**一．平衡二叉树：**

1**.性质：**

他的左子树和右子树都是平衡二叉树，且左子树和右子树高度之差的绝对值不超过1。

平衡二叉树的常用实现方法有红黑树、AVL、替罪羊树、Treap、伸展树等。 最小二叉平衡树的节点的公式如下 F(n)=F(n-1)+F(n-2)+1 这个类似于一个递归的数列，可以参考Fibonacci数列，1是根节点，F(n-1)是左子树的节点数量，F(n-2)是右子树的节点数量。

**2.平衡因子：**  
某结点的左子树与右子树的高度(深度)差即为该结点的平衡因子（BF,Balance Factor）。  
平衡二叉树上所有结点的平衡因子只可能是 -1，0 或 1。

**3.平衡树有很多种, 其中有几类树维持平衡的方法。  
二叉左旋（RR：左单旋转）**一棵二叉平衡树的子树，根是Root，左子树是x，右子树的根为RootR，右子树的两个孩子树分别为RLeftChild和RRightChild。则左旋后，该子树的根为RootR，右子树为RRightChild，左子树的根为Root，Root的两个孩子树分别为x（左）和RLeftChild（右）。  
**二叉右旋（LL：右单旋转）**一棵二叉平衡树的子树，根是Root，右子树是x，左子树的根为RootL，左子树的两个孩子树分别为LLeftChild和LRightChild。则右旋后，该子树的根为RootL，左子树为LLeftChild，右子树的根为Root，Root的两个孩子树分别为LRightChild（左）和x（右）。

**LR平衡旋转（先左后右）**

当在ptr的左子树的右子树中插入一个结点后，造成了ptr平衡因子为-2的不平衡，将ptr向下找到当前结点的左孩子的右孩子，先进行左单旋ptr->left = subL，然后将ptr的右子树断开指向subR，此时便完成了旋转，最后将平衡因子进行更新。

**RL平衡旋转（先右后左）**

先右单旋再左单旋，是先左后右的镜像旋转。

左子节点与右子节点对称的树就是平衡树，否则就是非平衡树。  
非平衡树会影响树中数据的查询，插入和删除的效率。比如当一个二叉树极不平衡时，即所有的节点都在根的同一侧，此时树没有分支，就变成了一个链表。数据的排列是一维的，而不是二维的。在这种情况下，查找的速度下降到O(N)，而不是平衡二叉树的O(logN)。  
为了能以较快的时间O(logN)来搜索一棵树，需要保证树总是平衡的（或者至少大部分是平衡的）。这就是说对树中的每个节点在它左边的后代数目和在它右边的后代数目应该大致相等。

1. C➕➕
2. 在C语言中，我们通常会使用 scanf 和 printf 来对数据进行输入输出操作。在C++语言中，C语言的这一套输入输出库我们仍然能使用，但是 C++ 又增加了一套新的、更容易使用的输入输出库。
3. C++ 中的输入与输出可以看做是一连串的数据流**，输入**即可视为从文件或键盘中输入程序中的一串数据流，而**输出**则可以视为从程序中输出一连串的数据流到显示屏或文件中。  
     
   3.在编写 C++ 程序时，如果需要使用输入输出时，则需要包含头文件iostream，它包含了用于输入输出的对象，例如常见的**cin表示标准输入、cout表示标准输出、cerr表示标准错误**。  
   4.iostream 是 Input Output Stream 的缩写，意思是“输入输出流”。  
   cout 和 cin 都是 C++ 的内置对象，而不是关键字。C++ 库定义了大量的类（Class），程序员可以使用它们来创建对象，cout 和 cin 就分别是 ostream 和 istream 类的对象，只不过它们是由标准库的开发者提前创建好的，可以直接拿来使用。这种在 C++ 中提前创建好的对象称为内置对象。  
     
   5.使用 cout 进行输出时需要紧跟<<运算符，使用 cin 进行输入时需要紧跟>>运算符，这两个运算符可以自行分析所处理的数据类型，因此无需像使用 scanf 和 printf 那样给出格式控制字符串。
4. "Please input a int number:"这样的一个字符串，以提示用户输入整数，其中endl表示换行，与C语言里的\n作用相同。当然这段代码中也可以用\n来替代endl，这样就得写作：  
   cout<<"Please input an int number:\n";  
     
   7.endl 最后一个字符是字母“l”，而非阿拉伯数字“1”，它是“end of line”的缩写。cout 能够连续地输出。同样 cin 也是支持对多个变量连续输入的。