Assignment 4 – Binary Search Tree

TA: Helen (j15935741@gmail.com)

Deadline: Nov. 27, 11:59pm

Main menu (5%)

In this assignment, you have completed two parts. One is to implement a binary search tree using linked lists, and the other is to implement a treasure hunter game. The main menu for this homework is illustrated as follows.

```
(1)Binary Search Tree.
(2)Treasure Hunter.
(3)Exit.
```

1. Implement a Binary Search Tree with linked lists (40%)

A binary search tree (BST) is a sorted tree structure, where each node has two children. The left node always needs to contain a value smaller than the parent and the right node always needs to contain a value equal or greater than the parent.

In this assignment, you need to implement a BST structure for storing integer numbers and providing the following functions:

- **(5%)Insert (i)** let the user insert a number i; error out when the number is already existing. The result is shown below:

```
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
I
Enter a number : 26
Number 26 is inserted.
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
I
Enter a number : 26
Error. Number 26 exists.
```

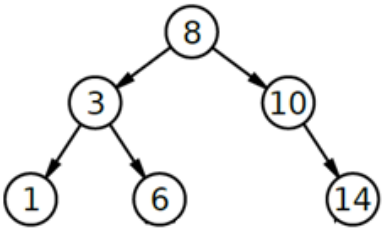
- **(10%)Delete (i)** let the user delete a number i; error out when the number does not exist. If the node with two child nodes is deleted, choose the smallest one on the right subtree to replace the deleted node. The result is shown below:

```
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
d
Enter a number to delete : 50
Number not found.
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
d
Enter a number to delete : 56
Number 56 is deleted.
```

- **(5%)Search (i)** search the tree; message out whether the number i is found or not. The result is shown below:

```
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
S
Enter the element to search : 56
SORRY. Number not found.
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
S
Enter the element to search : 37
BINGO! Number is found.
```

- **(20%)Print Infix&Level Order** print the whole tree in infix order (from left to right) and print the whole tree in level order (from up to down). The result is shown below:

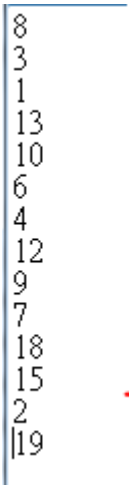



```
(I)nsert a number.
(D)elete a number.
(S)earch a number.
(P)rint Infix&Level order.
(R)eturn
P
The tree in infix order : 1 3 6 8 10 14
The tree in level order : 8 3 10 1 6 14
```

2. Treasure Hunter (50%)

The function of treasure hunter is very similar to Part one. In addition to implementing a BST. First, you need to implement a findKey function to get a key. Finally, you need to implement a findTreasure function to obtain the treasure. The scenario is described as follows:

In this part of assignment, you are given a map file to construct the maze, which is actually a binary search tree. The format of file is as follows.



Insert each number sequentially to construct a binary search tree. The illustration of the maze is shown in Figure 2. After constructing the maze, you need to enable user inputs to specify a key, a treasure and a bomb number. In particular, a bomb number *x* is used to specify all the bombs with a number containing *x.* The entrance of the maze is always from the root of the binary search tree.

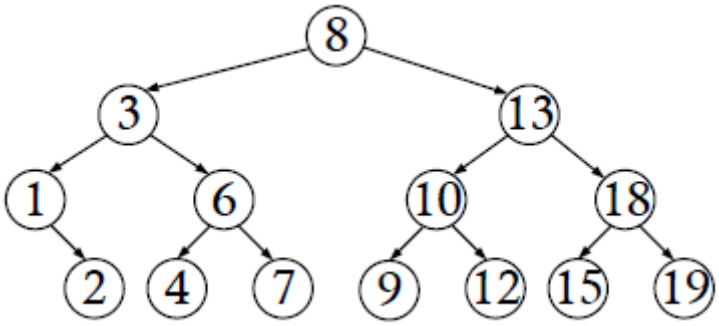


Figure 2: An example of a maze

For example, assume that a user sets the key at node 6 and the treasure at node 19. A bomb number is set to 3; therefore, the bombs are set at node 3 and node 13 in this case. The result is shown in Figure 3.

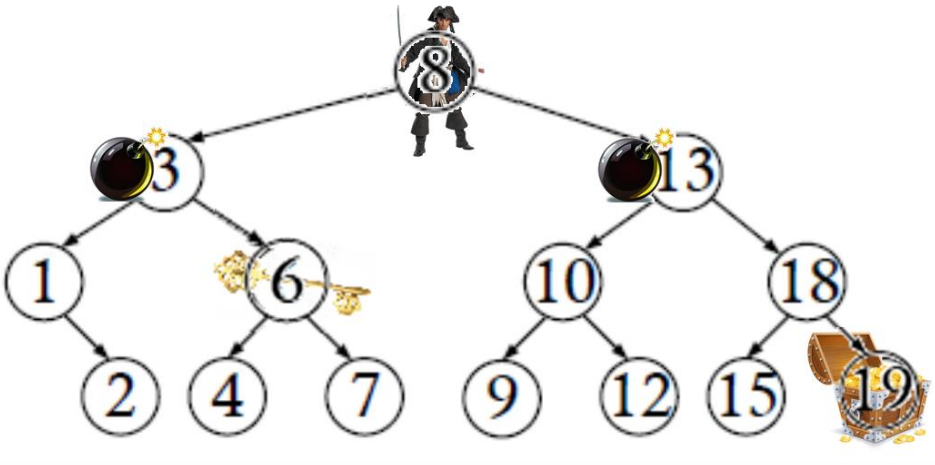


Figure 3: An example of the maze after setting up a key, a treasure, and bombs

When the treasure hunter enters the entrance at node 8, the maze is changed based on the rule. Node 3 and node 13 are deleted, because of the bombs. Afterward, node 4 and node15 are used to replace the deleted nodes (use the same rule in Part one). The result is shown in Figure 4.

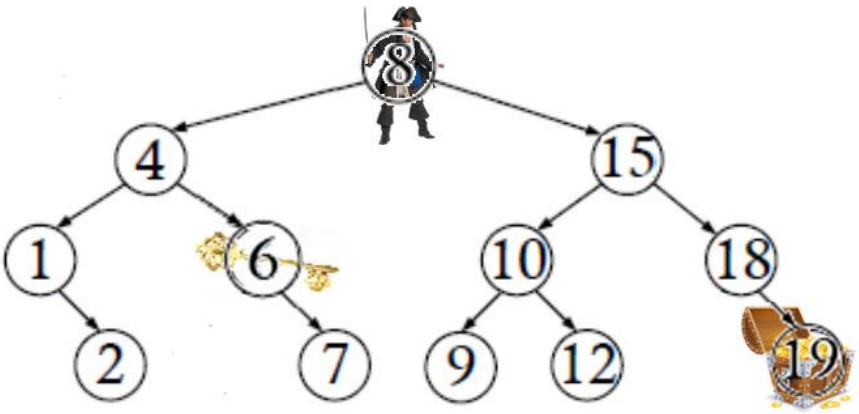


Figure 4: The result of the maze after the adjustment

Next, to find the treasure, the hunter needs to look for the key first. After the key is found, the hunter finds a shortest path from the key to the treasure. The definition of a shortest path here is a path with the minimum number of node visits. Figure 5 shows the shortest path the hunter went through to find the treasure. You need to print out the shortest path to the key and the treasure as the following format: 8->4-> 6->4-> 8->15-> 18->19, and the illustration is shown in Figure 6.

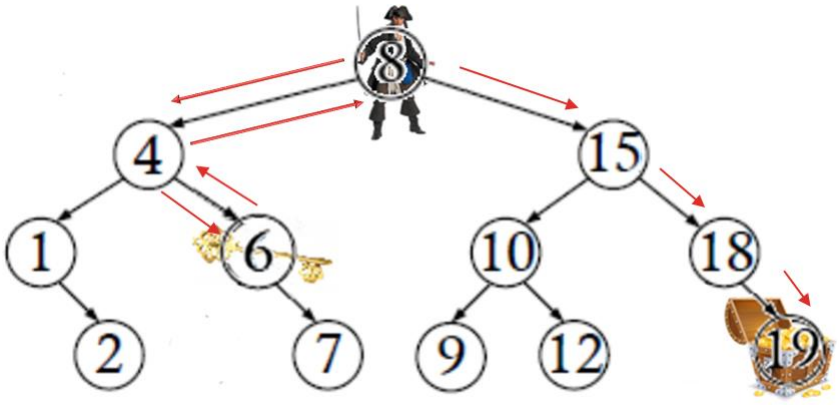


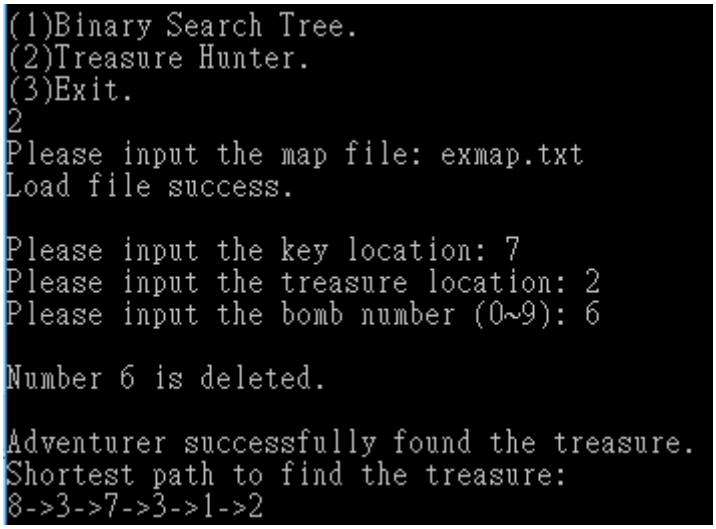
Figure 5: The shortest path to the key and the treasure

```
(1)Binary Search Tree.
(2)Treasure Hunter.
(3)Exit.
2
Please input the map file: exmap.txt
Load file success.

Please input the key location: 6
Please input the treasure location: 19
Please input the bomb number (0~9): 3

Number 3 is deleted.
Number 13 is deleted.

Adventurer successfully found the treasure.
Shortest path to find the treasure:
8->4->6->4->8->15->18->19
```

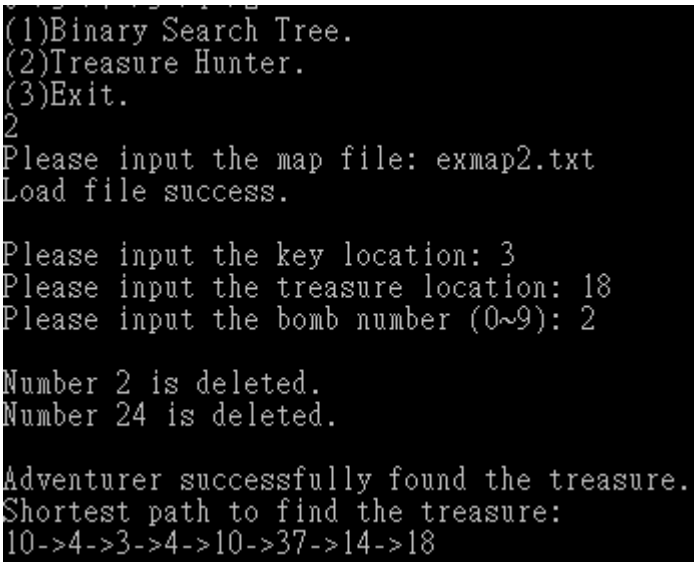
Figure 6: The action menu for the treasure hunter game and the results

3. Test case (Input data will not be the same as the test case when demoing)

(1) Input : exmap.txt



(2) Input : exmap2.txt



4. Output, readme, comments and style (5%)

The TA(s) will mark and give points according to the following policies:

- 5% Main menu
- 40% Part 1 source code can be compiled without any errors and the results are correct.
- 50% Part 2 source code can be compiled without any errors and the results are correct.
  - 5% read data
  - 10% delete bombs
  - 15% find the key path
  - 20% find the shortest path to the treasure
- 5% Readme file, code style, and comments in source code

Readme file should include your name, class ID, a brief description of the code, and other issues students think that will be helpful for the TAs to understand their

homework.

5. How to submit

To submit your files electronically, enter the following command from the csie workstation:

turnin DS_I_2018.hw4 [your files…]

To check the files you turnin, enter the following command from the csie workstation:

turnin –ls DS_I_2018

You can see other description about turnin from following link:
https://www.cs.ccu.edu.tw/lab401/doku.php?id=turninhowto