Homework #12 B09705039劉惟恩

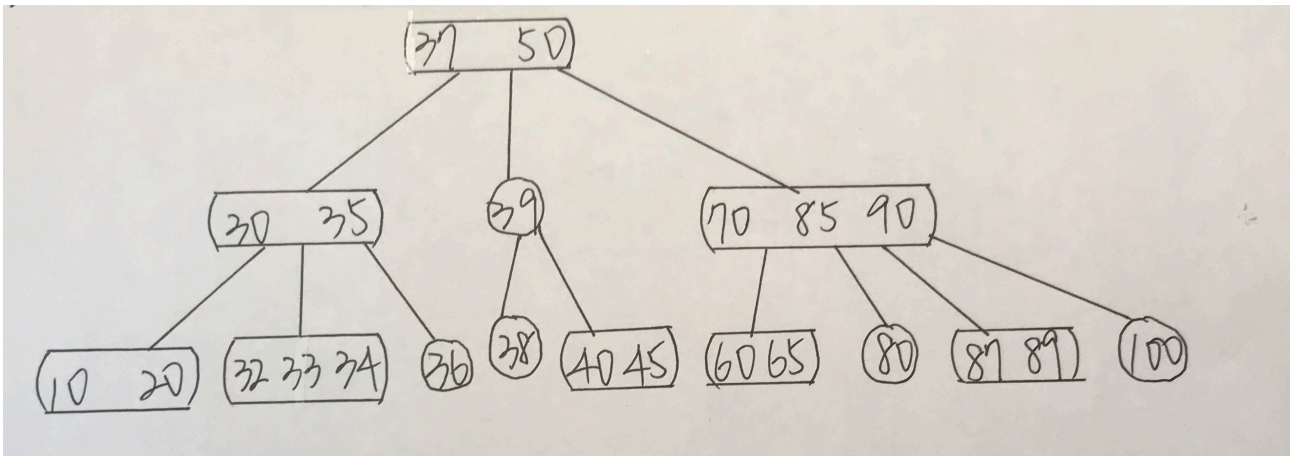
Solutions:

1.

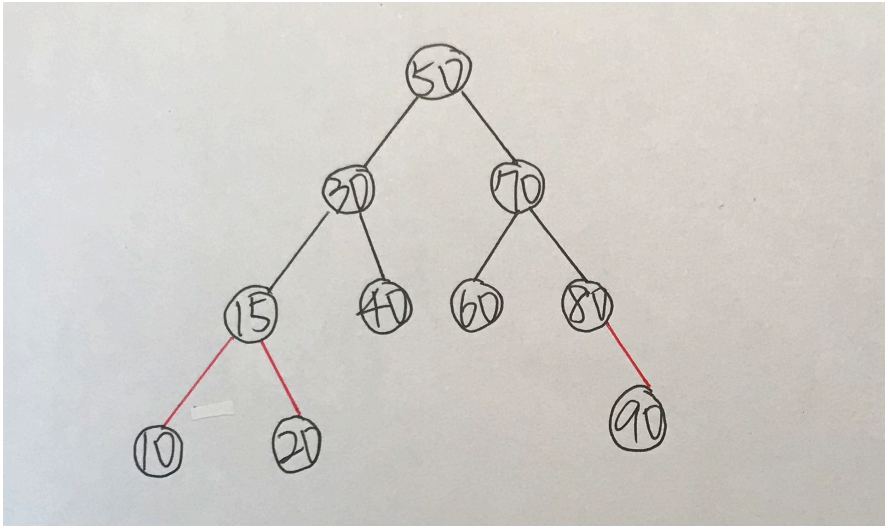
```
//insertion:
void insertion(node data)
{
   if(tree is empty)
      root = data
   else
   {
      while(current node != nullptr)
      {
         colorChange(current node)

         if(data is bigger than current node)
            travel right
         else
            travel left
      }
      use a red pointer to point the previous node to the data
   }
}

void colorChange(node pointer node)
{
   if(node as two red pointers)
   {
      change color of the two pointers into black

      if(parent node exists)
         change the pointer pointed to the node into red
         if(the pointer of grandparent node to parent node is also red)
            do rotation to parent nodes
   }
}

//removal:
bool removal(node data)
{
   if(tree is empty)
      return false
   else if(tree only has a node)
      delete root
   else
   {
      while(current node != data)
      {
         if(data is bigger than current node)
            travel right
         else
            travel left
      }
      if(data is a leaf)
         delete the leaf
      else
         move the data to a leaf nearby by swapping it with its child
         then check colorChange(current ptr) after every swap
         delete the leaf
```

```
    }
}

//retrieval and traversal operations: Same as the algorithms used in binary search tree.

node pointer Retrieval(int data)
{
    while(current node != data)
    {
        if(data is bigger than current node)
            travel right
        else
            travel left
    }
    return current node pointer
}

//Traversal same as inorder traversal.
void inorderTraversal()
{
    if(current node == nullptr)
        return;

    inorderTraversal(current node -> left)
    node -> data
    inorderTraversal(current node -> right)
}
```

2.



3.



4.