Report：BST

Yue Zhang

**Code:**

Data structure:

class BST

{

struct node

{

int data;

string words;

node\* left;

node\* right;

};

The operations involved:

1. Insert():

Insert integer data into binary search tree

1. InsertWords():

Insert string data into binary search tree, such words of English txt.

1. findMin():

Finding the smallest node in a binary search tree.

1. Remove():

Delete the specified node.

1. Inorder():

In order traverse binary tree when data is integer.

1. InorderWords():

In order traverse binary tree when data is string.

1. Find():

Find nodes based on its values.

1. Height():

Calculate the height of the binary search tree.

1. Counter():

Calculate the number of nodes in the binary search tree.

**Find out its height h record the min, max and average of h**

When the binary tree has 1000 nodes, the average height is 21, max height is 28, min height is 18.

**Investigate the height of BST of English text**

string out\_word;

ifstream i\_file;

i\_file.open(filename);

if(i\_file.is\_open())

{

while(i\_file >> out\_word)

{

if(wordT.search(out\_word) == NULL)

wordT.insertWords(out\_word);

}

}

wordT.getHeight();

wordT.getWordsNum();

wordT.display\_word();

When in the binary tree, skips word when the same word is found in the binary search tree.

Test text: A part of <<Old man and the sea>>,

Total height is 31

Number of words is 5110.