3.列表

接口与实现

百只骆驼绕山走，九十八只在山后

尾驼露尾不见头，头驼露头出山沟

邓俊辉

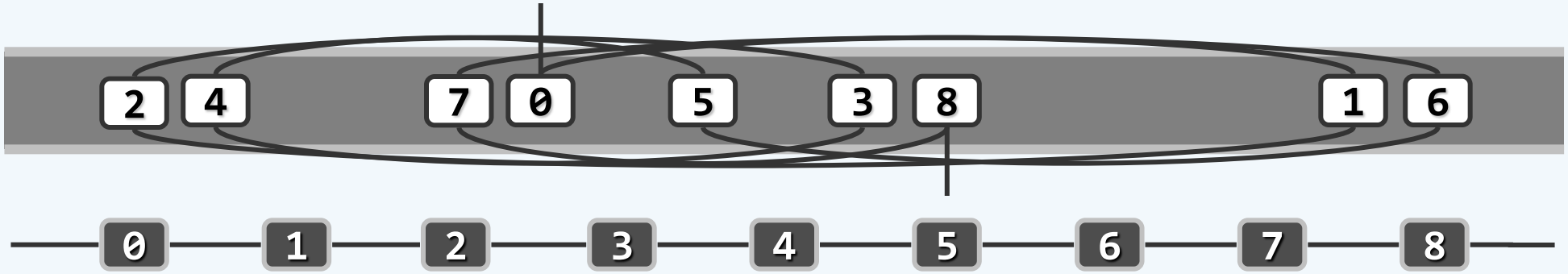
deng@tsinghua.edu.cn

列表节点：ADT接口

❖ 作为列表的基本元素，列表节点首先需要独立地“封装”实现

为此，可设置并约定若干基本的操作接口

操作	功能
pred()	当前节点前驱节点的位置
succ()	当前节点后继节点的位置
data()	当前节点所存数据对象
insertAsPred(e)	插入前驱节点，存入被引用对象e，返回新节点位置
insertAsSucc(e)	插入后继节点，存入被引用对象e，返回新节点位置



列表节点：模板类

❖ #define Posi(T) ListNode<T>* //列表节点位置（ISO C++.0x, template alias）

❖ template <typename T> //简洁起见，完全开放而不再过度封装

```
struct ListNode { //列表节点模板类（以双向链表形式实现）
   T data; //数值
   Posi(T) pred; //前驱
   Posi(T) succ; //后继
   ListNode() {} //针对header和trailer的构造
   ListNode(T e, Posi(T) p = NULL, Posi(T) s = NULL)
      : data(e), pred(p), succ(s) {} //默认构造器
   Posi(T) insertAsPred(T const& e); //前插入
   Posi(T) insertAsSucc(T const& e); //后插入
};
```

列表：ADT接口

操作接口	功能	适用对象
size()	报告列表当前的规模（节点总数）	列表
first(), last()	返回首、末节点的位置	列表
insertAsFirst(e), insertAsLast(e)	将e当作首、末节点插入	列表
insertA(p, e), insertB(p, e)	将e当作节点p的直接后继、前驱插入	列表
remove(p)	删除位置p处的节点，返回其引用	列表
disordered()	判断所有节点是否已按非降序排列	列表
sort()	调整各节点的位置，使之按非降序排列	列表
find(e)	查找目标元素e，失败时返回NULL	列表
search(e)	查找e，返回不大于e且秩最大的节点	有序列表
deduplicate(), uniquify()	剔除重复节点	列表/有序列表
traverse()	遍历列表	列表

列表：模板类

```
#include "listNode.h" //引入列表节点类

template <typename T> class List { //列表模板类
private:    int _size; //规模
            Posi(T) header; Posi(T) trailer; //头、尾哨兵
protected: /* ... 内部函数 */
public:    /* ... 构造函数、析构函数、只读接口、可写接口、遍历接口 */
};
```

- 等效地，头、首、末、尾节点的秩可分别理解为-1、0、n-1、n

构造

```
❖template <typename T> void List<T>::init() { //初始化，创建列表对象时统一调用
    header  = new ListNode<T>; //创建头哨兵节点
    trailer = new ListNode<T>; //创建尾哨兵节点
    header->succ = trailer; header->pred = NULL; //互联
    trailer->pred = header; trailer->succ = NULL; //互联
    _size = 0; //记录规模
}
```