

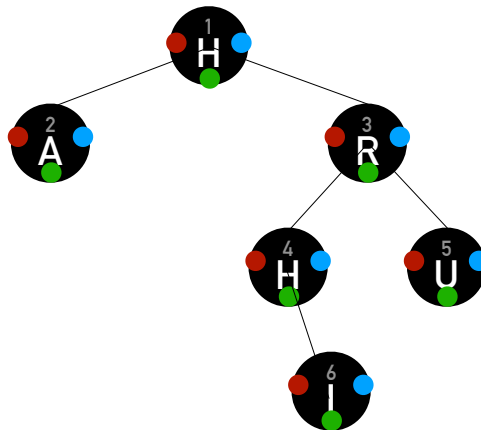
# Binary Search Tree

1. จงเขียนแผนภาพการทำงานของ 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');

```



หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น ..... HARHIU .....

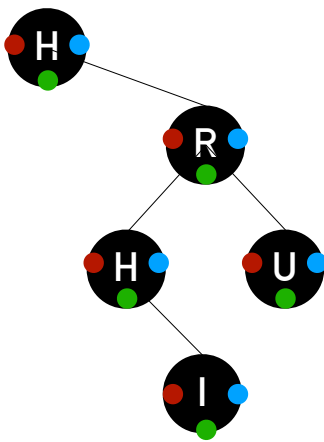
หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น ..... AHHIRU ..... AHHIRU

หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น ..... HAIURH .....

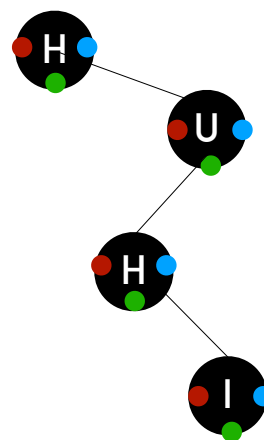
2. ต่อจากข้อ 1 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

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

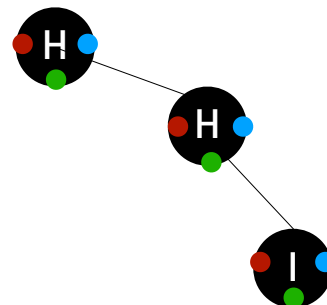
7



8



9



H H I

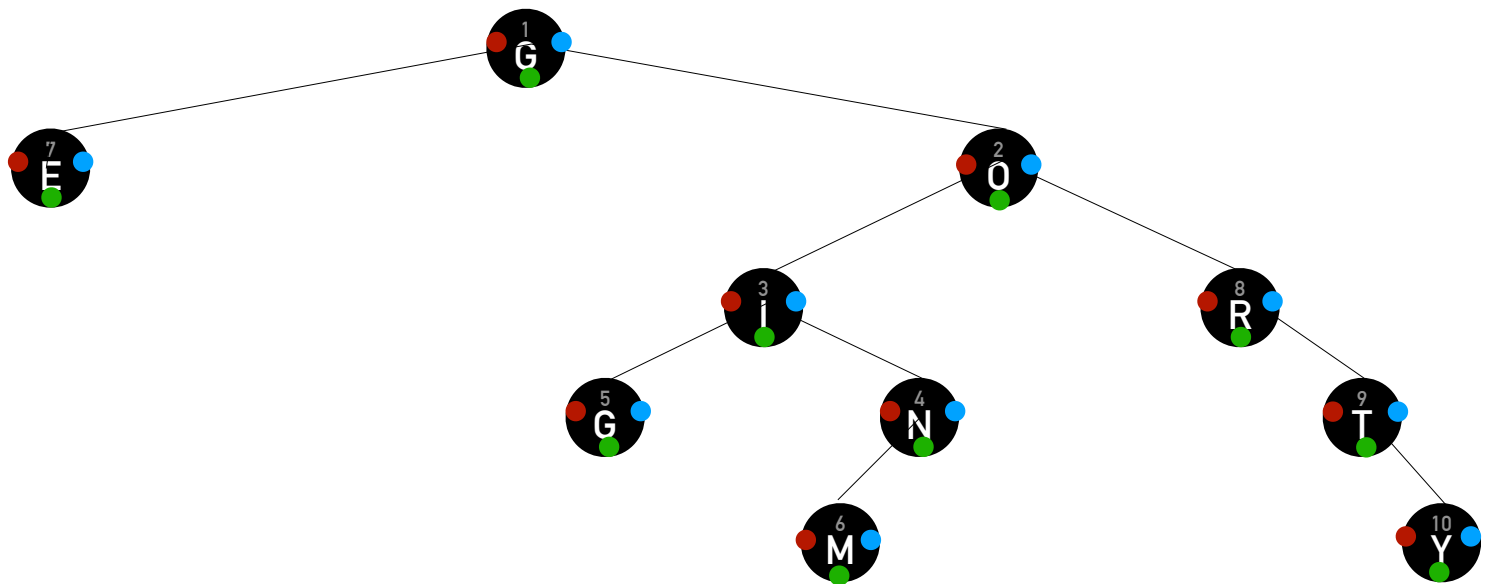
หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น .....

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. จงเขียนแผนภาพของการทำงานของ 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');
```



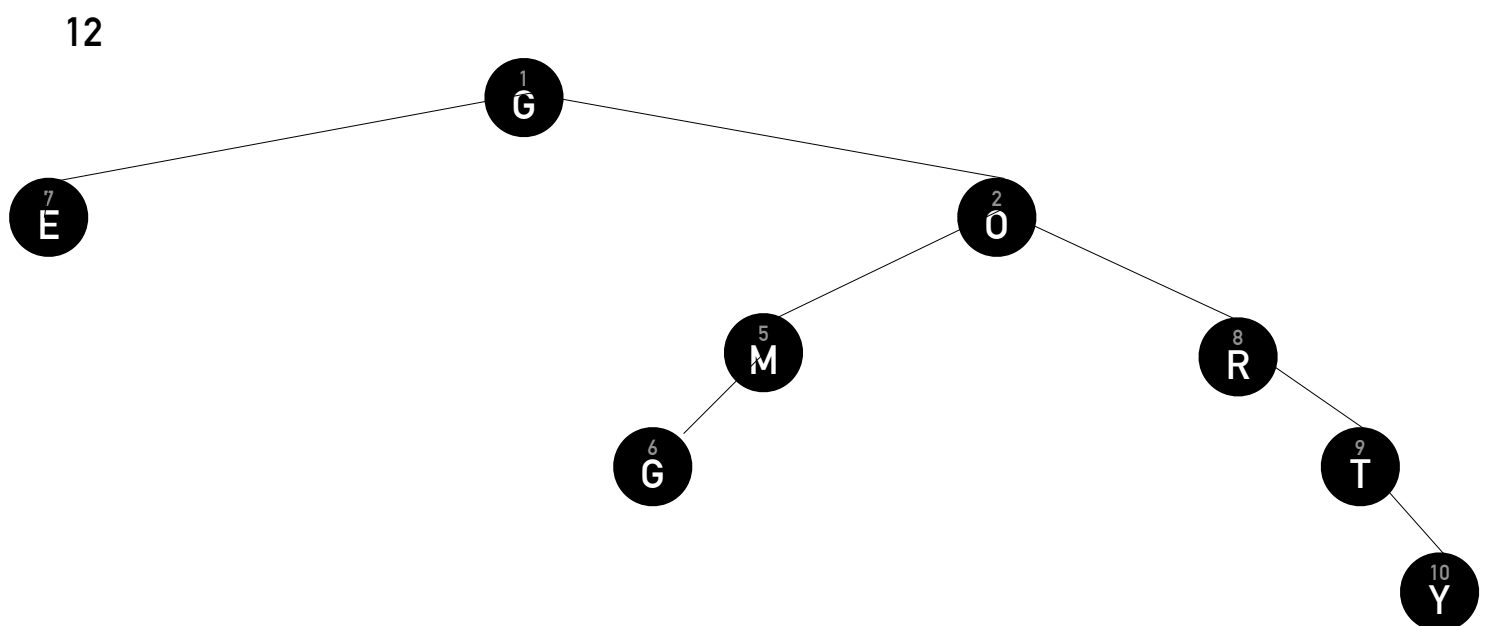
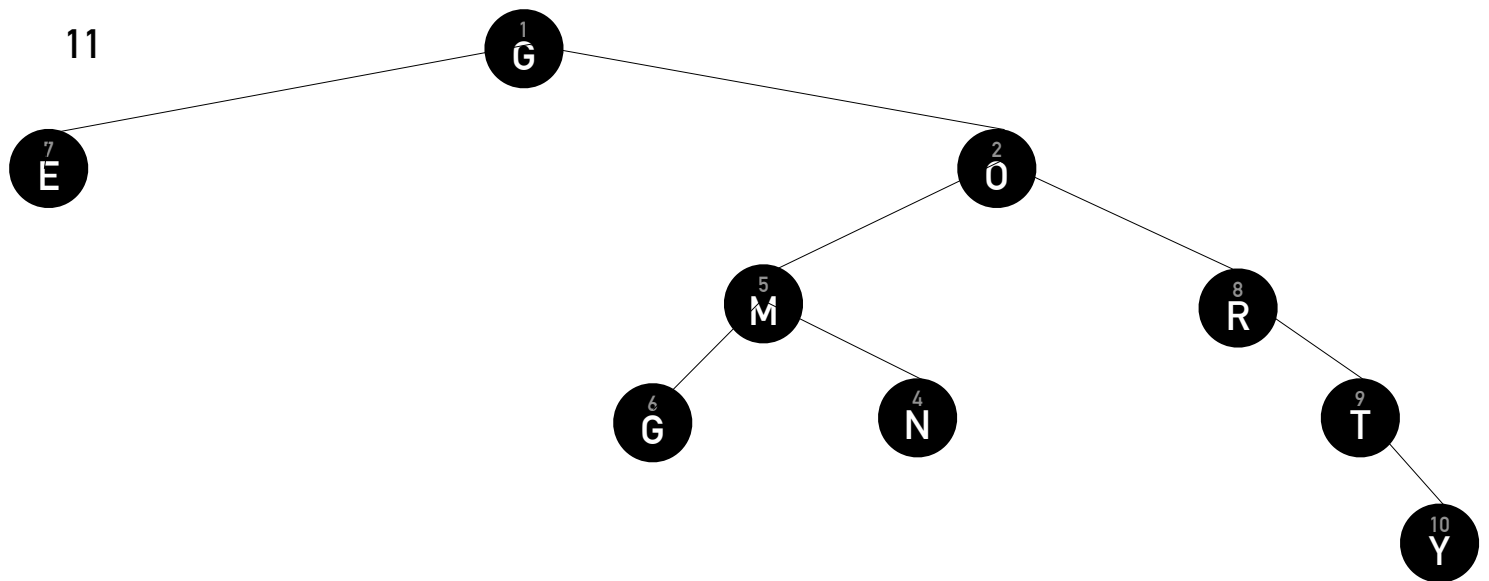
หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น ..... **GEOIGNMRTY** .....

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

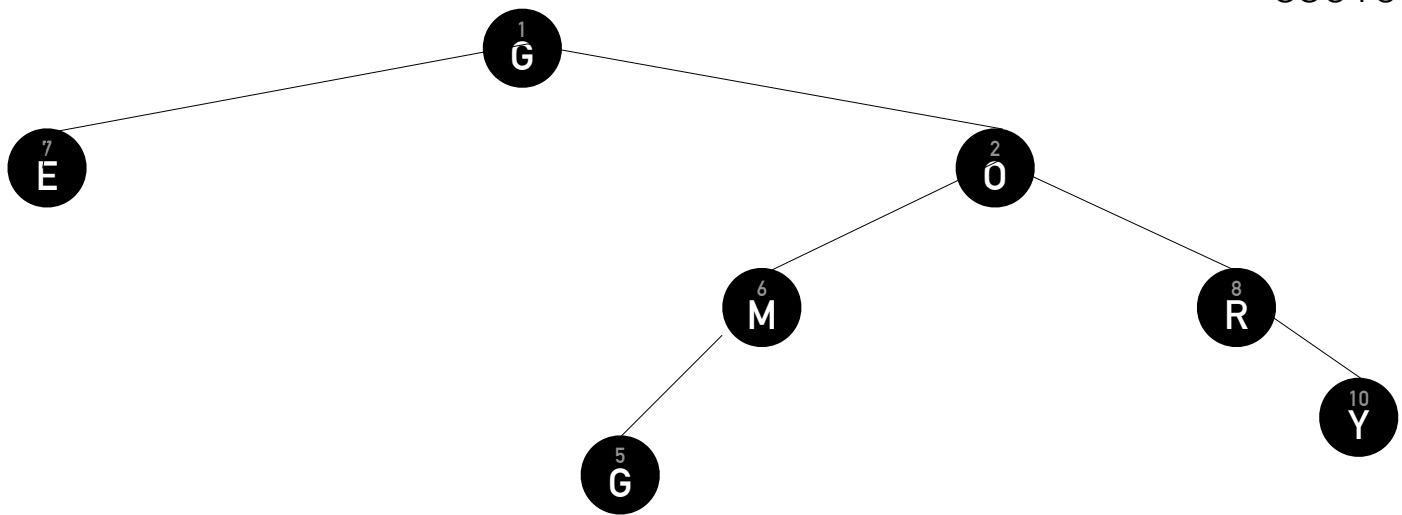
หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น ..... **EGMNIYTROG** .....

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

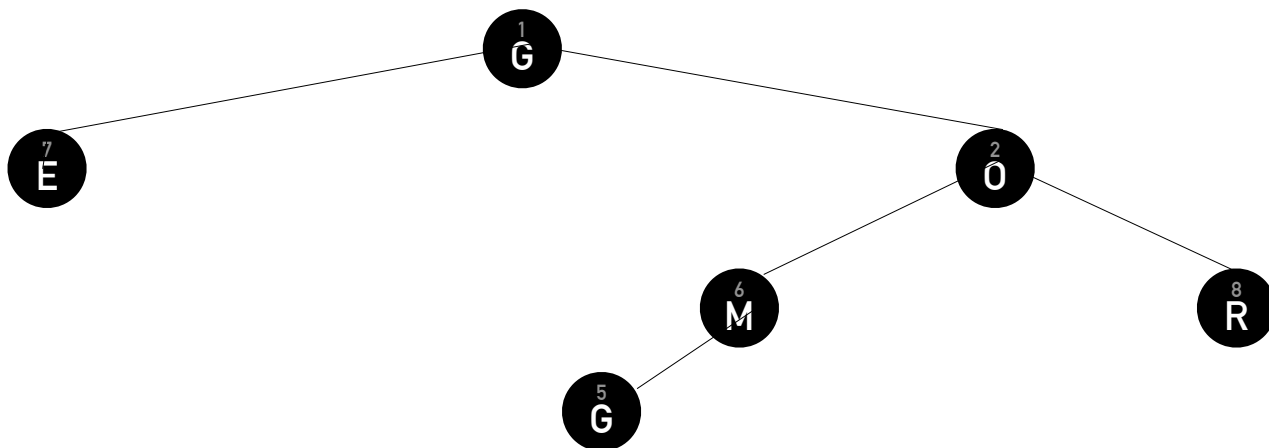
```
11. delete_node(&(tree2.root->right->left));  
12. delete_node(&((tree2.root->right->left)->right));  
13. delete_node(&((tree2.root->right->right)->right));  
14. delete_node(&((tree2.root->right->right)->right));
```



13



14



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

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

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

5. จงเขียนแผนภาพการทำงานของ 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');
```

ABCDEFGH

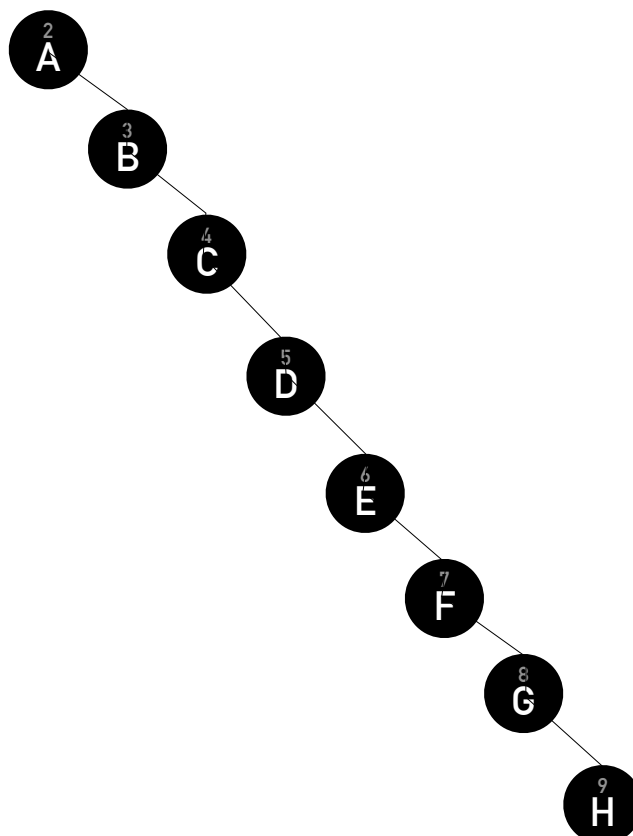
หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น .....

ABCDEFGH

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

HGFEDCBA

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

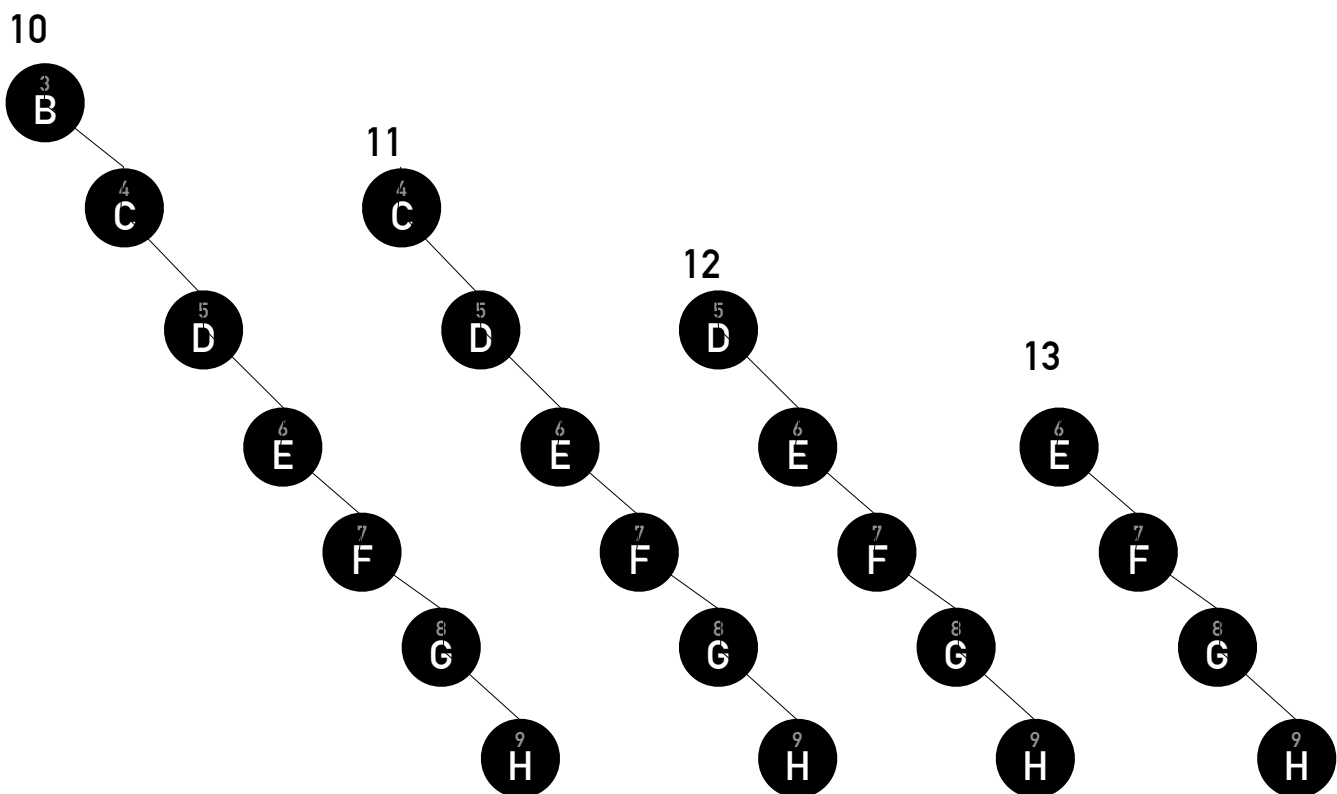


6. ต่อจากข้อ 3 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

```

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

```



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)

.....แบบไม่ balance เพราะแบบ balance มันจะทำให้ subtree สมดุล ไม่ได้ขึ้นลำดับใหม่

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

.....แบบ balance เพราะลำดับน้อย ไม่ต้องผ่าน node เยอะ

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

.....Balance

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

.....พยายามให้ tree อยู่ในรูปของ balance เพราะจะทำให้เวลาที่ใช้ในการหาข้อมูลน้อยลง

.....จากลำดับที่น้อย ซึ่งไม่ต้องผ่าน node เยอะ เพื่อที่จะหา node ที่ต้องการตัวเดียว

.....(การ balance ต้นไม้ อาจทำให้การเพิ่มหรือลบ โหนด ในต้นไม้มีความยืดหยุ่นน้อยลง)

.....ดังนั้นควรใช้แบบ Balance เพื่อประสิทธิภาพที่ดีกว่า