# 左偏树

## 左偏树的构建

算法一 暴力算法——逐个节点插入，时间复杂度为O(nlogn)

算法二 仿照二叉堆的构建算法（二分合并）

## 插入新节点

单节点的树一定是左偏树，因此向左偏树插入一个节点可以看作是对两棵左偏树的合并。

Procedure Insert(x, A)

B ← MakeIntoTree(x)

A ← Merge(A, B)

End Procedure

## 删除最小节点

在删除根节点后，剩下的两棵子树都是左偏树，需要把他们合并。

Function DeleteMin(A)

t ← key(root(A))

A ← Merge(left(A), right(A))

return t

End Function

## 左偏树的合并

Function Merge(A, B)

If A = NULL Then return B

If B = NULL Then return A

If key(B) < key(A) Then swap(A, B)

right(A) ← Merge(right(A), B)

If dist(right(A)) > dist(left(A)) Then

swap(left(A), right(A))

If right(A) = NULL Then dist(A) ← 0

Else dist(A) ← dist(right(A)) + 1

return A

End Function

## 删除任意已知节点

Procedure Delete(x)

q ← parent(x)

p ← Merge(left(x), right(x))

parent(p) ← q

If q ≠ NULL and left(q) = x Then

left(q) ← p

If q ≠ NULL and right(q) = x Then

right(q) ← p

While q ≠ NULL Do

If dist(left(q)) < dist(right(q)) Then

swap(left(q), right(q))

If dist(right(q))+1 = dist(q) Then

Exit Procedure

dist(q) ← dist(right(q))+1

p ← q

q ← parent(q)

End While

End Procedure