6.26 合并图6.58中的两个斜堆

6.27 写出将关键字1到15依序插入到一斜堆内的结果。

6.28 证明下述结论成立或不成立：如果将关键字1到2^k-1依序插入到一个初始为空的斜堆中，那么结果形成一棵理想平衡树(perfectly balanced tree)。

6.29 使用标准的二叉堆算法可以建立一个N个元素的斜堆。我们能否使用练习6.25中描述的同样的合并策略用于斜堆而得到$O(N)$运行时间？

6.30 证明二项树B_k以二项树B_0，B_1，...，B_{k-1}作为其根的儿子。

6.31 证明高度为k的二项树在深度d有$\binom{k}{d}$个节点。

6.32 将图6-59中的两个二项队列合并。

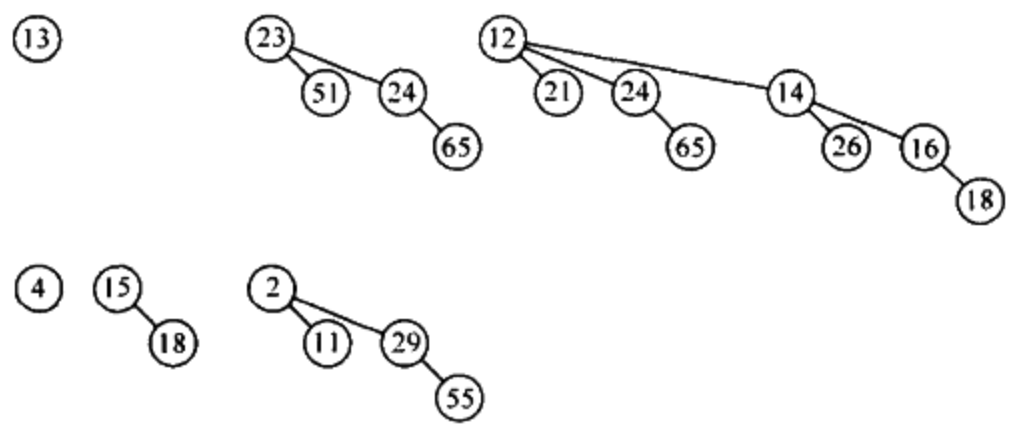

图6-59 练习6.32中的输入

6.33 a. 证明，向初始为空的二项队列进行N次insert在最坏情形下花费$O(N)$的时间。

b. 给出一个算法来建立有N个元素的二项队列，在元素间最多使用$N-1$次比较。

*c. 提出一个算法，以$O(M+\log N)$最坏情形运行时间将M个节点插入到N个元素的二项队列中。证明该算法的界。

6.34 写出一个高效的例程使用二项队列来完成insert操作。不要调用merge方法。

6.35 对于二项队列：

a. 如果没有树留在H_2中且carry树为null，则修改merge例程以终止合并。

b. 修改merge使得较小的树总被合并到较大的树中。

*6.36 假设我们将二项队列扩充为允许每个结构同一高度至多有两棵树。我们能否在其他操作保留为$O(\log N)$时得到插入为$O(1)$的最坏情形时间？

6.37 设有许多盒子，每个盒子都能容纳总重量C，而物品i_1，i_2，i_3，…，i_N分别重w_1，w_2，w_3，…，w_N。现在想要把所有的物品包装起来，但任一盒子都不能放置超过其容量的重物，而且要使用尽量少的盒子。例如，若$C=5$，物品分别重2，2，3，3，则我们可用两个盒子解决该问题。

一般说来，这个问题很困难，尚不知有高效的解决方案。编写程序高效地实现下列各近似解法：

*a. 将重物放入能够承受其重量的第一个盒子内(如果没有盒子拥有足够的容量就开辟一个新的盒子)(该方法以及后面所有的方法都将得出3个盒子，这不是最优的结果)。

b. 把重物放入对其有最大空间的盒子内。

*c. 把重物放入能够容纳它而又不过载的装填得最满的盒子中。