

Binary Search Tree

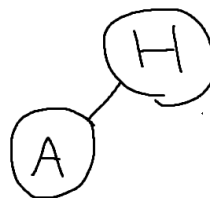
- จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

```
0.   BST tree;
1.   tree.insert('H');
2.   tree.insert('A');
3.   tree.insert('R');
4.   tree.insert('H');
5.   tree.insert('U');
6.   tree.insert('I');
```

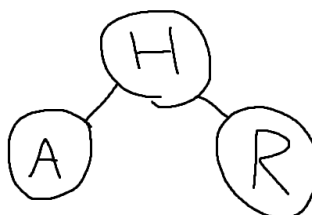
1.



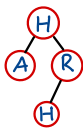
2.



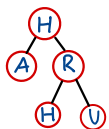
3.



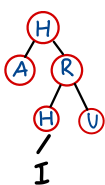
4.



5.



6.



หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น H A R H I V

หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น A H H I R U

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น A I V H H R H

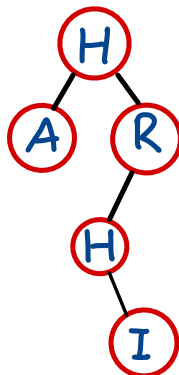
2. ต่อจากข้อ 1 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

```
7.delete_node(&(tree.root->left)); // A  
8.delete_node(&(tree.root->right));  
9.delete_node(&(tree.root->right));
```

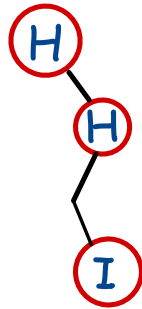
7.



8.



9.



หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น**H H I**.....

หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น **H H I**.....

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น**I H H**.....

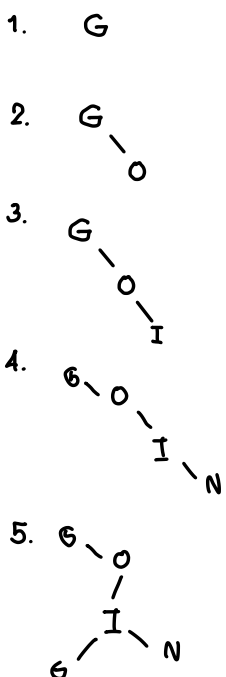
3. จงเขียนแผนภาพการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว (ออกแบบบรรทัดเองเลยครับ)

```
0.   BST tree2;
1.   tree2.insert('G');
2.   tree2.insert('O');
3.   tree2.insert('I');
4.   tree2.insert('N');
5.   tree2.insert('G');
6.   tree2.insert('M');
7.   tree2.insert('E');
8.   tree2.insert('R');
9.   tree2.insert('T');
10.  tree2.insert('Y');
```

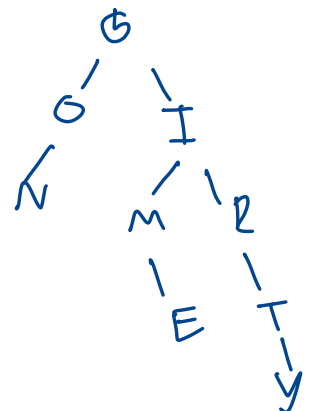
หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น G O I G N M E R T Y

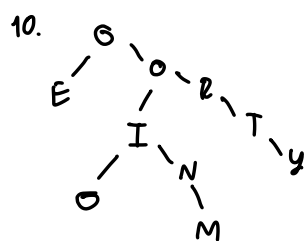
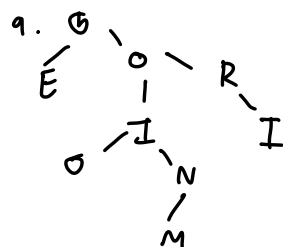
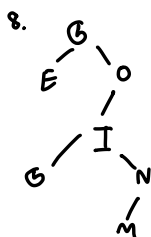
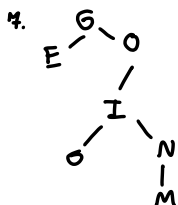
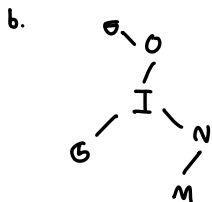
หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น E G G I M N O R T Y

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น E G M N I Y T R O G



G O N I M E R T Y
G I M N O E R T Y
N M E T Y R I O G





หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น GEOIONMAT y

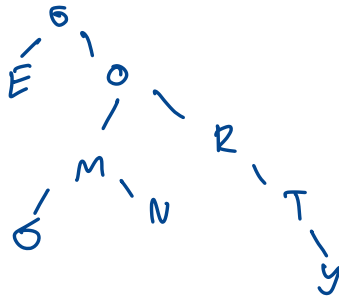
หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น EGGIMNORTy

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น EGMNI YTROG

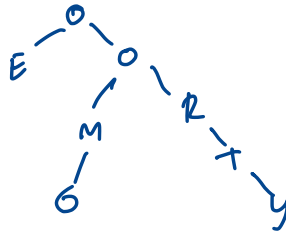
4. ต่อจากข้อ 3 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

```
11. delete_node(&(tree2.root->right->left));  
12. delete_node(&((tree2.root->right->left)->right));  
13. delete_node(&((tree2.root->right->right)->right));  
14. delete_node(&((tree2.root->right->right)->right));
```

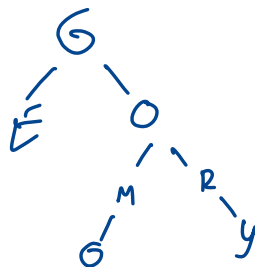
1.



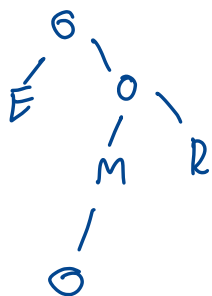
2.



3.



4.



หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น B E O M R O

หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น E B O M R O

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น E O M R B O

5. จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว (ออกแบบบรรทัดเองเลยครับ)

```
1.  BST tree3;  
2.  tree3.insert('A');  
3.  tree3.insert('B');  
4.  tree3.insert('C');  
5.  tree3.insert('D');  
6.  tree3.insert('E');  
7.  tree3.insert('F');  
8.  tree3.insert('G');  
9.  tree3.insert('H');
```

หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น **A B C D E F G H**

หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น **A B C D E F G H**

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น **H G F E D C B A**

1. A

2. A-B

3. A-B-C

4. A-B-C-D

5. A-B-C-D

6. A-B-C-D-E

7 . A-B-C-D-E-F

8 . A-B-C-D-E-F-G

9 . A-B-C-D-E-F-G-H

6. ต่อจากข้อ 3 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

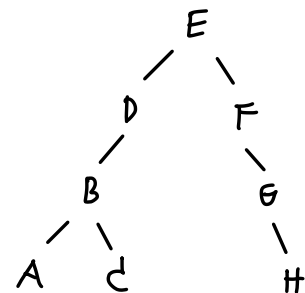
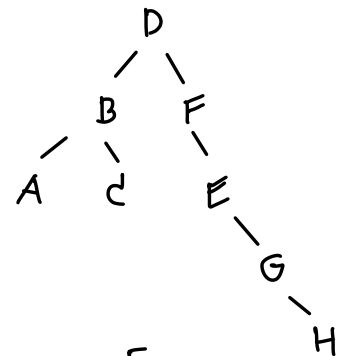
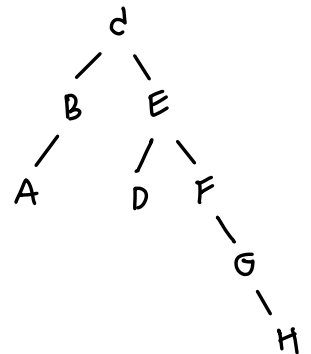
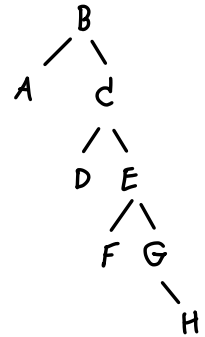
```
10. delete_node(&(tree3.root));  
11. delete_node(&(tree3.root));  
12. delete_node(&(tree3.root));  
13. delete_node(&(tree3.root));
```

10. B - C - D - E - F - G - H

12. C - D - E - F - G - H

13. D - E - F - G - H

14. E - F - G - H



หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น E F G H

หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น E F G H

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น H G F E

7. BST ที่ balance กับ BST ที่ไม่ balance แบบไหนมีลำดับชั้นที่มากกว่ากัน หากจำนวนสมาชิกเท่ากัน เนื่องจากอะไร (ขอสั้นๆ)

BST ที่ไม่ balance จะมีลำดับชั้นมากกว่า เนื่องจากถ้ามี node ที่มี Left subtree, Right subtree ทั้งสอง
node นั้นๆ ทำให้เกิดสาขาไม่สมดุล

8. BST ที่ balance กับ BST ที่ไม่ balance หากต้องการ search แบบไหน ให้เวลาในการค้นหาน้อยกว่ากัน
อย่างไร (ขอสั้นๆ)

ควรใช้ BST แบบ balance เพราะจะมีการกระจายทั่วทุกความสูง

9. Tree ที่ balance กับ tree ที่ไม่ balance แบบใดโดยทั่วไปจะมีประสิทธิภาพดีกว่ากัน (ขอ1 คำ)

แบบ Tree balance

10. ดังนั้นการคิด algorithm และ data structure เราควรพยายามให้ tree อยู่ในรูปของ balance หรือ
unbalance เนื่องจากอะไร (ขอยาวๆ)

ควรให้อยู่ในรูปของ balance เนื่องจากใช้เวลาที่สั้นกว่า / เวลาแทรกและลบข้อมูลจะสั้นกว่า
/ ประสิทธิภาพที่เหมือนกัน