12-F3

优先级队列

左式堆：合并算法

邓俊辉

deng@tsinghua.edu.cn

左之左之，君子宜之；右之右之，君子有之

LeftHeap

❖ template <typename T> //基于二叉树，以左式堆形式实现的优先级队列

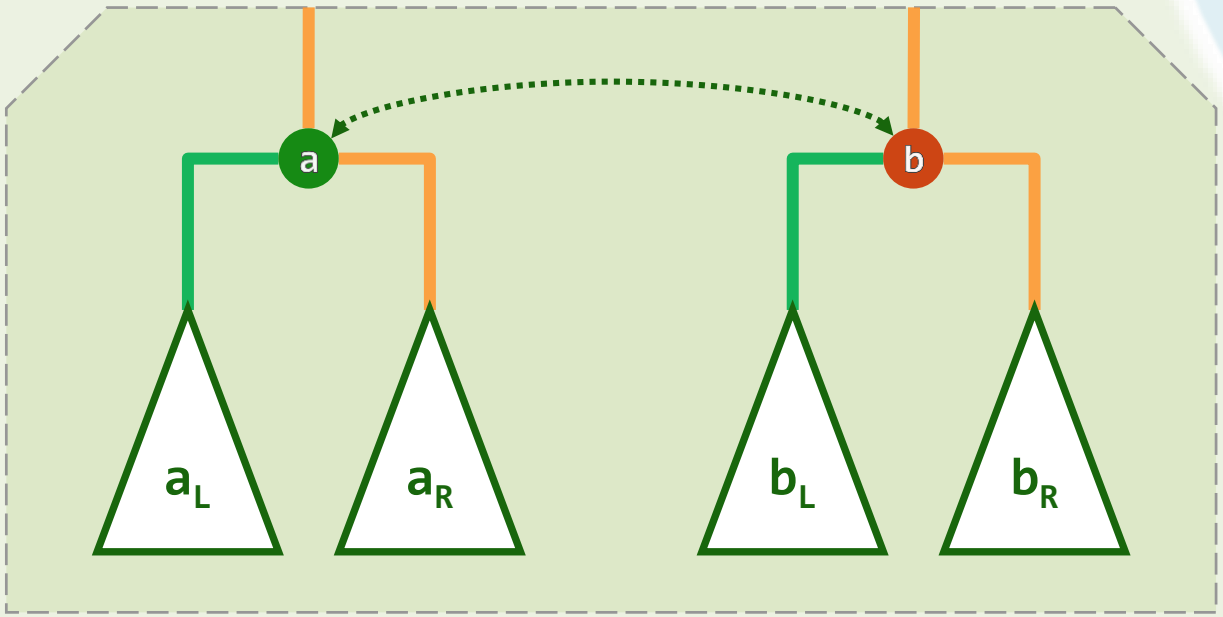
```
class PQ_LeftHeap : public PQ<T>, public BinTree<T> {
public:
   T getMax() { return _root->data; }
   void insert(T); T delMax(); //均基于统一的合并操作实现...
};
```

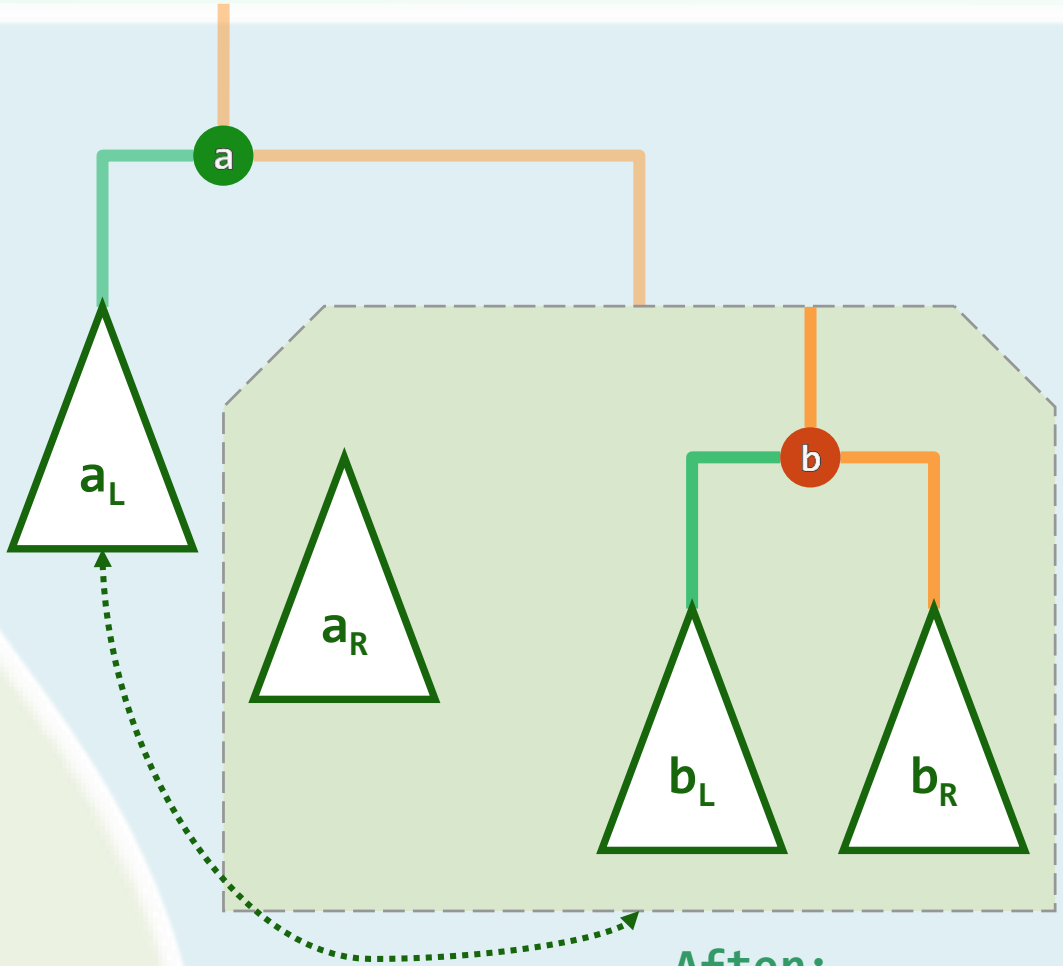
❖ template <typename T>

```
static BinNodePosi<T> merge( BinNodePosi<T>, BinNodePosi<T> );
```

递归：前处理 + 后处理

Before:
compare priority &
swap if necessary



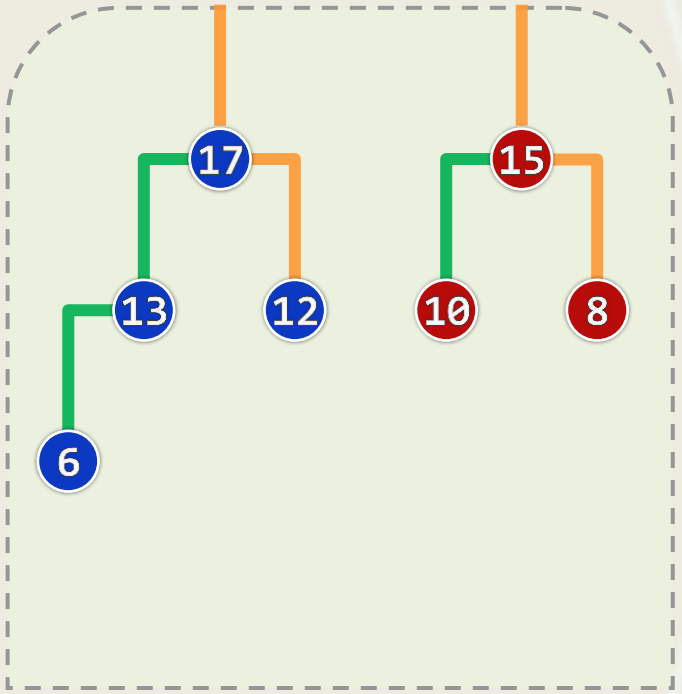


After:
compare NPL &
swap if necessary

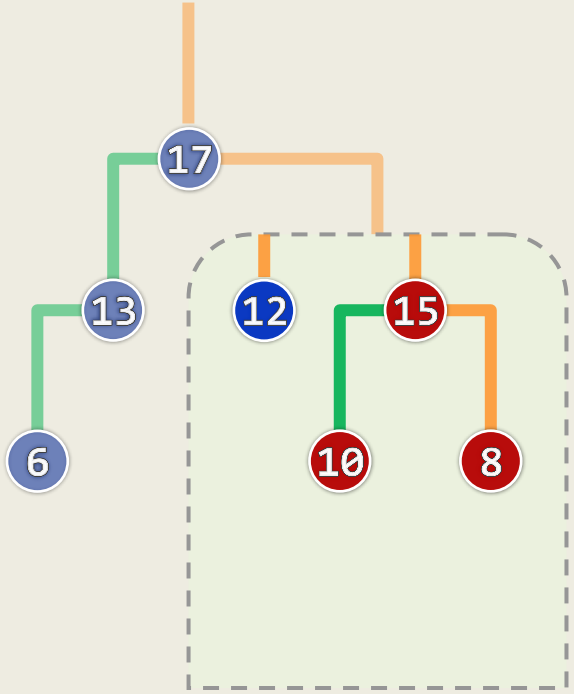
实现

```
❖ template <typename T>
  static BinNodePosi<T> merge( BinNodePosi<T> a, BinNodePosi<T> b ) {
     if ( !a ) return b; if ( !b ) return a; //递归基
     if ( lt( a->data, b->data ) ) swap( b, a ); //确保a不小
     ( a->rc = merge( a->rc, b ) )->parent = a; //将a的右子堆，与b合并
     if ( ! a->lc || a->lc->npl < a->rc->npl ) //若有必要
        swap( a->lc, a->rc ); //交换a的左、右子堆，以确保左子堆的npl不小
     a->npl = a->rc ? 1 + a->rc->npl : 1; //更新a的npl
     return a; //返回合并后的堆顶
  }
```

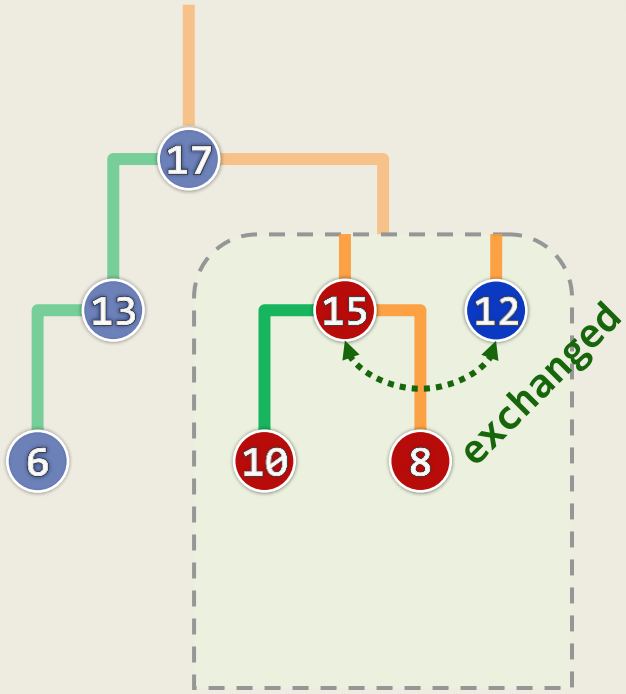
实例（1/5）



(a)

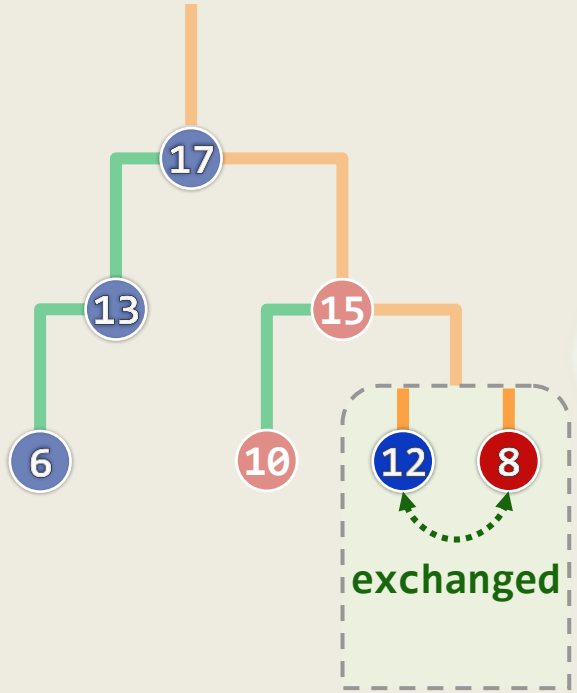


(b)

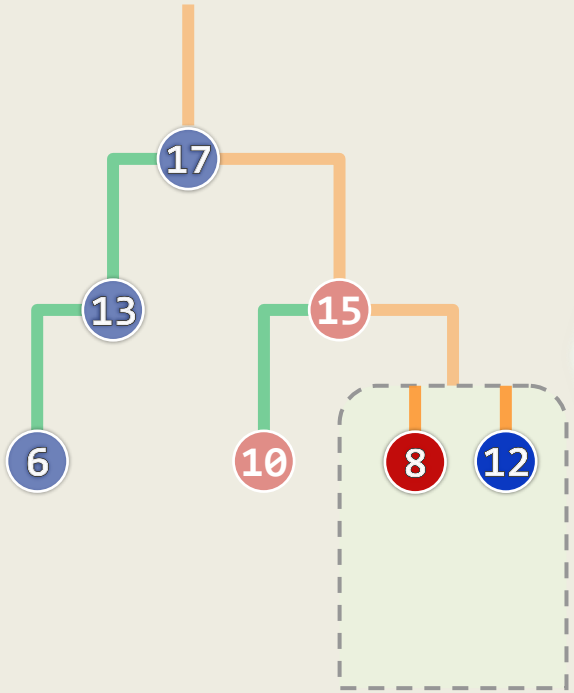


(c)

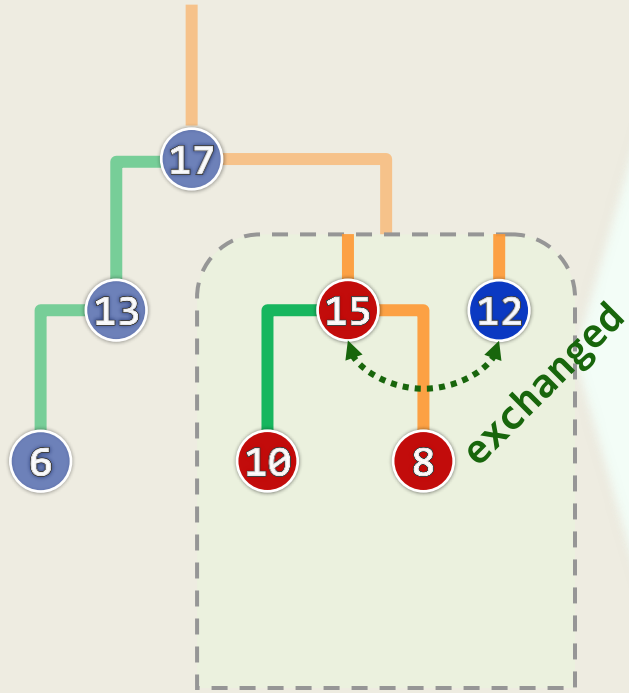
实例（2/5）


17
13
15
6
10
12
8
exchanged

(e)

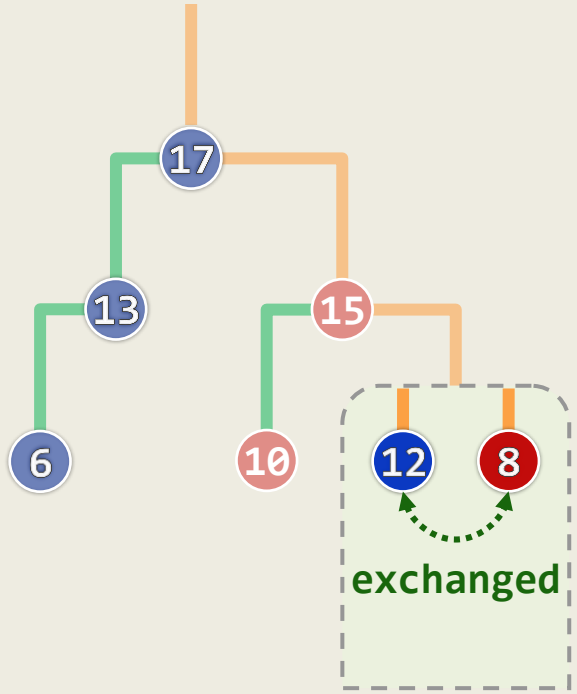

17
13
15
6
10
8
12

(d)

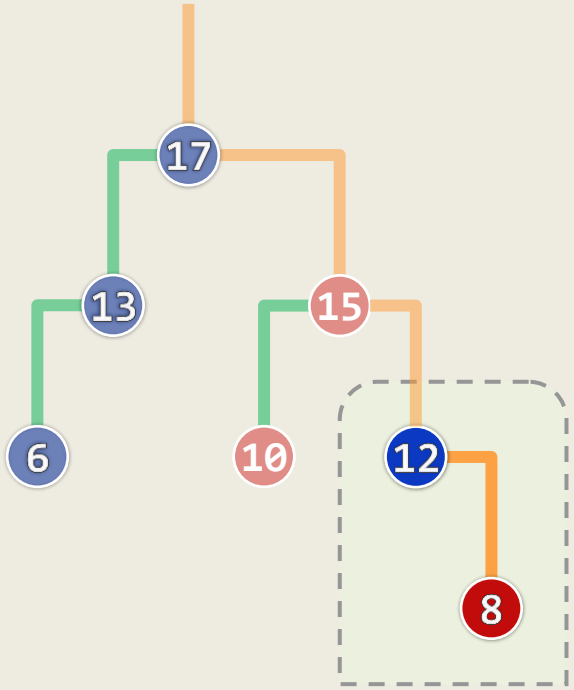

17
13
15
12
6
10
8
exchanged

(c)

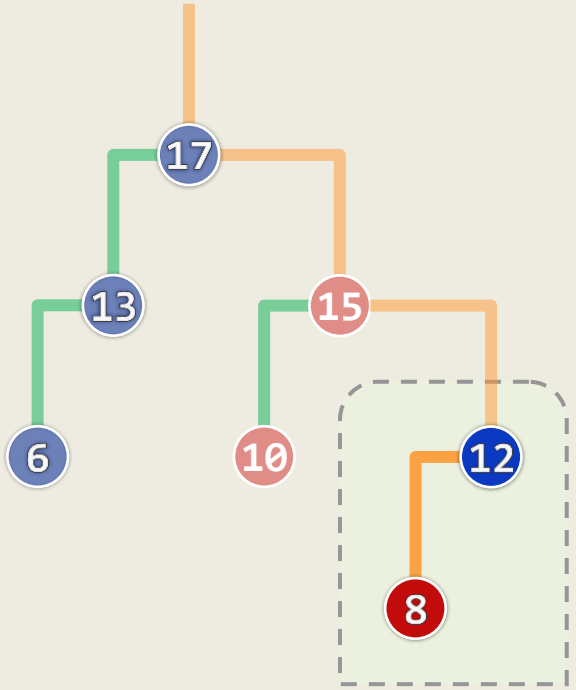
实例（3/5）



(e)

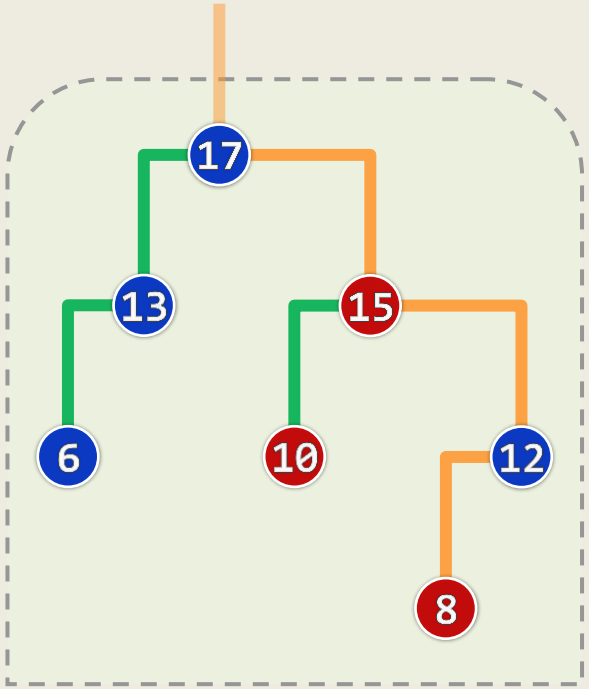


(f)

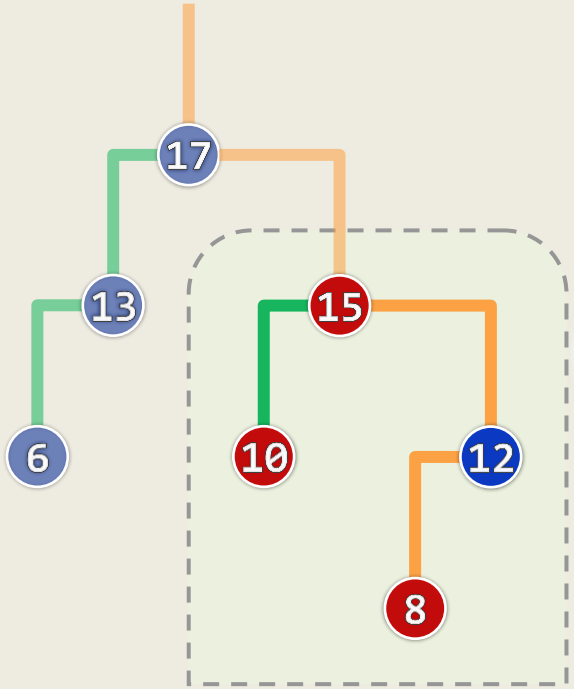


(g)

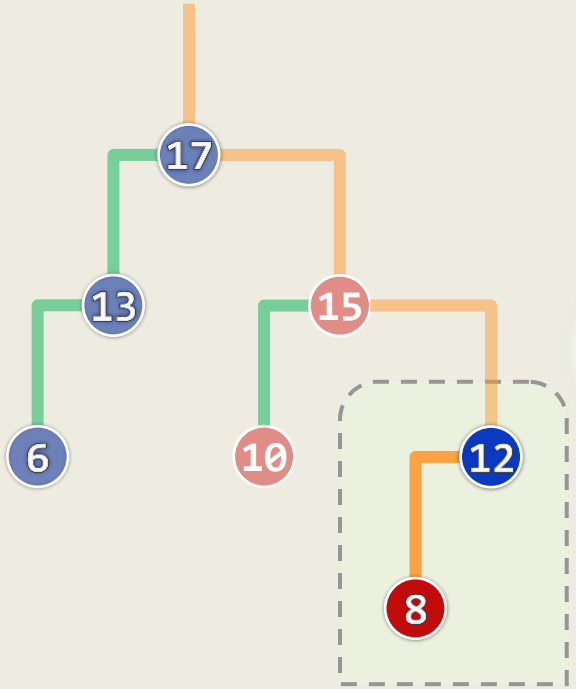
实例（4/5）



(i)

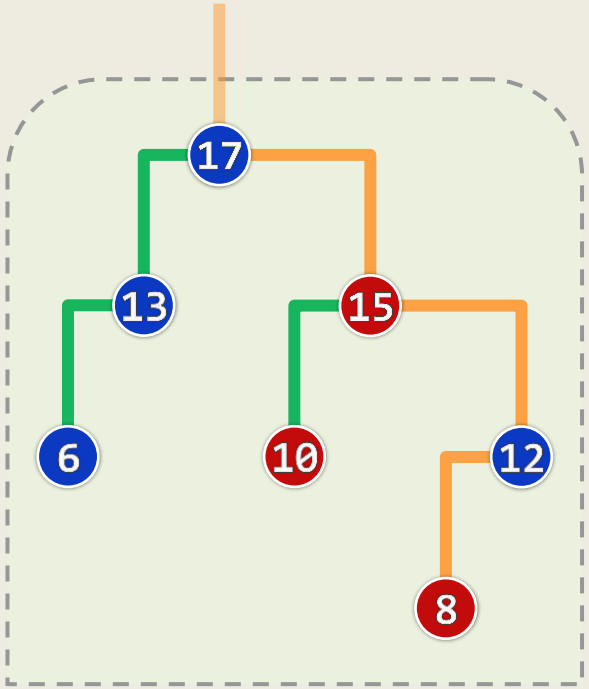


(h)

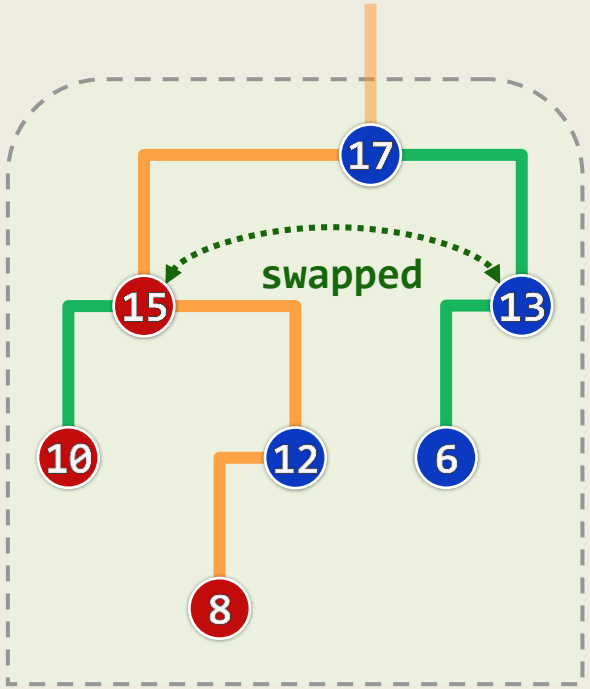


(g)

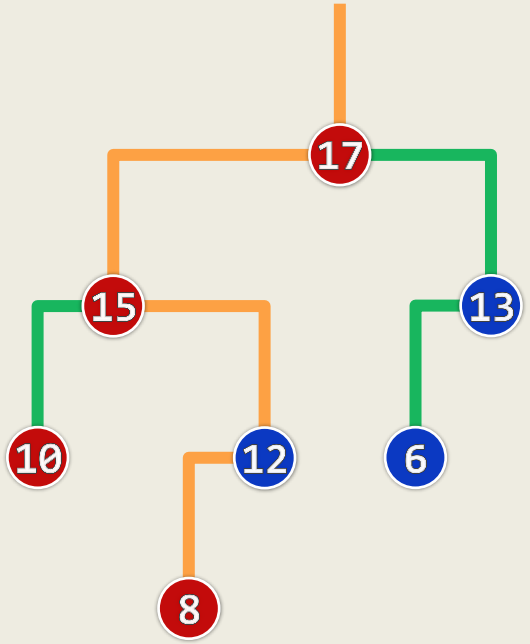
实例（5/5）



(i)



(j)



(k)