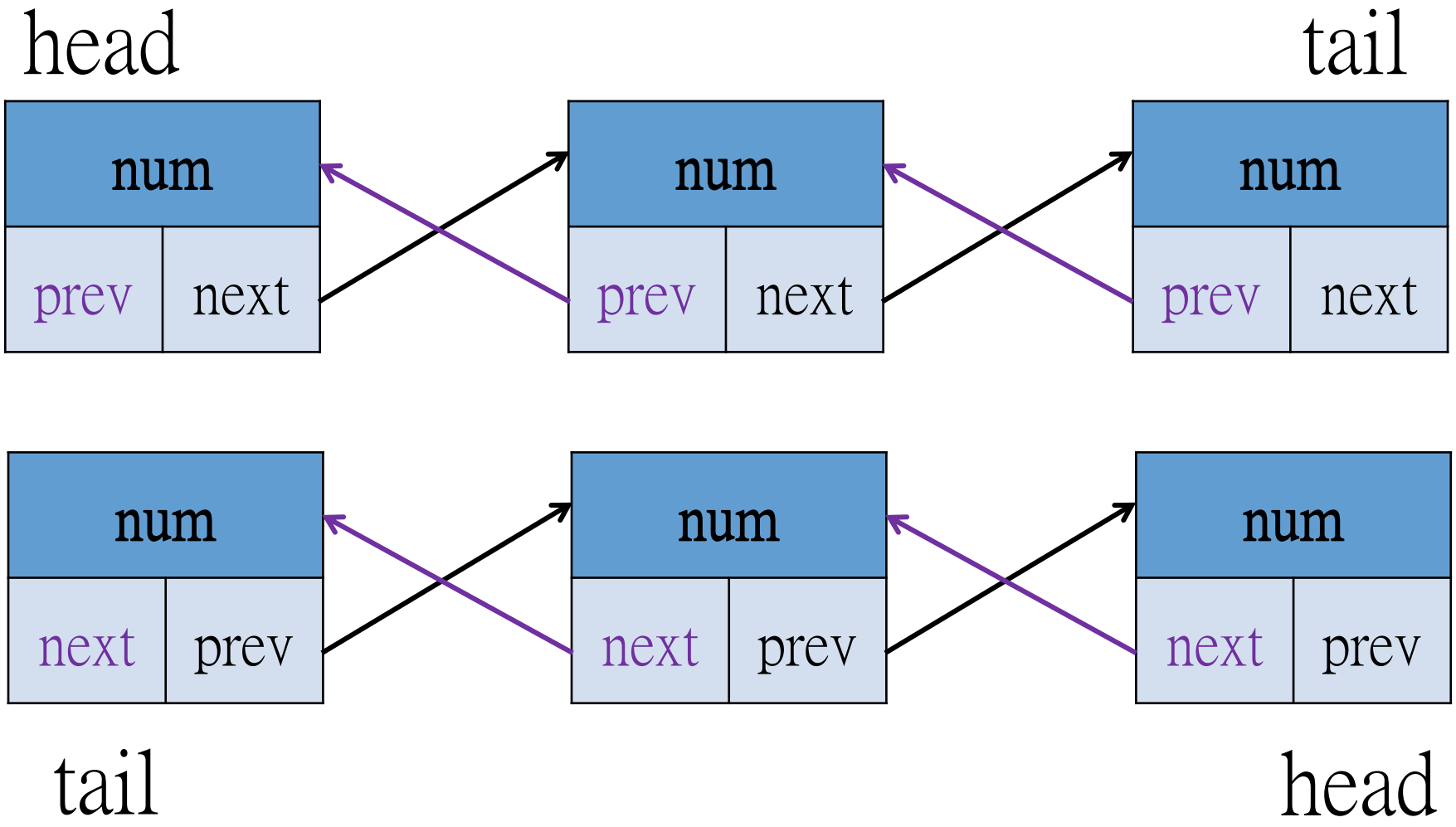
- 新增的
  - reverse: 把每個 node 的 prev 和 next 交換
- 修改的
  - 為了要維護 head 和 tail 的值
    - insertPrev / insertNext
    - delPrev / delNext

需要的函數

- 新增的
  - reverse: 把每個 node 的 prev 和 next 交換

- 修改的
  - 為了要維護 head 和 tail 的值
    - insertPrev / insertNext
    - delPrev / delNext

reverse


head
tail
num
prev
next
num
prev
next
num
prev
next
num
next
prev
num
next
prev
num
next
prev
tail
head

swap

- 先用一個 temp 把其中一個存起來

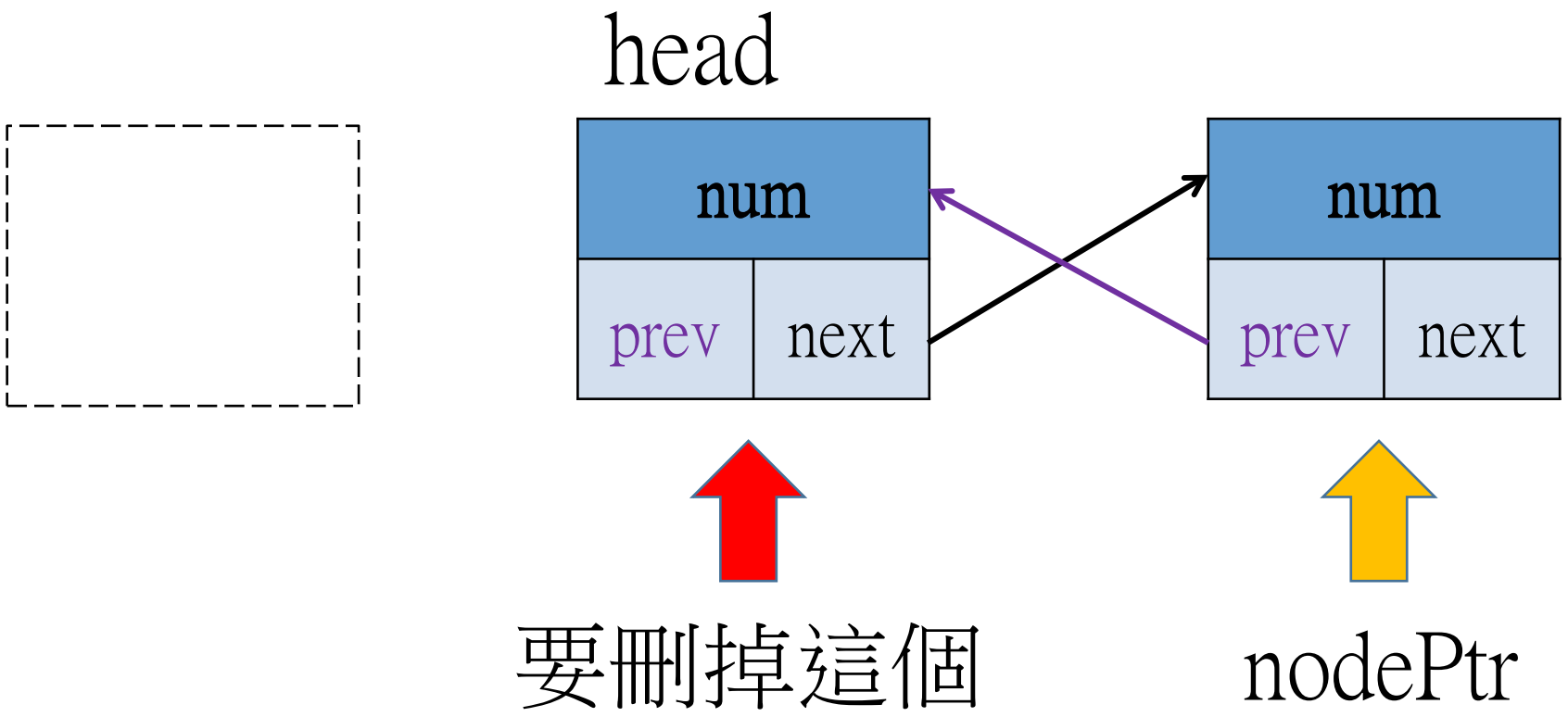
```
struct dnode *temp;
temp = nodePtr->next;
nodePtr->next = nodePtr->prev;
nodePtr->prev = temp;
```

- 交換 prev / next, head / tail 都可以用

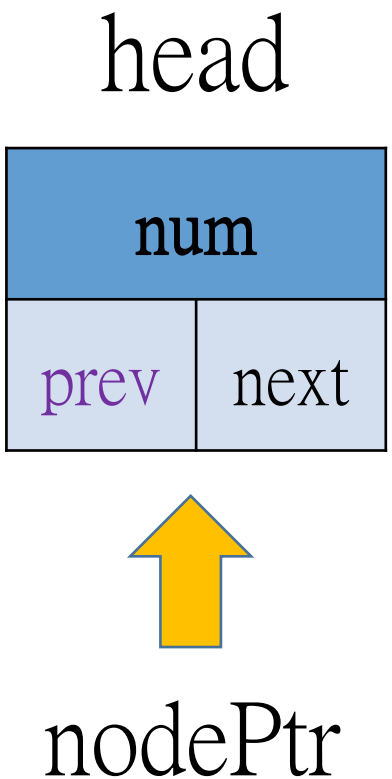
需要的函數

- 新增的
  - reverse: 把每個 node 的 prev 和 next 交換
- 修改的
  - 為了要維護 head 和 tail 的值
    - insertPrev / insertNext
    - delPrev / delNext

delPrev / delNext

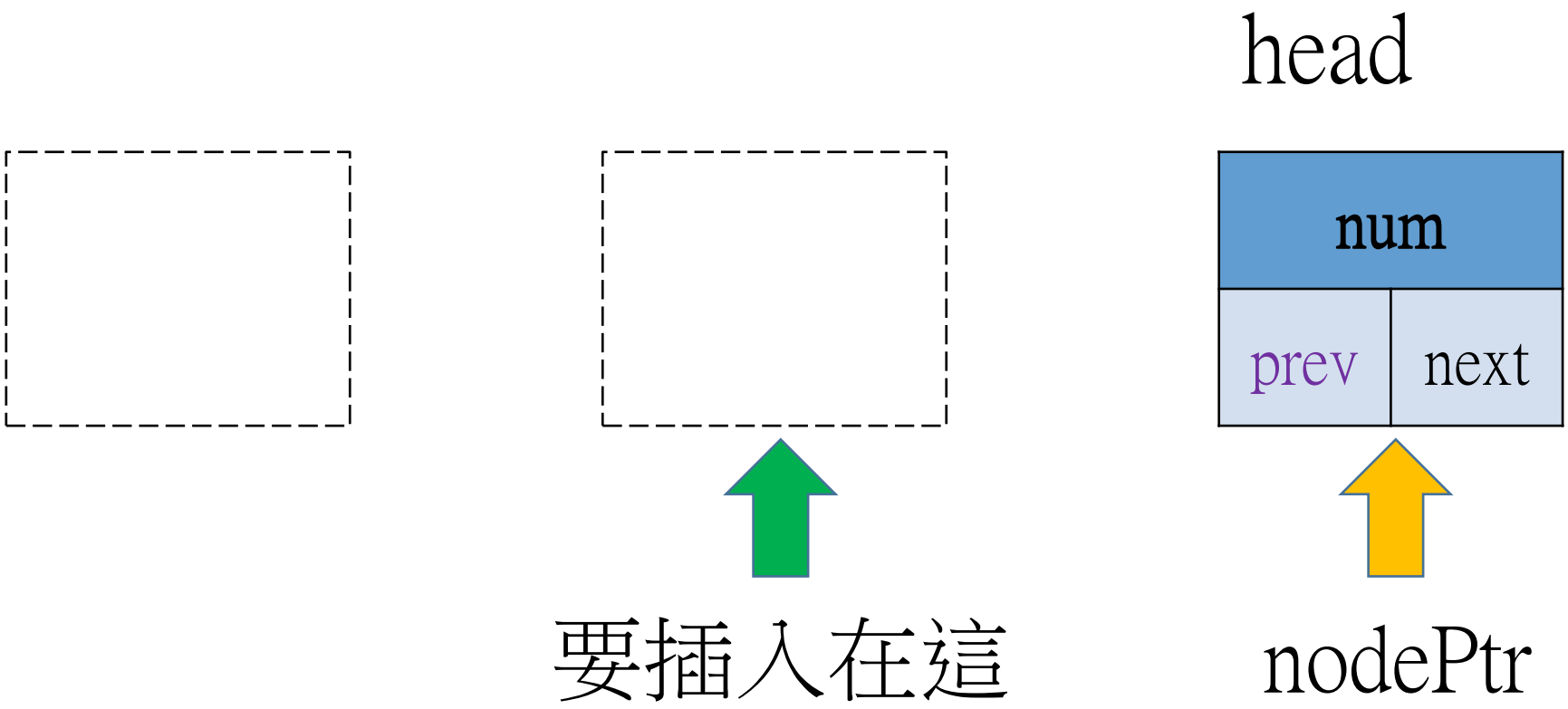


delPrev / delNext (Cont.)



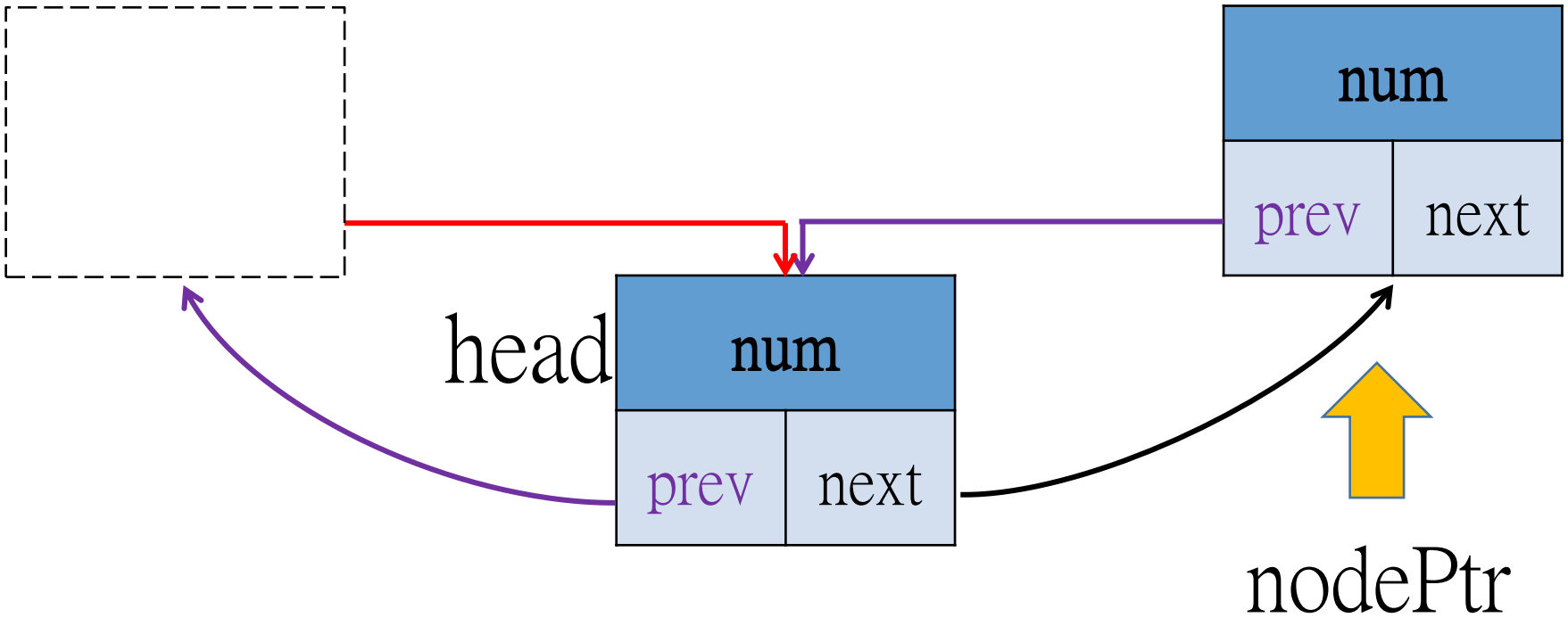
Step 2’ : head = nodePtr

insertPrev / insertNext



insertPrev / insertNext (Cont.)

Step 3’ : head = newPtr



Debug

推薦的功能

- 視覺化的部分
  - printList
  - 把整個鏈結串列從頭到尾印出來
    - HEAD <-> ele 1 <-> ... <-> ele n <-> TAIL
  - 還有從尾到頭
    - TAIL <-> ele n <-> ... <-> ele 1 <-> HEAD
- 方便 debug

Demo

- Demo

推薦的功能 (Cont.)

- 收尾的部分
  - finalize
  - 把還沒 free 掉的空間都 free 掉
  - 不然你的電腦記憶體會越來越少
  - 越來越少…
  - 然後就會跑很慢
  - 其實重開機就會全部還回去了

測資

會把測資寄大家

- sample
- basic
- bonus

- 比較答案和自己的輸出
  - winMerge

One more thing...

星期四記得來上課

- 預定行程
  - 發 test 2
  - 檢討 test 2
  - TEST 3!!!!!
- 請大家告訴大家