石家莊鐵道大學
SHIJIAZHUANG TIEDAO UNIVERSITY

在线开放课程

线性表

循环链表、双向链表

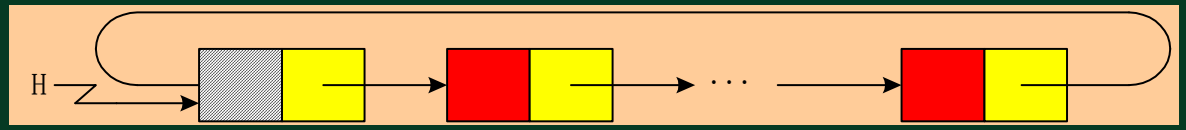
主讲 ： 刘辉

目录

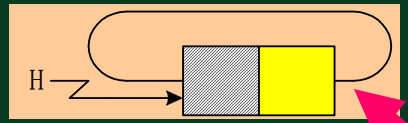
- 1 循环链表及其操作
- 2 双向链表及其操作

一、循环链表及其操作

- **循环链表（Circular Linked List）**
  - **特点：表中最后一个结点的指针域指向头结点，整个链表形成一个环**



（a）非空单循环链表



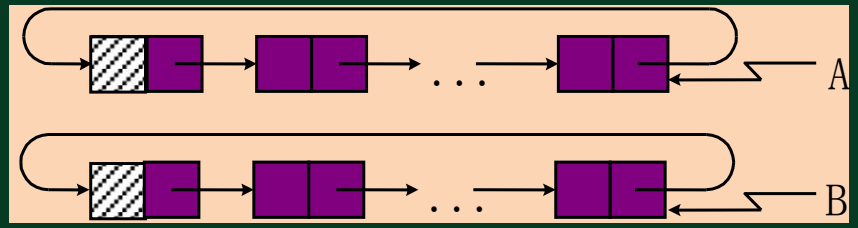
（b）空表

H->next=H

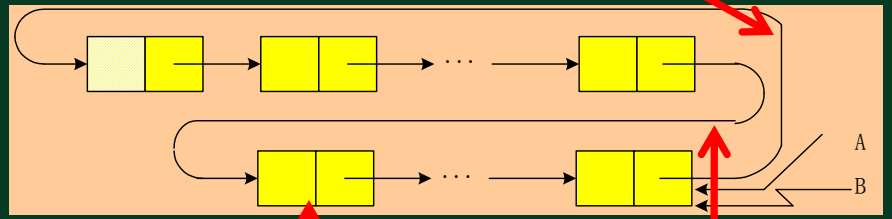
为了使某些操作实现起来方便，在循环单链表中也可设置一个头结点。 这样，空循环链表仅由一个自成循环的头结点表示。

一、循环链表及其操作

- **循环链表的基本操作—合并**



B->next=A->next;

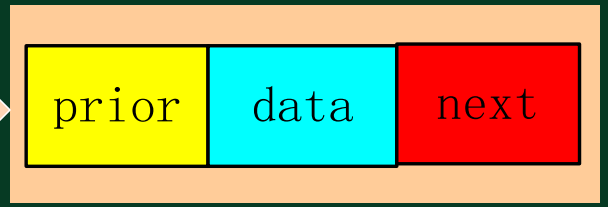


p=B->next->next;

A->next=p

二、双向链表及其操作

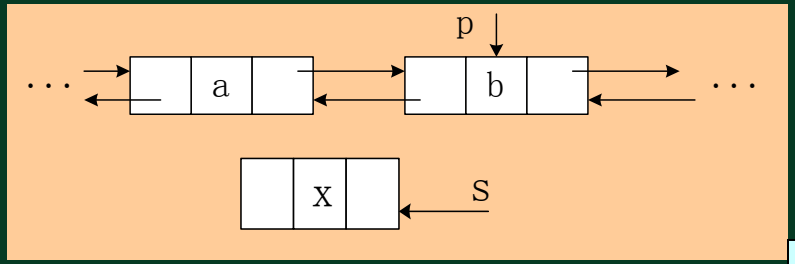
- **双向链表**
  - 结点中有两个指针域

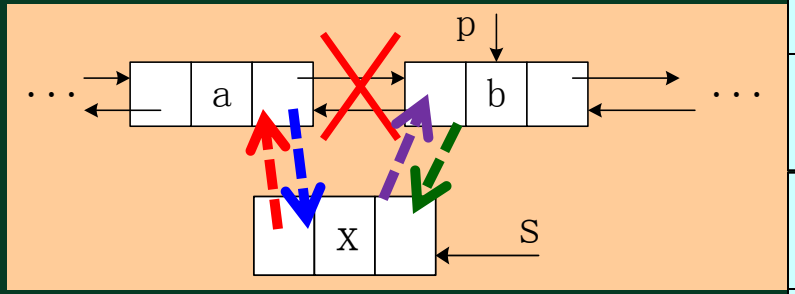



```
typedef struct DuLNode{
    ElemType   data;
    struct DuLNode   *prior;
    struct DuLNode   *next;
} DuLNode, *DuLinkList
```

二、双向链表及其操作

双向链表的基本操作—插入






```
s->prior=p->prior;
p->prior->next=s;
s->next=p;
p->prior=s;
```

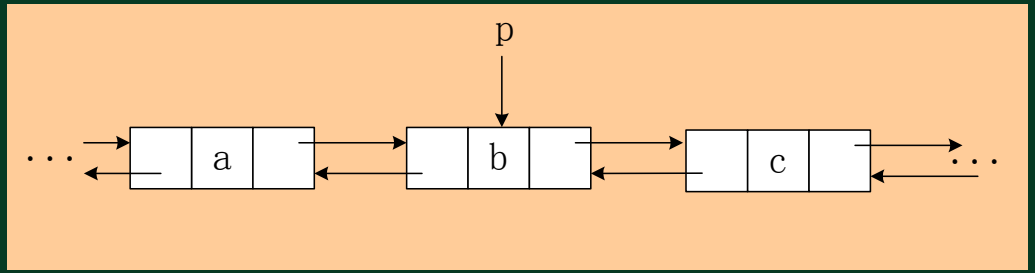
二、双向链表及其操作

双向链表的基本操作—插入

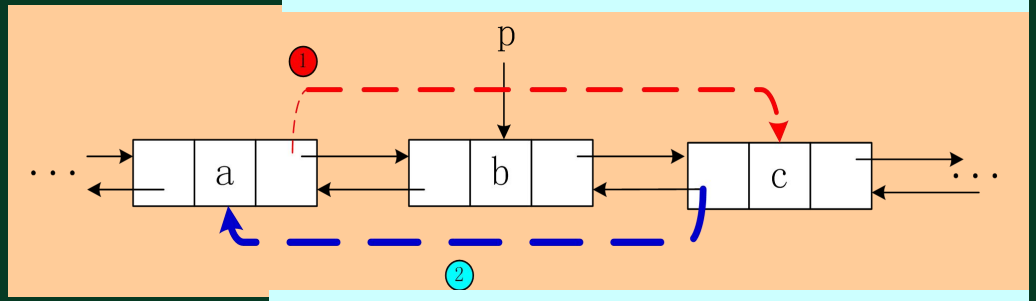
```
Status ListInsert_DuL(DuLinkList &L,int i,ElemType e){
    if(!(p=GetElemP_DuL(L,i))) return ERROR;
    s=new DuLNode;
    s->data=e;
    s->prior=p->prior;
    p->prior->next=s;
    s->next=p;
    p->prior=s;
    return OK;
}
```

二、双向链表及其操作

双向链表的基本操作—删除



1. p->prior->next=p->next;



2. p->next->prior=p->prior;

二、双向链表及其操作

双向链表的基本操作—删除

```
Status ListDelete_DuL(DuLinkList &L,int i,ElemType &e){
    if(!(p=GetElemP_DuL(L,i)))    return ERROR;
    e=p->data;
    p->prior->next=p->next;
    p->next->prior=p->prior;
    delete p;
    return OK;
}
```

小结

- 了解循环链表的实现和合并算法的实现
- 了解双链表的实现
- 理解双链表的插入和删除算法

谢谢！