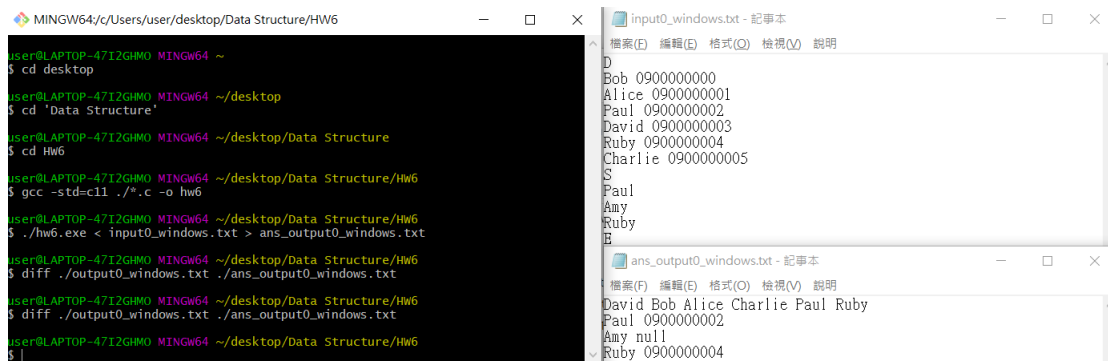


1. Result screenshot



```
MINGW64/c/Users/user/desktop/Data Structure/HW6
user@LAPTOP-47I2GHMO MINGW64 ~
$ cd desktop
user@LAPTOP-47I2GHMO MINGW64 ~/desktop
$ cd 'Data Structure'
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure
$ cd HW6
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure/HW6
$ gcc -std=c11 ./*.c -o hw6
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure/HW6
$ ./hw6.exe < input0_windows.txt > ans_output0_windows.txt
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure/HW6
$ diff ./output0_windows.txt ./ans_output0_windows.txt
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure/HW6
$ diff ./output0_windows.txt ./ans_output0_windows.txt
user@LAPTOP-47I2GHMO MINGW64 ~/desktop/Data Structure/HW6
$
```

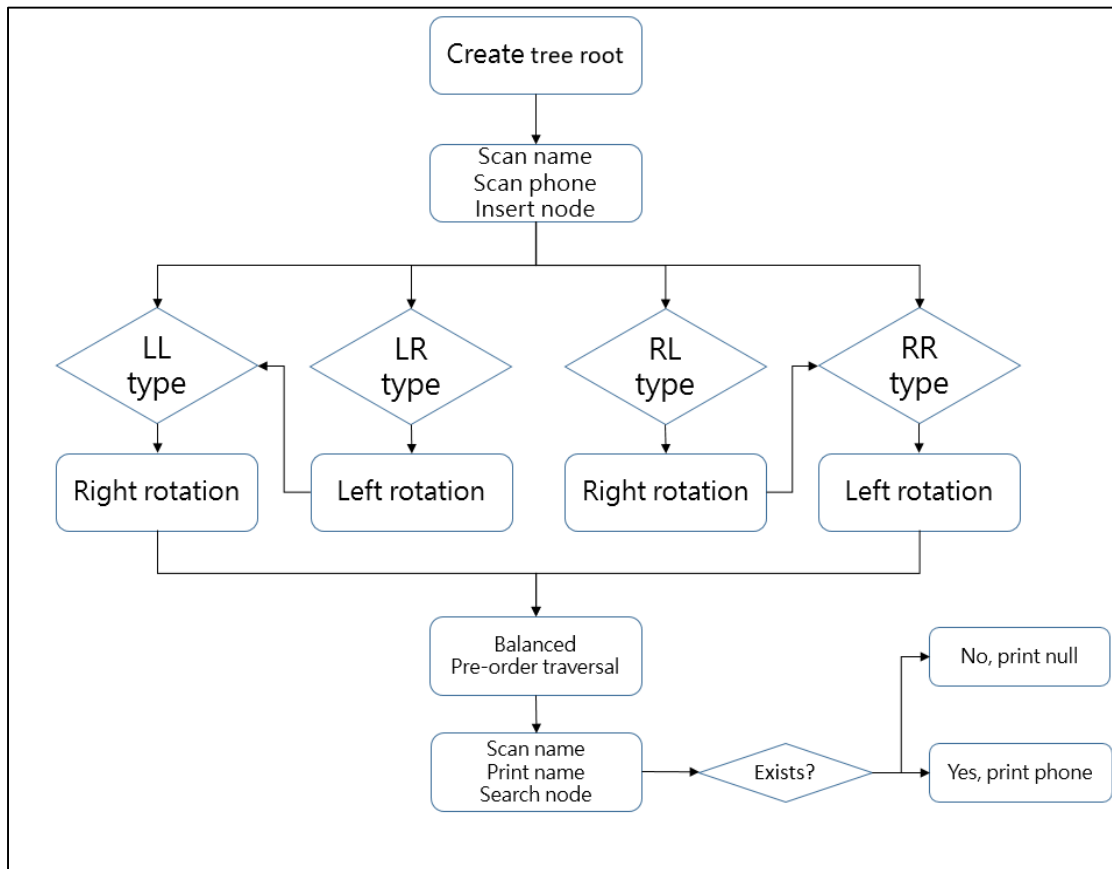
input0_windows.txt - 記事本

```
D
Bob 0900000000
Alice 0900000001
Paul 0900000002
David 0900000003
Ruby 0900000004
Charlie 0900000005
S
Paul
Amy
Ruby
E
```

ans_output0_windows.txt - 記事本

```
David Bob Alice Charlie Paul Ruby
Paul 0900000002
Amy null
Ruby 0900000004
```

2. Program structure



3. Program functions

(1)

int max(int a, int b):用來找出a跟b哪個比較大，這邊用到這個函式主要目的是後面要用來判斷node height的大小

a – data1(其中一個node的height)

b – data2(其中一個node的height)

return – a和b之中比較大的那一個

(2)

int height(tree_pointer node):用來找出node的height

node – 要找height的那一個node

return – node的height

(3)

tree_pointer newNode(char name[], char phone[]):用來建立一個新的node

name – 要放入node->name中的資料(ex:Bob)

phone – 要放入node->phone中的資料(ex:0900000000)

return – 建立好的新node

(4)

tree_pointer search(tree_pointer root, char name[]):依照name給的資料，從root開始往下找node->name = name的資料

root – 要往下搜尋的Tree的root

name – 用來判斷是否搜尋到的是要找的node的資料

return – node->name = name的那一個node，若此Tree中沒有這個node，則回傳NULL

(5)

tree_pointer rightRotate(tree_pointer node_1):讓Tree往右邊做rotate來達到balance

node_1 - 以node_1為子樹的root做right rotate

return – right rotate後成為子樹新root的那個node

(6)

tree_pointer leftRotate(tree_pointer node_2):讓Tree往左邊做rotate來達到balance

node_2 - 以node_2為子樹的root做left rotate

return – left rotate後成為子樹新root的那個node

(7)

int getBF(tree_pointer node):得到node的BF(balance factor)

node – 要取得BF的那個node

return – 用left node height減掉right node height得到的BF

(8)

tree_pointer insert(tree_pointer node, char name[], char phone[]):用來insert一個新的node到Tree中(需考慮各種情況:LL、LR、RR、RL)

node – Tree的root

name – 要insert進來的node的node->name

phone – 要insert進來的node的node->phone

return – Tree的root