二叉堆的性质

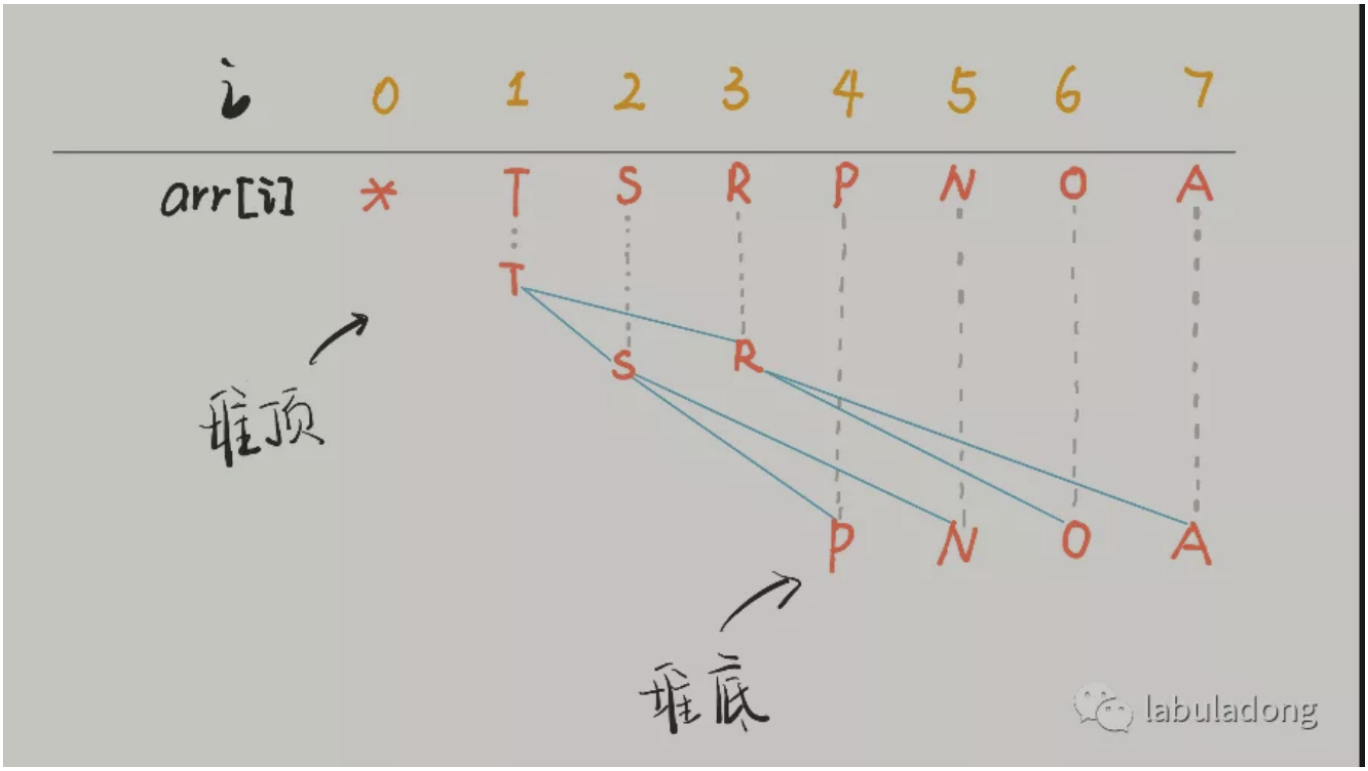
二叉堆是一颗完全二叉树。存储在数组中。

```
// 父节点的索引
int parent(int root){
  return root / 2;
}

// 左孩子索引
int left(int root) {
  return root * 2 ;
}

// 右孩子索引
int right(int root) {
  return root * 2 + 1;
}
```

如图：



二叉堆分为大根堆和小根堆，大根堆中每个节点都大于等于其子节点；小根堆中每个节点都小于等于其子节点。

优先级队列概览

```
#include <vector>

// 优先级队列
```

```
class MaxHeap {
public:
  MaxHeap(int cap) {
    _pq = std::vector<int>(cap + 1, 0);
  }

  // 获取最大值
  int max() const {
    return _pq[1];
  }

  // 插入元素
  void insert(int e) {
    _pq[++_count] = e;
    swim(_pq[_count]);
  }

  // 删除并返回当前队列中最大元素
  int delMax() {
    int top = _pq[1];
    swap(1, _count);
    _pq[_count] = 0;
    _count--;
    sink(1);
    return top;
  }

  // 上浮第k个元素，维护最大堆性质
  void swim(int k) {
    while (k > 1 && less(parent(k), k)) {
      swap(parent(k), k);
      k = parent(k);
    }
  }

  // 下沉第k各元素，维护最大堆性质
  void sink(int k) {
    // 下沉到堆底，不再操作
    while (left(k) <= _count) {
      // 假设左边节点比较大
      int older = left(k);
      if (right(k) <= _count && less(older, right(k))) {
        older = right(k);
      }

      // 两边节点都小于根节点，跳出循环
      if (less(older, k)) {
        break;
      }

      swap(older, k);
      k = older;
    }
  }
```

```
  void swap(int i, int j) {
    std::swap(_pq[i], _pq[j]);
  }

  bool less(int i, int j) {
    return _pq[i] < _pq[j];
  }

  // 父节点的索引
  int parent(int root) {
    return root / 2;
  }

  // 左孩子索引
  int left(int root) {
    return root * 2;
  }

  // 右孩子索引
  int right(int root) {
    return root * 2 + 1;
  }

private:
  int              _count;  // 当前堆中元素
  std::vector<int> _pq;     // 数组元素
};
```