F64051114 陳俊宇 資訊工程系 二甲

* program architecture, program function, how I design your program

此次程式設計作業，我把它分成四大架構下去做，大體上全部用treePointer指標。第一是先做切字串的動作，運用insert函數建構出binary search tree。

第二透過del函數，把binary search tree內的數刪除，此函數設計上經過多個if statement來判斷各種情況，大體上分成4種情況：0個兒子、1個左兒子、1個右兒子、2個兒子，而會有一個Search函數做判斷機制，確認binary search tree內有此數，才能做刪除。

第三則是使用levelorder階序走訪，大體上運用環狀柱列，還有變數來控制換行，來把binary search tree印出來。

第四架構最重要，是我DEBUG很久才衍生出來的，它就是clean函數，因為每次做完levelorder階序走訪，我的tree架構就會被我用-1補成full binary search tree，所以levelorder印完樹的樣子之後，需要再把剛剛加上去的-1用cleam函數清除並設置為NULL。

* The time complexity of my function
* Search函數

最差狀況為找到樹的最底部，假設樹高h，time complexity為O(h)

* Insert函數

最差狀況為將新的treenode insert在樹的最底部，假設原本樹高h，time complexity為O(h)

* del函數

第一部分為先找到要刪除的treenode，最差狀況會在樹底部，time complexity為O(h)。第二部分開始做刪除，最差狀況會是有兩個兒子的時候，且刪掉的是樹根，假設刪掉前有n個treenodes，刪除此樹根之後，剩餘n-1個treenodes都要做挪動，所以time complexity為O(n-1)=O(n)

* clean函數

假設原本的treenode跟多了-1的treenode總共有n個，此函數每一treenode都會走訪一次，所以time complexity為O(n)

* treeheight函數

肯定要走到樹的最末端，沒有所謂最差狀況，所以time complexity為O(h)

* queueFull函數

用來控制levelorder函數的環狀柱列結構，沒有time complexity

* addq函數

用來控制levelorder函數的環狀柱列結構，沒有time complexity

* deleteq函數

用來控制levelorder函數的環狀柱列結構，沒有time complexity

* levelorder函數

每一個treenode都會走訪過，且還會自己再長出-1的treenode來使其變成full binary search tree。假設樹高h，此函數就一定會有個treenode要走訪，所以time complexity為O()

* main 函數

先是切字串部分，看到有逗號就要切一次，假設有n個逗號，則while迴圈做n次，time complexity為O(n)。接著開始delete treenode部分，最差狀況是把全部treenodes刪除，因為有n個逗號，所以有n+1個treenodes ，故time complexity為O(n+1)=O(n)。