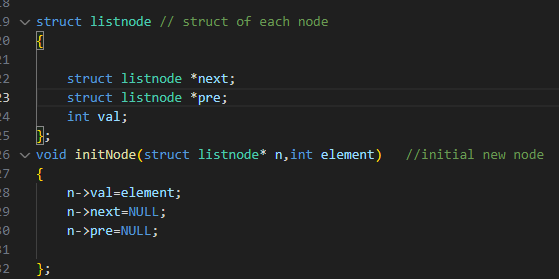
**stored\_dbl\_link\_list document**

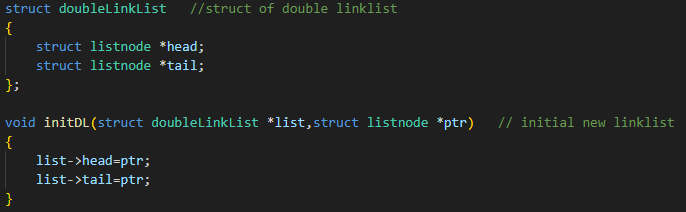
**資料結構設定**

我總共使用2個struct來表示整個資料結構

1.listnode單純用來建立node，每個node有三個成員，分別是pre(前一項)，next(後一項)，val(值)，並使用initNode()來初始化每個node的值。



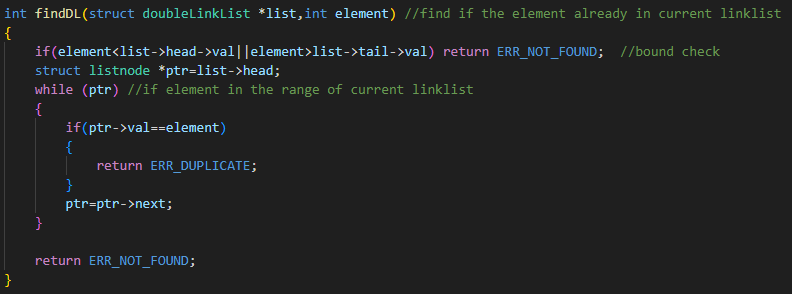
2.第二個struct為double link list的結構，此結構內有兩個指標分別為head以及tail，用意是可以把listnode包進來，並可透過head，tail來存取整個linklist。而初始化是使用initDL()，將整個linklist的頭尾都先初始化。會想這樣包起來的用意是若我今天有多個linklist，我在要選擇讀取哪個linklist時比較好閱讀(ex:list1->head list2->head)

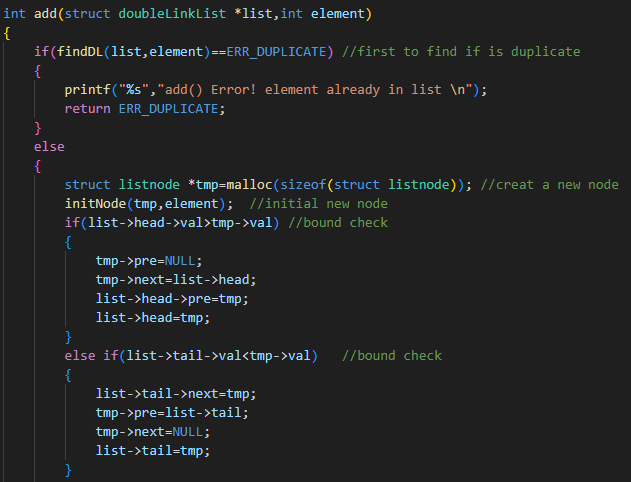


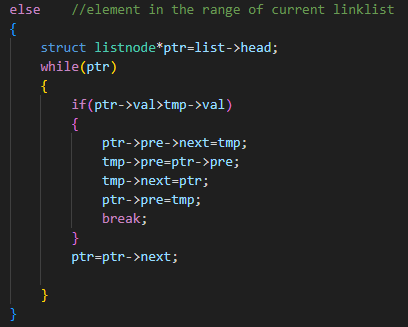
**功能實現**

首先是add()

在add的運算中，會先判斷原有的linklist中是否已經有重複的值。而在判斷重複的值上我使用另一個findDL()函數來確認是否有重複。接著將要加入的資料與linklist頭尾比較判斷要加在哪。

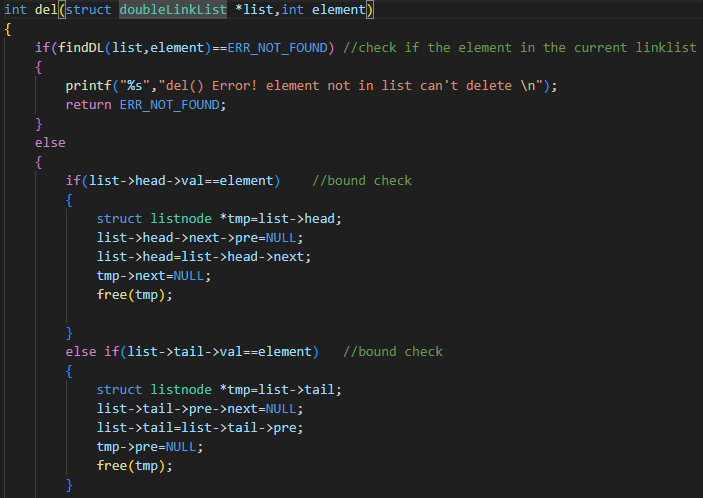


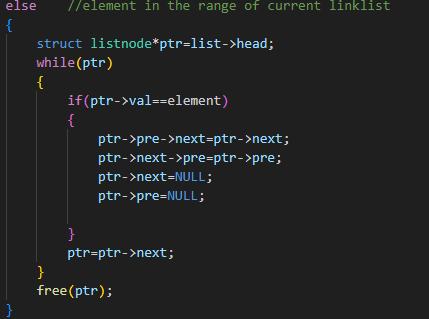




再來是del()

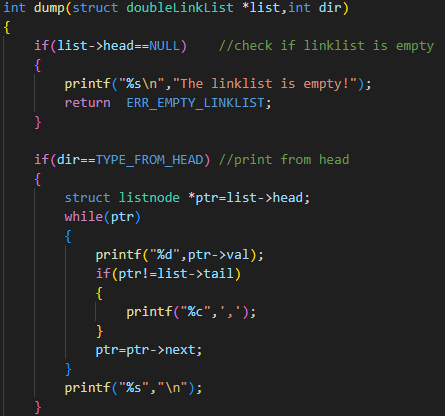
其運作的方式與add()類似，一樣先判斷資料是否存在，接著再判斷要刪除的是否為頭尾或中間，並在delete後free掉空間。

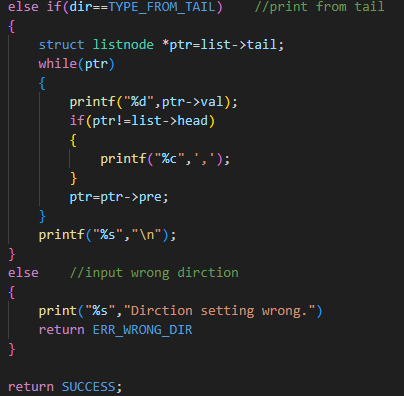




dump()

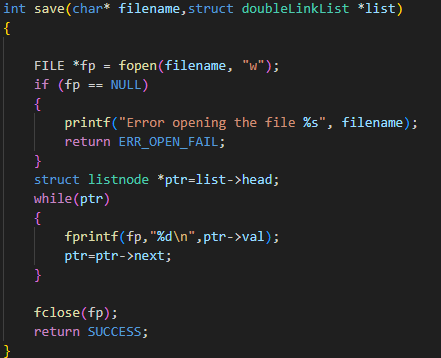
此函數主應用於依照頭到尾或尾到頭來依序輸出資料，一開始我會判定此linllist是否是空的依此回傳。否則就用傳入的方向參數來決定輸出順序。並加入了若今天輸入的方向參數有誤時的錯誤訊息。



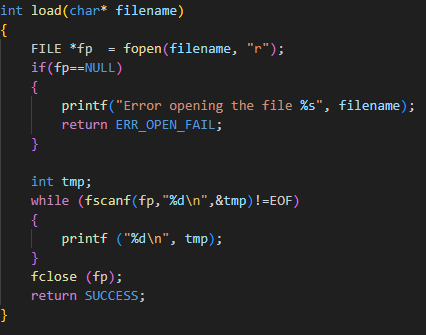


save()以及load()

save()函數用於存檔，透過fp來檢查是否開檔成功，並將檔案儲存

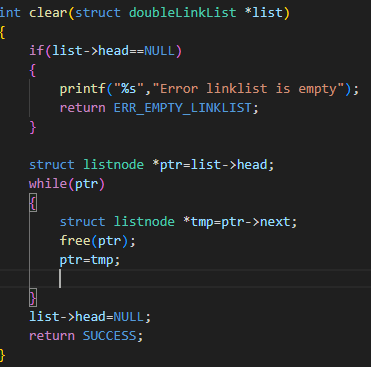


load()函數用於讀檔，透過fp檢查檔案開啟狀態，並把檔案中的資料輸出。



clear()

首先會先判定此linklist是否為空，若為空就回傳錯誤訊息。反之，依序將每個節點free掉，最後把該linlist的head指向null。



main()

主程式主要會先建立新節點並包入新linklist，接著依照題目運行。

