**1.**

(a) What is the key of each node in the AVL tree?

里面包括从i到j之和（i，j）

What other attributes are stored in each node?

Self.key = key #(I,j)

Self.sub\_sum = s #i到j之和

Self.left = left

Self.right = right

左支存的是（向下取整）：从a（i）到a（n/2）

右支存的是（向上取整）：a（n/2）到a（n）

真实的ai全是在leaf。

(b) Write the pseudo-code of your PARTIAL-SUM operation

(c) Describe in clear and concise English how your CHANGE operation works,

如果我们key存的是partial sum的话那如果我们change了一个key，他后面的所有node的key就都得变，所以这已经是linear time了。

**2.**

如果有等价替换，我们就跳过前三位，因为后一位无论前面怎么换都会在一样的位置

如果没有等价替换，我们就

如果是前三位是从小到大的（一个数列），那第四位的层级可能是在第一层，第二层，第三层，所以我们就跳过第一个，

但如果前三位是full的，在结构上就可以被视为一个东西，里面的互换和外面没有关系，外界对他们没有影响。直接跳过那三个，看下面三个

但如果前三个左右右/右左左的结构，就跳过第一个，看后两个和再后一个的关系。

在list里，从第一个比他小的数开始都是他的左支，从第一个比他大的数开始都是他的右支，都可以交换（会产生一样的结果），其中，每个小tree里的左支和右支又都是可以交换的。

孩子不可以在自己的祖先前面，但孩子可以在自己的siblings前面。

右支不可能超过他的顺序。

Level 高的不能在level低的之后。

右支可以在他父亲之前的任何位置。（大的东西先出现后出现没什么关系）

一个root，找到第一个比他大的结点，把他的所有的做孩子清理出来，左支的所有