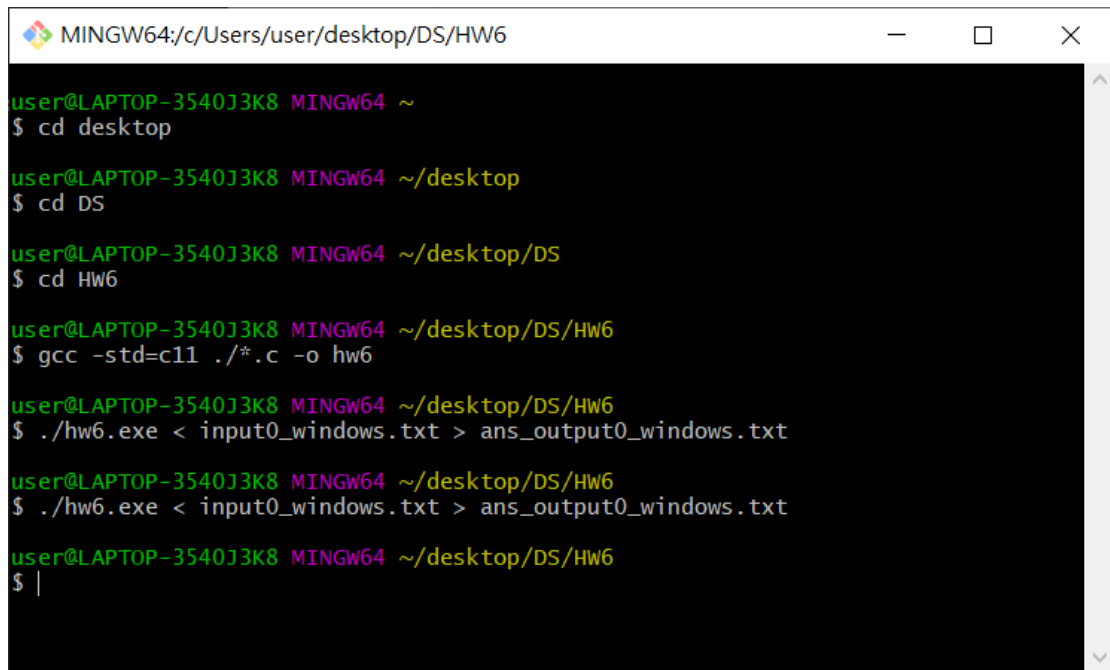
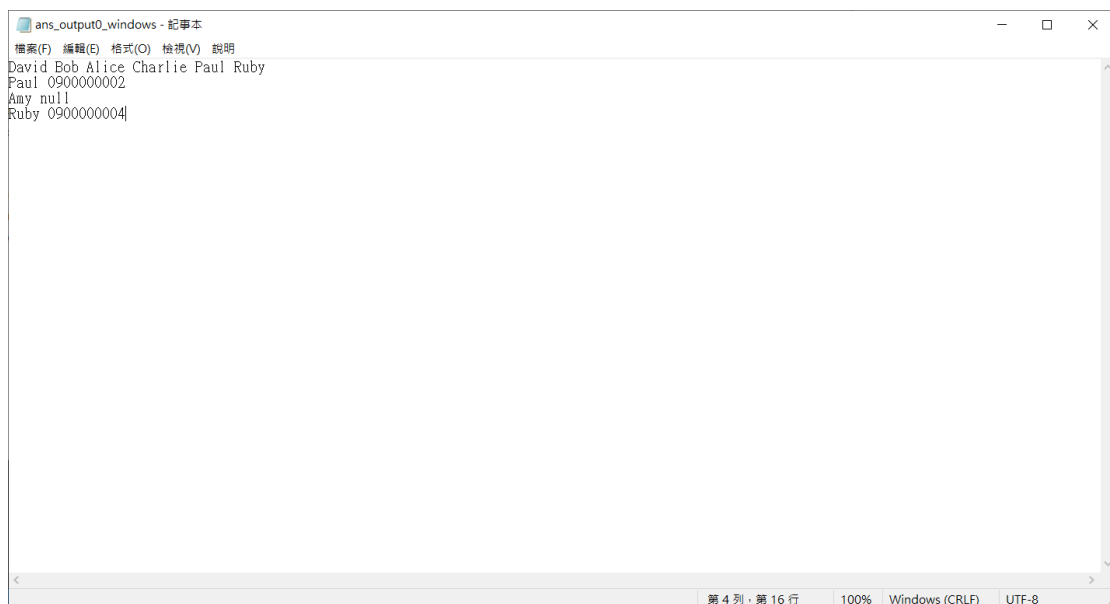


Result Screenshots



```
MINGW64:/c/Users/user/desktop/DS/HW6
user@LAPTOP-3540J3K8 MINGW64 ~
$ cd desktop
user@LAPTOP-3540J3K8 MINGW64 ~/desktop
$ cd DS
user@LAPTOP-3540J3K8 MINGW64 ~/desktop/DS
$ cd HW6
user@LAPTOP-3540J3K8 MINGW64 ~/desktop/DS/HW6
$ gcc -std=c11 ./*.c -o hw6
user@LAPTOP-3540J3K8 MINGW64 ~/desktop/DS/HW6
$ ./hw6.exe < input0_windows.txt > ans_output0_windows.txt
user@LAPTOP-3540J3K8 MINGW64 ~/desktop/DS/HW6
$ ./hw6.exe < input0_windows.txt > ans_output0_windows.txt
user@LAPTOP-3540J3K8 MINGW64 ~/desktop/DS/HW6
$ |
```

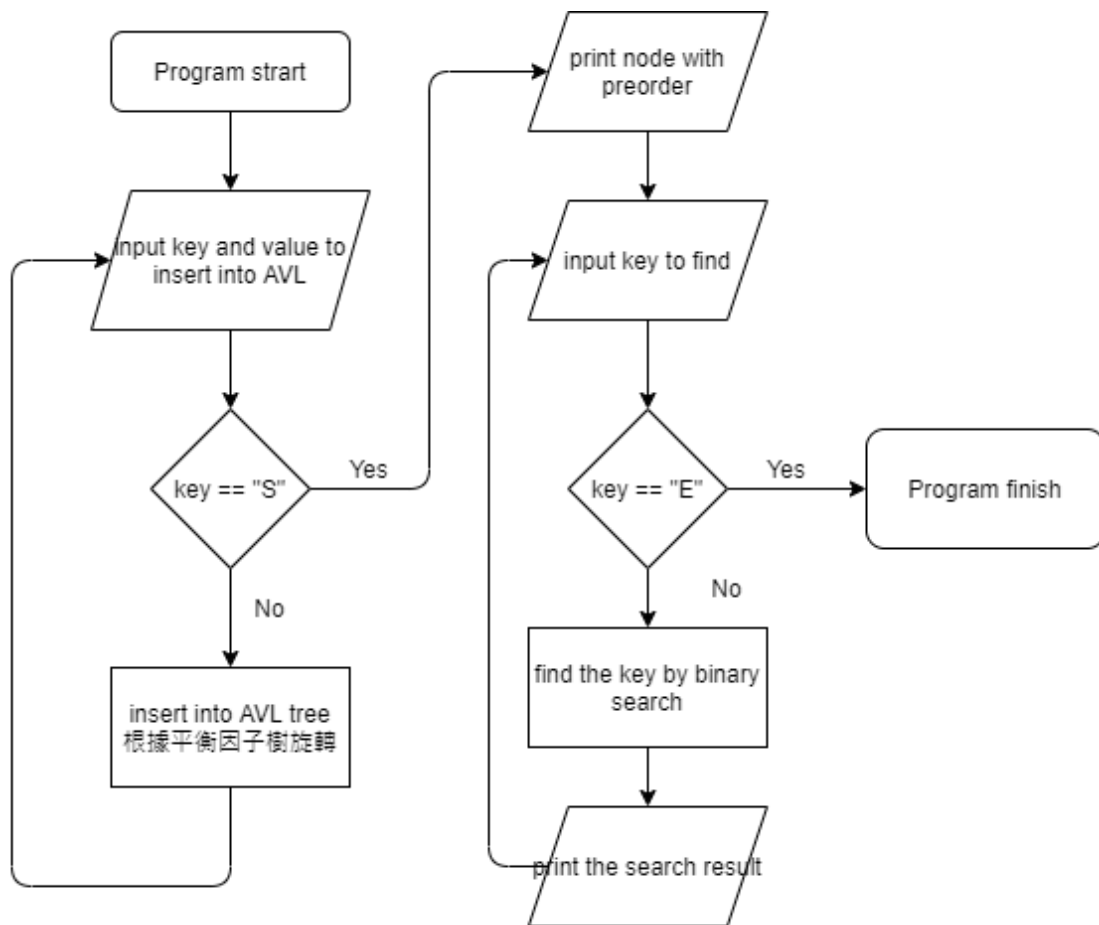
編譯與執行指令截圖



```
ans_output0_windows - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明
David Bob Alice Charlie Paul Ruby
Paul 0900000002
Amy null
Ruby 0900000004|
```

ans_output0_windows.txt 截圖

Program Architecture



Program Functions

```
node* Lrotate(node* root);
```

//對節點進行左旋轉

Parameters:

node* root: 要進行旋轉的節點

Return value:

新的 root 節點

```
node* Rrotate(node* root);
```

//對節點進行右旋轉

Parameters:

node* root: 要進行旋轉的節點

Return value:

新的 root 節點

```
node* insert(node *N, char key[10000], char val[10000]);
```

//將 key 以及 value 插入到 AVL 樹之中

Parameters:

Node* N: 當前走訪到的節點

char key[10000]: 要插入的 key(人名)

char val[10000]: 要插入的 value(電話號碼)

Return value:

當前此節點的新指標

```
node* find(node* N, char key[10000]);
```

//尋找 key 在 AVL 樹之中的位置

Parameters:

node* N: 當前走訪到的節點

char key[10000]: 要在 AVL 之中尋找的 key

Return value:

找到 key 所在的位置, 若找不到則返回 NULL

```
void preorder(node* N, int flag);
```

//以 preorder 的順序遍歷整棵 AVL 樹

Parameters:

node* N: 當前走訪到的節點

int flag: 用來判斷是否要多輸出空格

Return value:

No return value.

Program design

讀入要 insert 到 AVL 樹之中的 key 和 value

根據二分搜，遞迴找到要插入的葉節點位置，並且為它建立新的節

點，依序判斷插入過程之中走過的節點是否有平衡因子 <-2 或者 >2

的情況，若有，則依據 LL,LR,RR,RL 四種不同的情況作相對應的旋轉

讀入要尋找的 key

依照 2 分搜尋尋找 key，若走到葉節點仍然沒找到要查詢的 key 則返

回 NULL.

Operating System

Windows10 家用版

Compiler

gcc version 8.2.0 (MinGW.org GCC-8.2.0-3)

Compile

```
gcc -std=c11 ./*.c -o hw6
```

Run

```
./hw6.exe < input0_windows.txt > ans_output0_windows.txt
```