410921302 凃韻淑

**W12-1 demo**

這題程式主要是要建構一個general tree(普遍樹即每個節點的子節點不一定只有兩個)，利用LinkList()來儲存每個節點的子節點，LinkList類別裡以ListNode型態來儲存當節點資料，而ListNode類別繼承於Node類別，從題目已給的ListNode類別和Node類別可知：每一個ListNode用來儲存前一個以及後一個ListNode類型資料；而每一個節點的資料(此題為數字)以Node類別來做儲存。

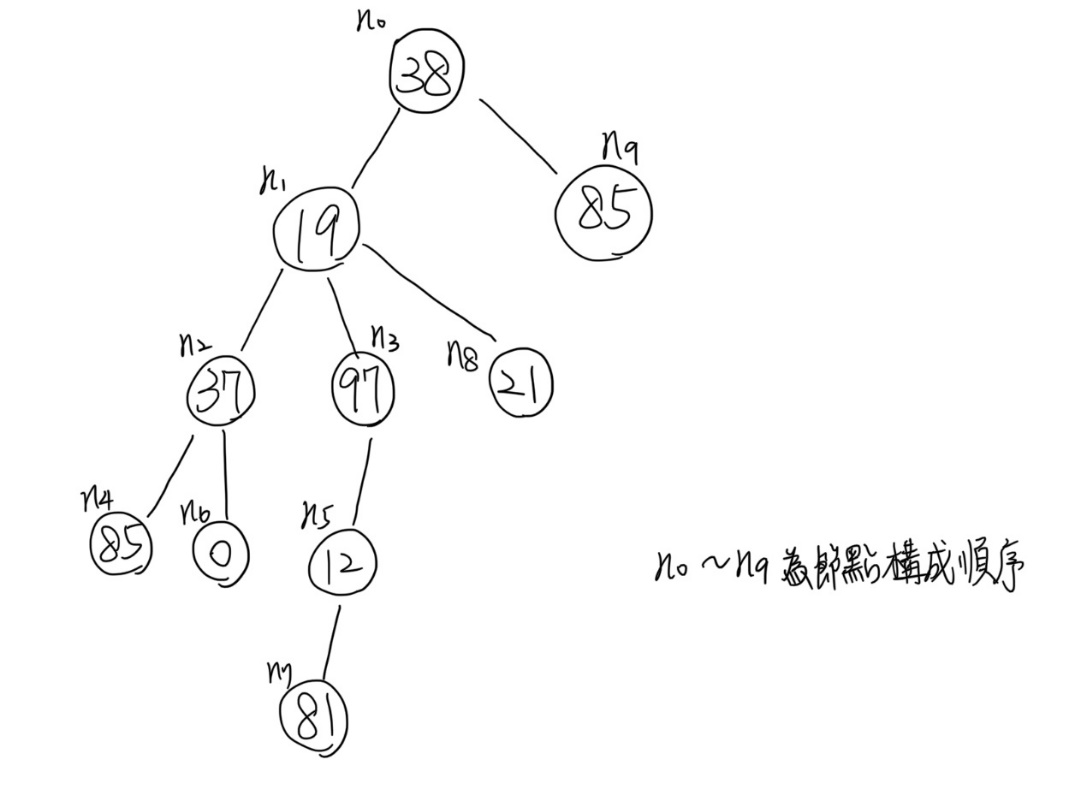
這題我們被要求要完成TreeNode類別以及Tree類別。首先是TreeNode類別，繼承了Node類別，在兩種建構子都呼叫父類別Node來儲存節點資料，且建構子都建立了LinkList型態的child來儲存此節點的子節點。在TreeNode類別中的addchild函數，child呼叫了LinkList類別中的addFromTail函數從linklist後面加入新的TreeNode；在TreeNode類別中進行了運算子[]的多載，這裡的用途是會輸出這個節點的資料(數字)，程式碼(\*child)[n].getData() = child-> operator[](n).getData()。

在Tree類別中，也有一個運算子[]多載的函數，在這裡的作用是將某筆資料(數字)增加到這棵樹，並且會return這個數字的父節點。我利用Queue來儲存運算位置的資料，以for迴圈代表從根部開始找n個節點，while迴圈跑每個節點的子節點，若存在就儲存到Queue，否則會跳出while迴圈(break)，繼續下一個for迴圈。例子：

一張含有 文字 的圖片

自動產生的描述

以node4為例：main程式裡k是在目前的節點數中隨機出一個數字當作進行的步數，n4的k=2，也就是說for迴圈最多會跑2次。Queue在最剛開始已經有root在，因此進入while迴圈後，root有一個子節點n1(此時j=0)，n1被加到Queue中，因為只有一個節點，當j=1的時候，會跳回for迴圈進行第二次迴圈。這次n1在queue的最前面，因此使用n1查看有無子節點，同時也將n1pop出queue，n2為n1子節點，所以n2存入queue，由於已經完成最大步數的for迴圈，此時queue最前面的節點為n2，n4成為n2的子節點。



Count函數也是用queue的方式去儲存目前存在的節點，以一層一層的方式，先探測root的子節點並放入queue，接著用q.size()來計算一層總數，計算完之後加上前面加總過的counts裡，每探測一個節點的子節點前都會把當節點從queue當中去除以免有多計算的問題發生。levelOrder的方式跟count相似，都是以queue存取節點，只是levelOrder是在每次pop的時候都將數字輸出。最後就是postorder跟preorder函數，兩者皆是用遞迴方式運作，postorder是左子樹->右子樹->中子樹，所以將輸出的程式碼放在判斷是否有子節點的while迴圈後，若當節點已是樹葉，那麼將會跳出while輸出當節點，再依序往回做判斷；preorder則是中子樹先，因此輸出的程式碼放於while迴圈前，在判斷前先輸出當節點。setRoot跟getRoot兩個函數的建立一個是為了建立根節點，另一個主要是在main函數裡用來讓tree可以直接呼叫root做preorder跟postorder的遞迴。

一張含有 文字 的圖片

自動產生的描述

**W14-1 DEMO**

題目的要求是要建立一個字典樹，並且能夠檢查輸入的字串是否存在。從題目給予的樣本可知：輸入的英文字母皆為小寫，因此共有26個字母。

首先，我創建了一個TrieNode類別，每個TrieNode都有判斷字串是否全部都已經建立在樹上的isword布林函數，只要字串完整建立到樹上，isword = true。TrieNode型態的childs[26]，以a為基礎(=0)，總共會有26個空間，初始化的狀態下childs[26]這個一維陣列空間將會所有都是NULL。

Trie類別，大致上有兩個重要的函數，一個是insert()，另一個是search()：

insert函數：用作連結字典樹，會從root開始往下儲存，利用(auto &c : s) 讓s string中的字元可以一個字一個字的輸入，if判斷我現在造訪的節點的childs[string字串中的字元 – ‘a’]是否存在，若不存在則建立一個新的TrieNode在字元對應的一維陣列位置上，至於為什麼要減’a’，是因為在a~z字母裡我們以a作為基礎，陣列childs[0]對應的會是字母a，childs[1]會是b，以此類推。當字串全部建立後，現在造訪的TrieNode裡的判斷isword會等於true。

search函數：用來搜尋某個字串是否有在字典樹上，跟建立字典樹一樣，搜尋字典樹也是從root開始往下，可以知道當造訪的子節點中的陣列[某字母對應位置]存在時，那麼這個字母就有建立在字典樹上，接著再將造訪節點往那個節點移動繼續比對下一個節點的字母是否相同，直到搜尋斷裂或是字串中的字元都搜尋過才會結束。例如：字典樹中已建有apple和add兩個單字，我要搜尋adopt是否在字典樹上，於是從root開始，childs[0]存在，所以a字母在樹上；childs[3]也存在，所以d也在樹上；但是childs[14]不存在，所以可以知道adopt這個單字是不存在在字典樹上的。

Main主函數，第一步是要建立一個新的記憶體來儲存一棵新的字典樹，因為題目提到一開始會輸入兩個數字n, m，n是用來建立字典樹的資料數，m則是搜尋在字典樹上是否存在的資料數，以兩個for迴圈分別做建構跟搜尋，當搜尋發現這個字串跟字典樹上的字串完全吻合時，會輸出1；反之，會輸出0。