作業三貼文

(一)簡介:

任務零 ： 文字檔轉存二進為檔

確認是否有先前建立的資料庫，如果有便刪掉，先尋找同名的二進位檔，如果不存在同名二進位檔，再尋找同名的文字檔，若同名文字檔仍不存在則輸出錯誤訊息，反之則讀入資料建成資料庫，再以此資料庫為基礎，建立另一個同名二進位檔。存在同名的二進位檔後，便將需要的資料讀入，建立一個雜湊用的資料庫。

任務一 ： 以線性探測建立雜湊表 X

成功找到同名二進位檔後，確立資料量，並找到大於資料量 1.2 倍的質數，作為雜湊表大小與限用函數的除數，並開始逐步放入資料並記錄「搜尋存在值總值」。

在放入資料時，先將 id 內每個數字的 ascii 碼相乘，再除以雜湊表大小，找到其雜湊值，並紀錄到資料格內，「搜尋存在值總值」也加一，之後確認同位置的格子是否已有資料放入，若沒有就直接放入，反之則一個、一個繼續往下找直到找到空格子放入(超過雜湊表大小，則從位置 0，繼續算下去)，在尋找期間「搜尋存在值總值」也跟隨往下找的動作，逐步增加。

雜湊表完成後，便建立指定名稱的雜表文字檔，之後便計算「搜尋不存在值總值」，將每一格抵達下一格空格所需的格數總加起來，便是「搜尋不存在值總值」，最後將「搜尋存在值總值」除以資料數量、「搜尋不存在值總值」除以雜湊表大小，便可輸出「搜尋存在值平均比較次數」與「搜尋不存在值平均比較次數」。

任務二 ： 以雙重雜湊建立雜湊表 Y

同理，在成功找到同名二進位檔後，確立資料量，並找到大於資料量 1.2 倍的質數，作為雜湊表大小與限用函數的除數，再找到大於資料量 1/3 的質數，作為最高步階，接著才開始逐步放入資料並記錄「搜尋存在值總值」。

在放入資料時，先將 id 內每個數字的 ascii 碼相乘，再除以雜湊表大小，找到其雜湊值，並紀錄到資料格內，「搜尋存在值總值」也加一，之後確認同位置的格子是否已有資料放入，若沒有就直接放入，反之則用(最高步階-(id 數字的 ascii 碼乘積/最高步階)算出的答案做為移動格數，每次都依照同樣格數移動，直到找到空格子放入(超過雜湊表大小，則從位置 0，繼續算下去)，在尋找期間「搜尋存在值總值」也跟隨每次往下找的動作，逐步增加。

雜湊表完成後，便建立指定名稱的雜表文字檔，最後將「搜尋存在值總值」除以資料數量，便可輸出「搜尋存在值平均比較次數」。

任務三 ： 以平方探測建立雜湊表

同理，在成功找到同名二進位檔後，確立資料量，並找到大於資料量 1.2 倍的質數，作為雜湊表大小與限用函數的除數，並開始逐步放入資料並記錄「搜尋存在值總值」。

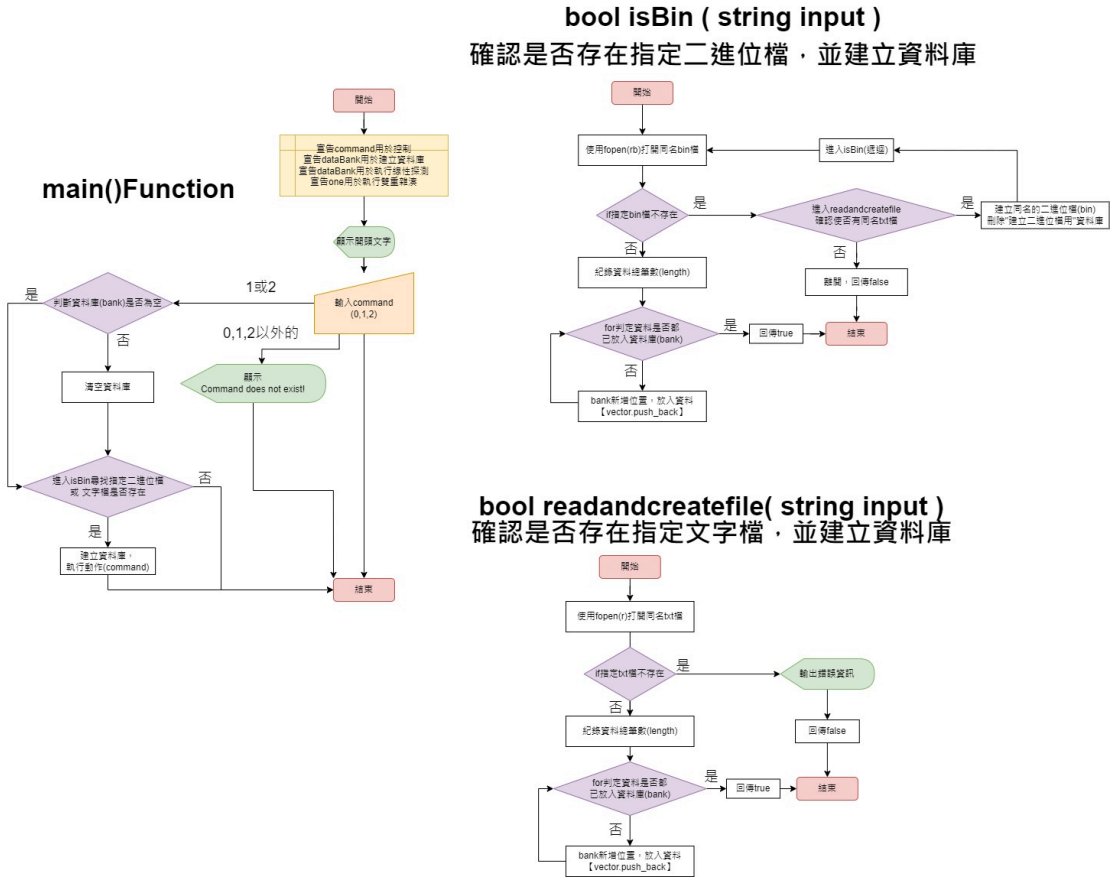
在放入資料時，先將 id 內每個數字的 ascii 碼相乘，再除以雜湊表大小，找到其雜湊值，並紀錄到資料格內，「搜尋存在值總值」也加一，之後確認同位置的格子是否已有資料放入，若沒有就直接放入，反之步數從 1 開始，移動的格數從雜湊值開始算，每次增加步數的平方，遇到有資料的格子，步數再加一並重新開始算步數，直到找到空格放入(超過雜湊表大小，則從位置 0，繼續算下去)，在尋找期間「搜尋存在值總值」也跟隨往下找的動作，逐步增加。

雜湊表完成後，便建立指定名稱的雜表文字檔，之後便計算「搜尋不存在值總值」，將每一格抵達下一格空格所需的格數總加起來，便是「搜尋不存在值總值」，最後將「搜尋存在值總值」除以資料數量、「搜尋不存在值總值」除以雜湊表大小，便可輸出「搜尋存在值平均比較次數」與「搜尋

不存在值平均比較次數」。

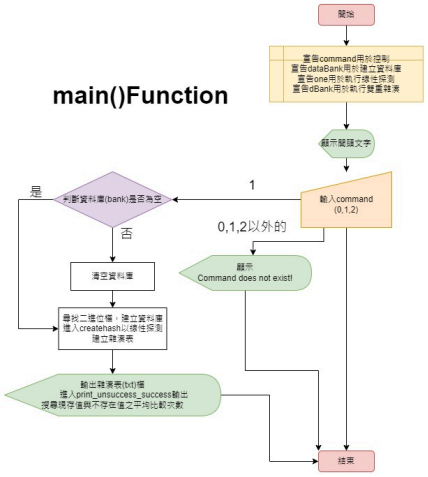
心得:
任務零的部分，在轉成二進位檔和讀二進位檔時，遇到不少困難，因為第一次遇到這種檔案，花了一些時間，習慣它的讀檔方式和寫檔方式。
在處理任務一的時候，我們有發現不同的電腦在遇到一些判定的時候，會出現答案不同的情況，明明程式碼沒變，兩個電腦出現的答案卻不同，之後是修改了判定的條件，才解決了結果在不同電腦間，產生差異的問題。
任務二在最後輸出文字檔時，有發生轉中文字出現錯誤的問題，在某個特定名字後面，多出了轉換錯誤出現的問號，至今仍沒有找到解決方法處理它，同時還發現在另外一台電腦，似乎就不會發生，這個特定名字就不會出現轉換錯誤的問題，合理推測可能也是受電腦系統的影響，造成的結果偏差。

(二)圖示:
任務零:

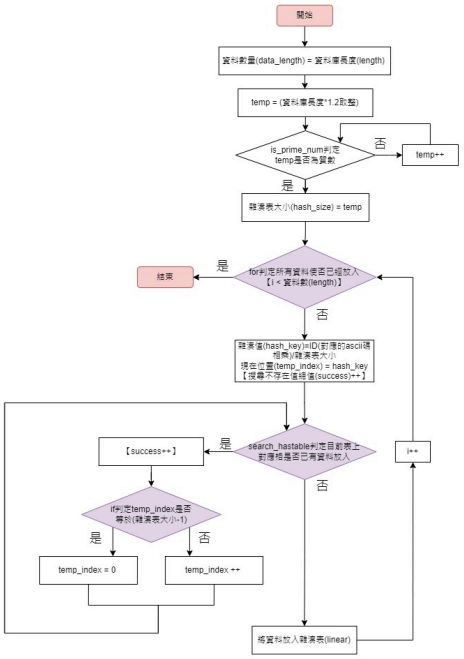


任務一:

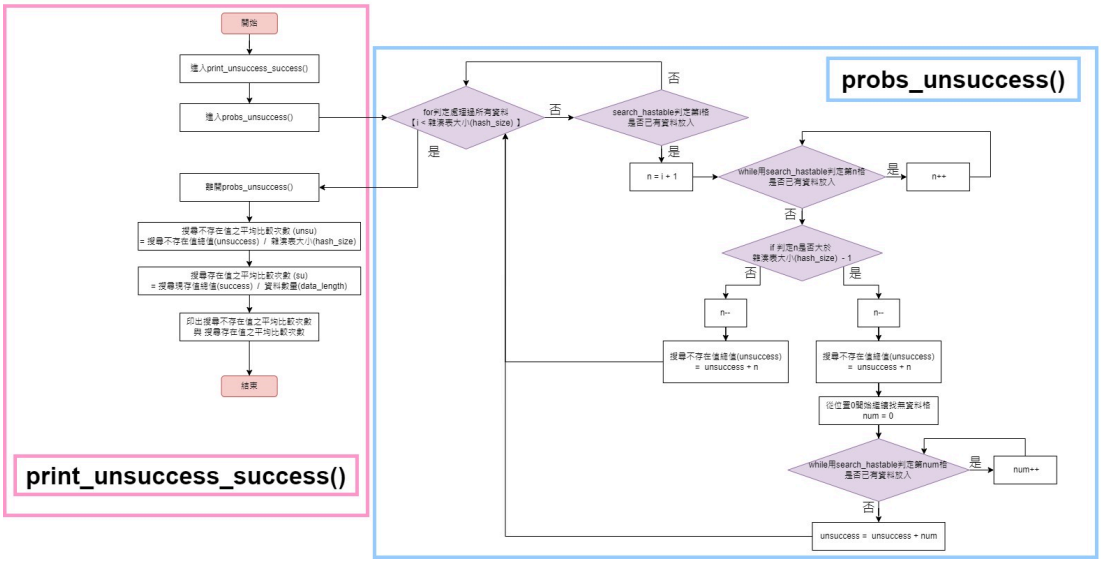
main()Function



void createhash(dataList dataBank)
將資料依照線性探測，逐步放入雜湊表

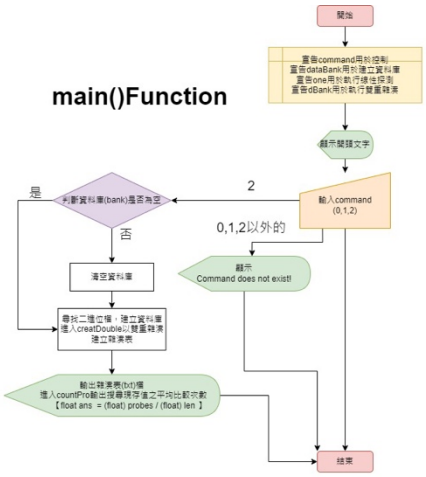


void print_unsuccess_success()
& void probs_unsuccess()
印出搜尋現存值與不存在值之平均比較次數

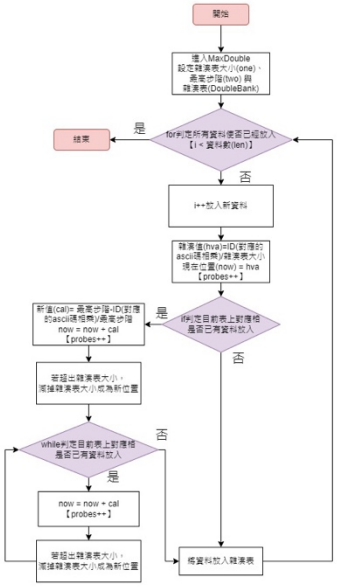


任務二:

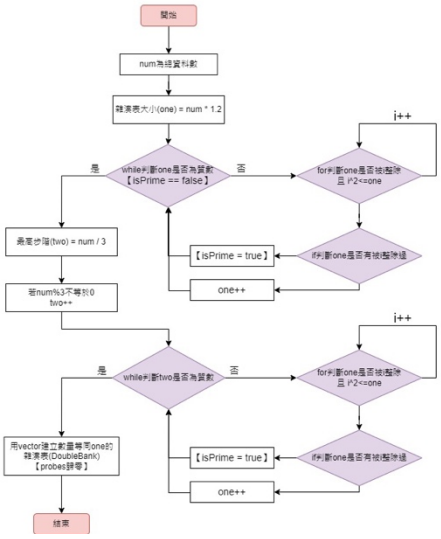
main()Function



void creatDouble(vector<Data> bank, int len)
將資料依照雙重雜湊，逐步放入雜湊表



void MaxDouble(int num)
確定雜湊表大小、算出最高步階、建立雜湊表



(三)答問:

在實驗比較上，雙重雜湊在資料二的搜索平均時間，勝過於平方探測(0.00377 毫秒<0.01887 毫秒)，其他資料因時間過於相近視為平手，在理論比較上，雙重雜湊的平均搜索次數在資料檔 1 與資料檔 2，勝過平方探測(3.23113 次<3.64623 次、3.12453 次<3.48679 次)，就理論上雙重雜湊大多時候都勝過平方探測，可以合理推測平方探測在多數情況下比雙重雜湊更容易發生碰撞，但畢竟是理論計算，並且兩種方法的平均搜索次數本就相近，因此在做平均搜索的實際測試時，仍會受資料的群聚情況影響，而出現搜索時間相近的情況。