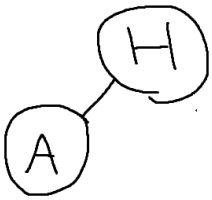
Binary Search Tree

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

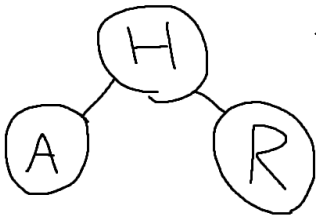
```
0.    BST tree;
1.    tree.insert('H');
2.    tree.insert('A');
3.    tree.insert('R');
4.    tree.insert('H');
5.    tree.insert('U');
6.    tree.insert('I');
```

1.

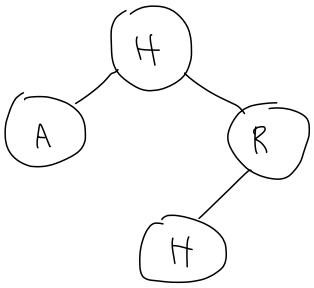
2.



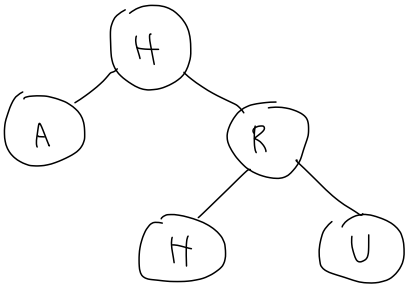
3.



4.

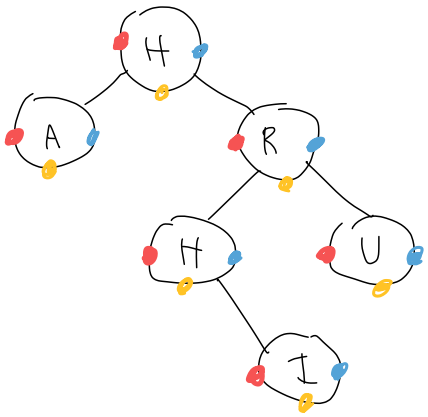


5.



ABCDEFGHIJKLMNOP
QRSTUVWXYZ

6.



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

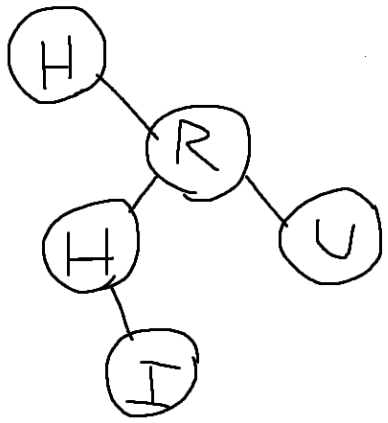
หาก travers tree ดังกล่าว แบบ In-order จะได้ output เป็น ..... A H H I R U.....

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

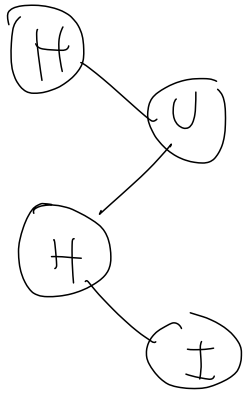
2. ต่อจากข้อ 1 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

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

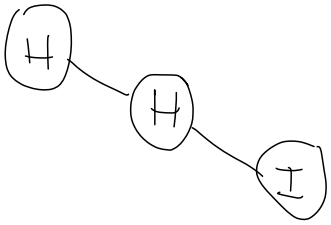
7.

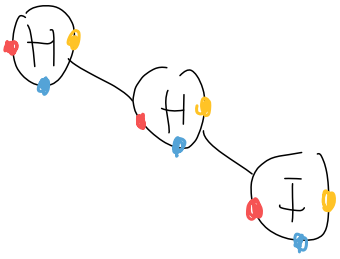


8.



9.





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

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

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

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

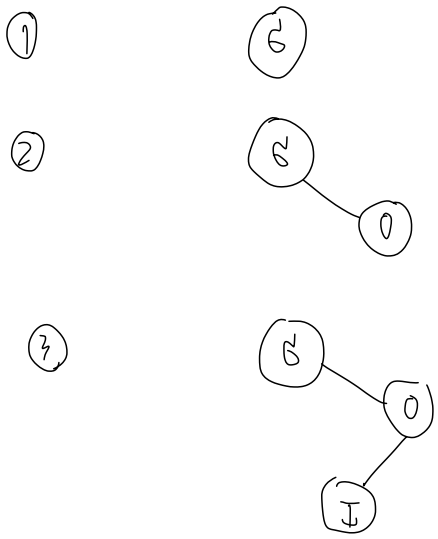
```
0.    BST tree2;
1.    tree2.insert('G');
2.    tree2.insert('O');
3.    tree2.insert('I');
4.    tree2.insert('N');
5.    tree2.insert('G');
6.    tree2.insert('M');
7.    tree2.insert('E');
8.    tree2.insert('R');
9.    tree2.insert('T');
10.   tree2.insert('Y');
```

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

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

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

ABCDEFGHIJKLMNOP
QRSTUVWXYZ



4

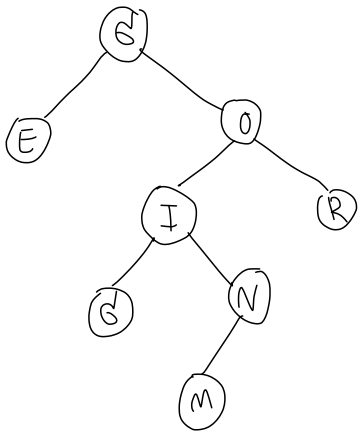
5

6

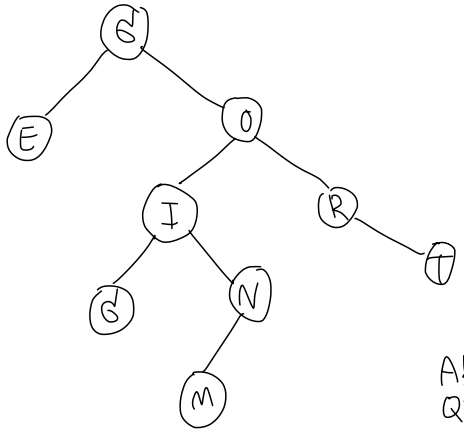
A B C D E F G H I J K L M N O P
Q R S T U V W X Y Z

7

(c)

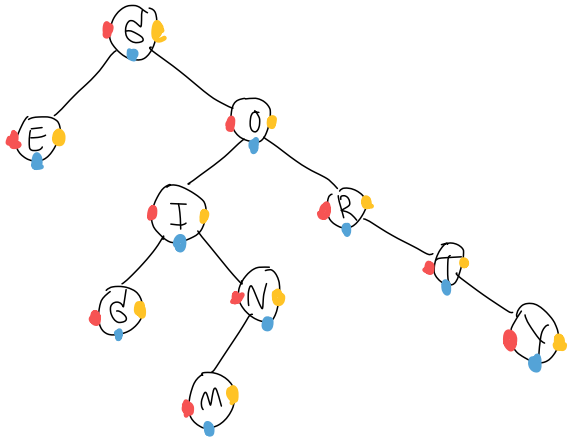


(g)



ABCDEFGHIJKLMNOP
QRSTUVWXYZ

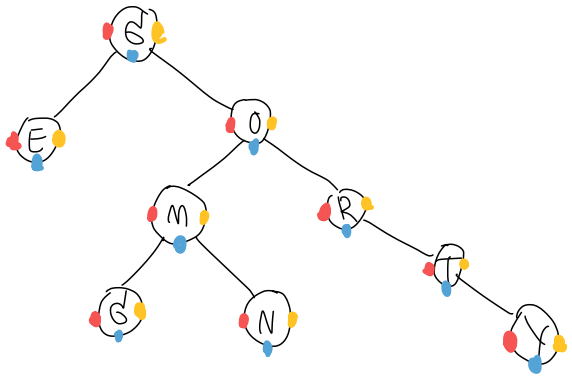
(10)



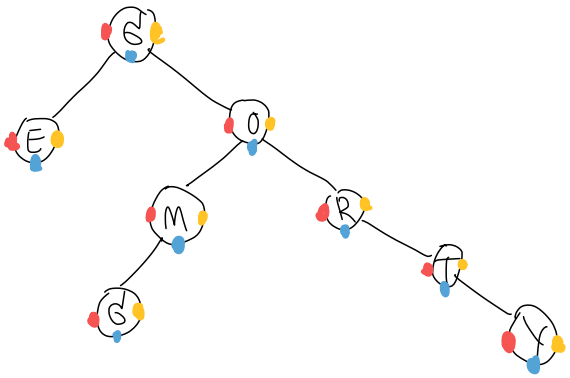
4. ต่อจากข้อ 3 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

```
11.    delete_node(&(tree2.root->right->left));//ฮ
12.    delete_node(&((tree2.root->right->left)->right));//ณ
13.    delete_node(&((tree2.root->right->right)->right));//ท
14.    delete_node(&((tree2.root->right->right)->right));//ย
```

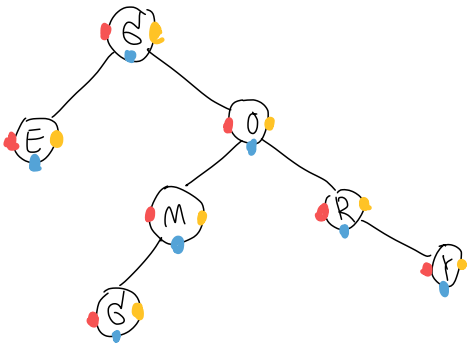
11



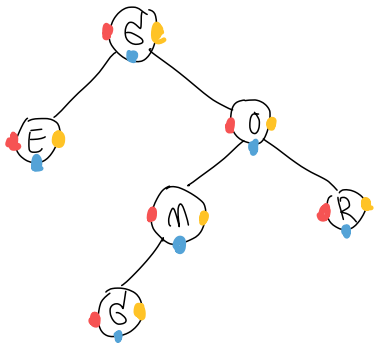
12



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.....

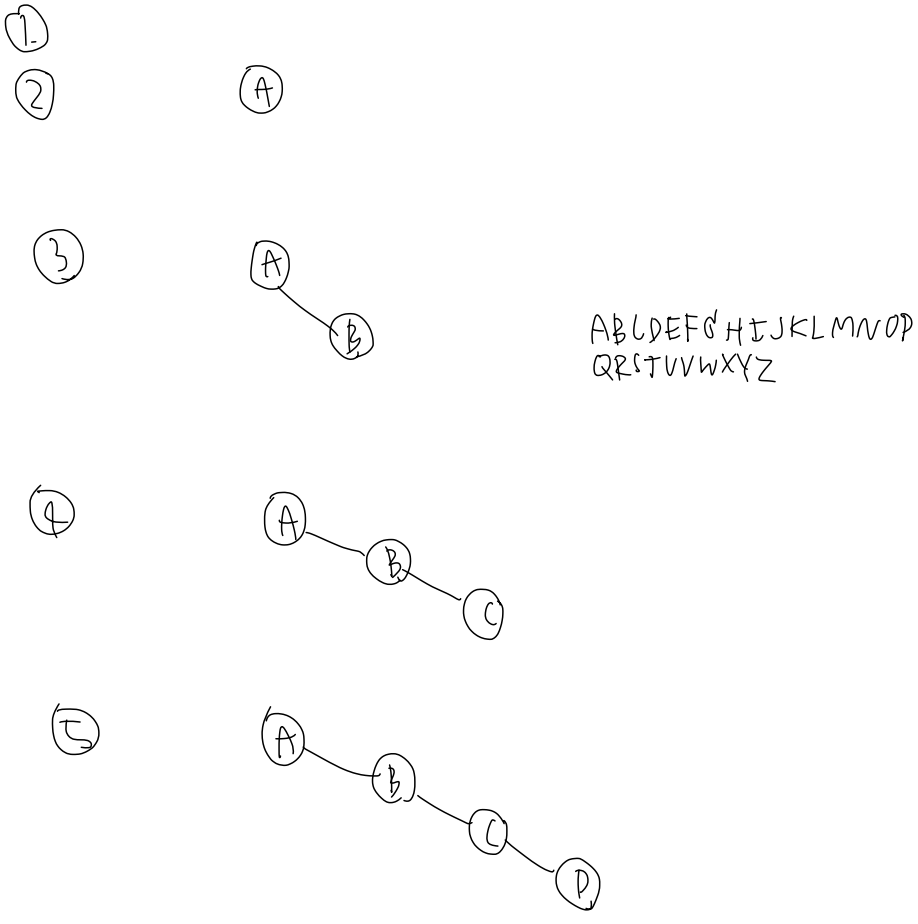
5. จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว (ออกแบบบรรทัดเองเลยครับ)

```
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');
```

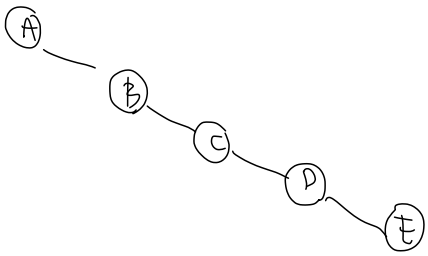
หาก travers tree ดังกล่าว แบบ Pre-order จะได้ output เป็น ...A B C D E F G H.....

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

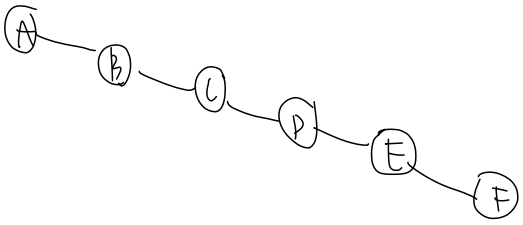
หาก travers tree ดังกล่าว แบบ Post-order จะได้ output เป็น ...H G F E D C B A.....



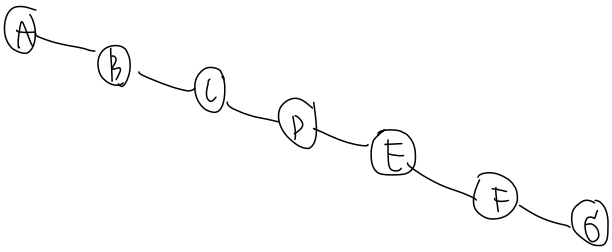
6



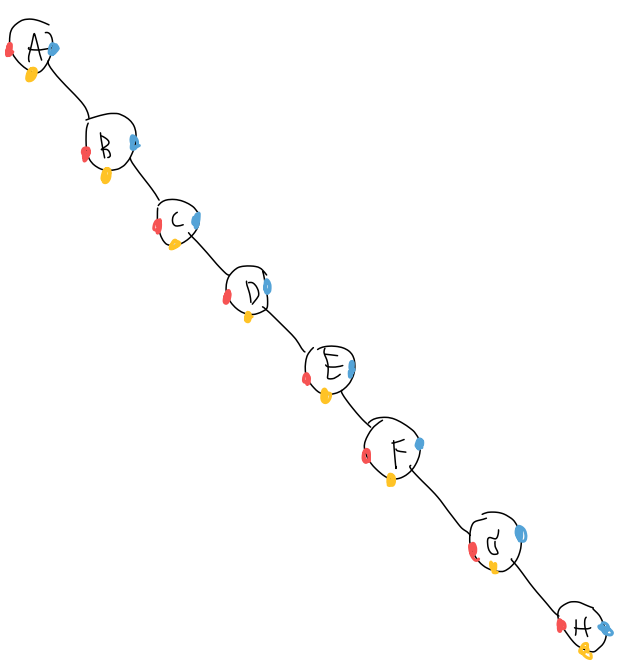
7



8



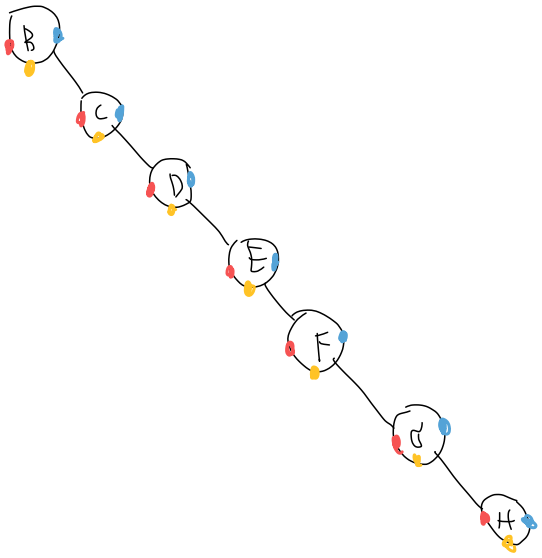
9



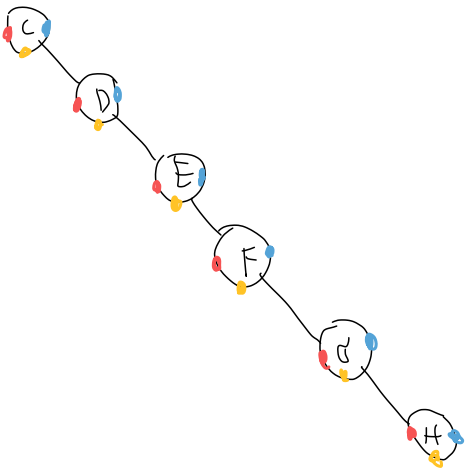
6. ต่อจากข้อ 3 หากใช้ code ดังต่อไปนี้ จงเขียนแผนภาพของการทำงานของ Binary search tree ในโปรแกรมต่อไปนี้ที่ละบรรทัด และตอบคำถามเกี่ยวกับการท่อง (Traversal) ไปใน tree ดังกล่าว

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

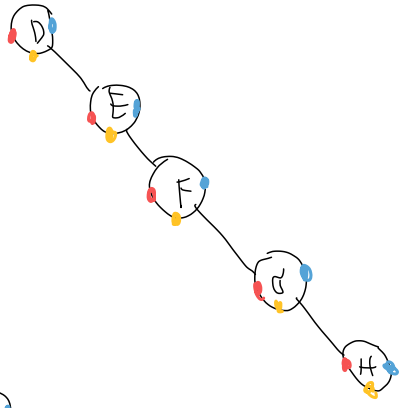
10



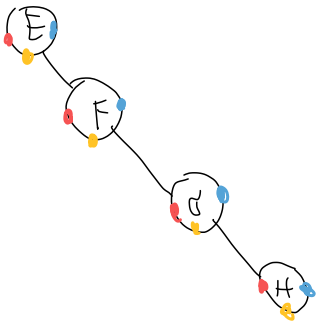
11



12



13



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

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

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

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

ไม่ balance ให้เวลามากกว่าเนื่องจากข้อมูลจะไปกองกันอยู่ด้านใดด้านหนึ่งทำให้ลำดับมากกว่าเมื่อเทียบกับแบบ balance ที่มีข้อมูลเท่ากัน

8. BST ที่ balance กับ BST ที่ไม่ balance หากต้องการ search แบบใหน ให้เวลาในการค้นหาน้อยกว่ากันอย่างไร (ขอสั้นๆ)

balance เพราะมีลำดับขั้นน้อยกว่า

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)

balance เนื่องจากเมื่อเทียบจำนวนชุดข้อมูลที่เท่ากันระหว่าง tree ที่ balance และไม่ balance พบว่า tree ที่ balance จะมีลำดับขั้นที่น้อยกว่า ทำให้มีประสิทธิภาพในการจัดเก็บ, ค้นหาได้ดีกว่า คือ ข้อมูลเป็นระเบียบ ไม่กองอยู่ฝั่งใดฝั่งหนึ่ง และทำให้ง่าย และรวดเร็วต่อการค้นหาข้อมูล เป็นความเร็ว log n เท่า