数据结构教程第九课循环链表与双向链表 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/138/2021_2022__E6_95_B0_E6_8D_AE_E7_BB_93_E6_c98_138121.htm 本课主题： 循环链表与双向链表 教学目的： 掌握循环链表的概念，掌握双向链表的的表示与实现 教学重点： 双向链表的表示与实现 教学难点： 双向链表的存储表示 授课内容： 一、复习线性链表的存储结构 二、循环链表的存储结构 循环链表是加一种形式的链式存储结构。它的特点是表中最后一个结点的指针域指向头结点。 循环链表的操作和线性链表基本一致，差别仅在于算法中的循环条件不是p或p->next是否为空，而是它们是否等于头指针。 三、双向链表的存储结构 提问：单向链表的缺点是什么？ 提示：如何寻找结点的直接前趋。 双向链表可以克服单链表的单向性的缺点。 在双向链表的结点中有两个指针域，其一指向直接后继，另一指向直接前趋。 1、线性表的双向链表存储结构 typedef struct DulNode{ struct DulNode *prior. ElemType data. struct DulNode *next. }DulNode,*DuLinkList. 对指向双向链表任一结点的指针d，有下面的关系： d->next->priou=d->priou->next=d 即：当前结点后继的前趋是自身，当前结点前趋的后继也是自身。 2、双向链表的删除操作 Status ListDelete_DuL(DuLinkList amp.e){ if(!(p=GetElemP_DuL(L,i))) return ERROR. e=p->data. p->prior->next=p->next. p->next->prior=p->pror. free(p). return OK. }//ListDelete_DuL 3、双向链表的插入操作 Status ListInsert_DuL(DuLinkList amp.e){ if(!(p=GetElemP_DuL(L,i))) return ERROR. if(!(s=(DuLinkList)malloc(sizeof(DuLNode))))

return ERROR. s->data=e. s->prior=p->prior. p->prior->next=s. s->next=p. p->prior=s. return OK. }//ListInsert_DuL 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com