LISTS

Doubly Linked List

Move in forward and backward direction.

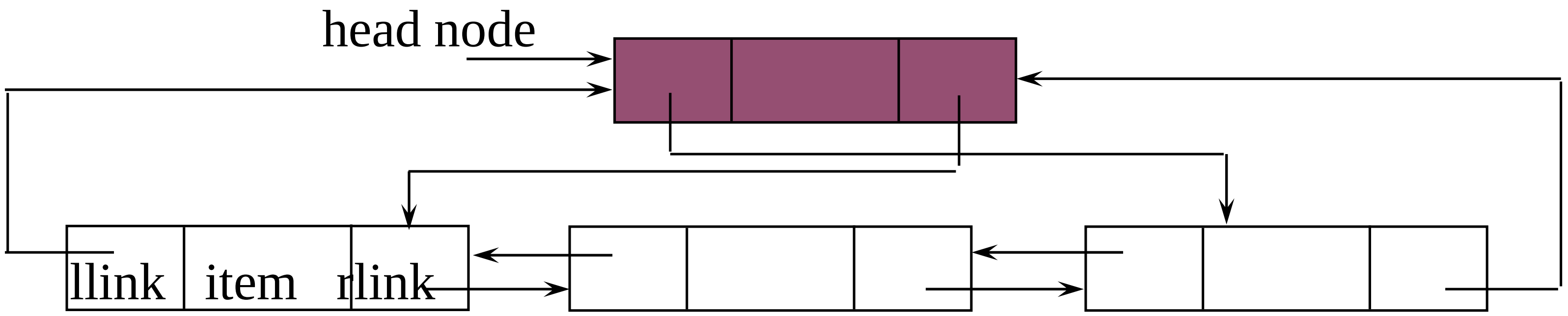
Singly linked list (*in one direction only*)
How to get the preceding node during deletion or insertion?
Using 2 pointers

Node in doubly linked list consists of:
1. *left link field* (llink)
2. *data field* (item)
3. *right link field* (rlink)

Doubly Linked Lists

```
typedef struct node *node_pointer;
typedef struct node {
    node_pointer llink;
    element item;
    node_pointer rlink;
}
```

ptr
= ptr->rlink->llink
= ptr->llink->rlink



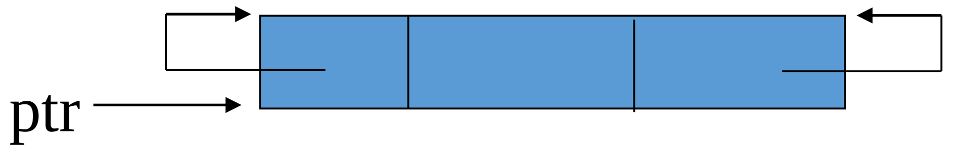


***Figure 4.24:**Empty doubly linked circular list with head node (p.180)

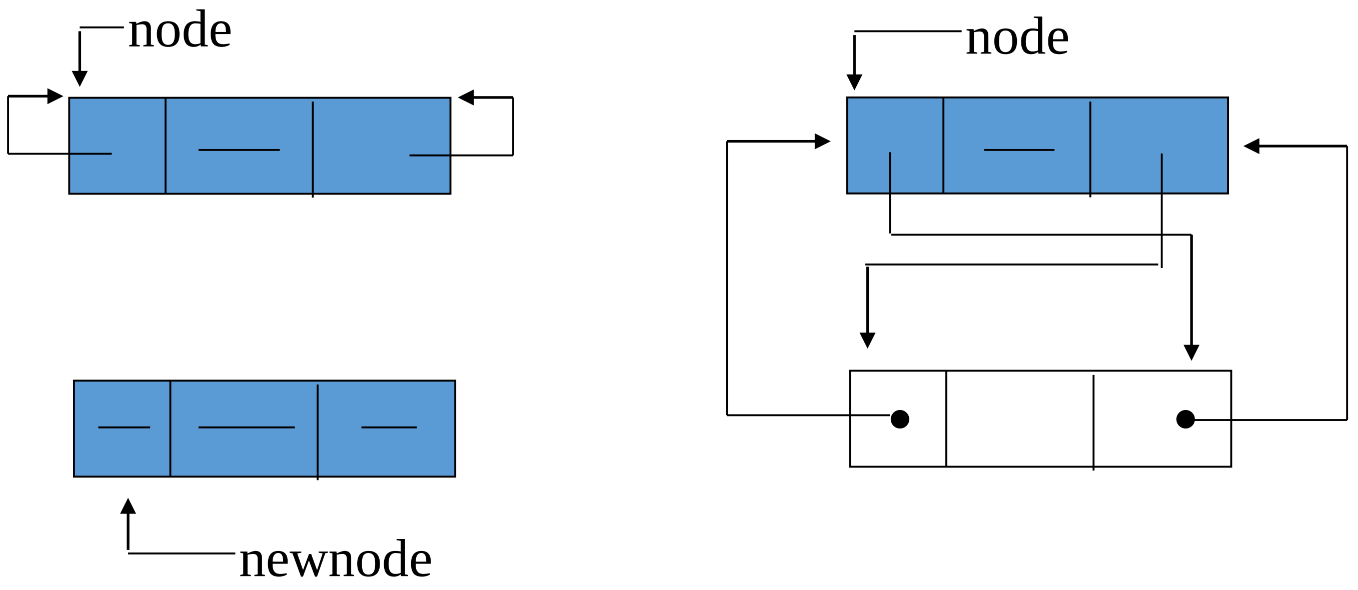
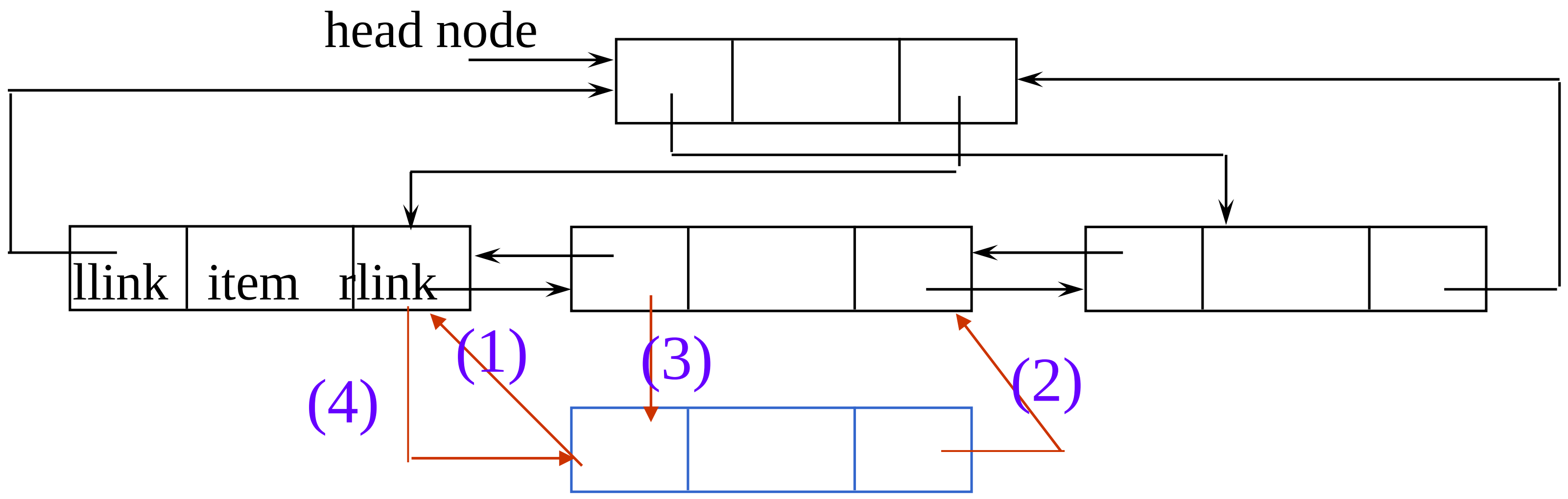


***Figure 4.25:** Insertion into an empty doubly linked circular list (p.181)

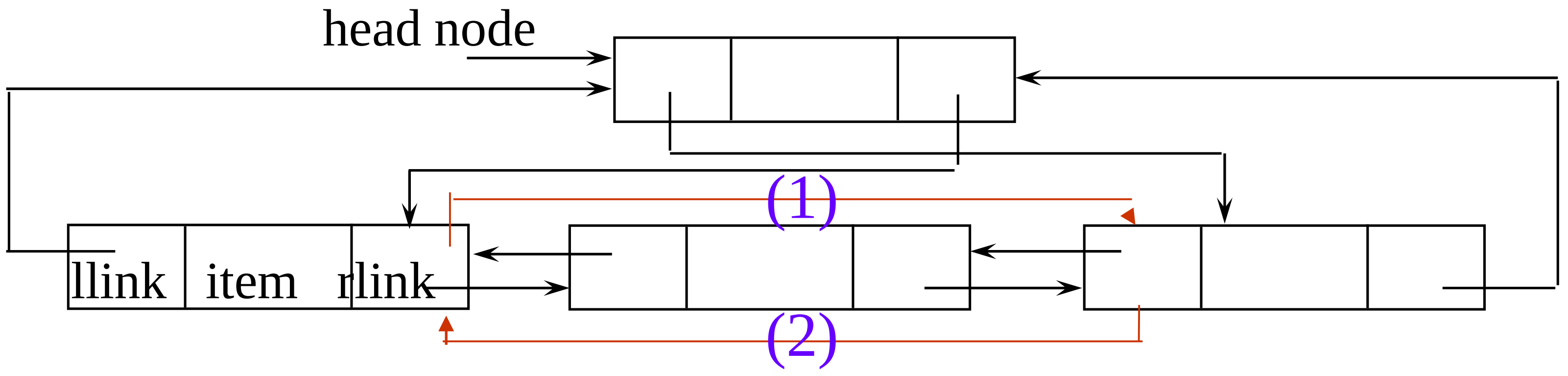
Insert

```
void dinsert(node_pointer node, node_pointer newnode)
{
      (1) newnode->llink = node;
      (2) newnode->rlink = node->rlink;
      (3) node->rlink->llink = newnode;
      (4) node->rlink = newnode;
}
```



Delete

```
void ddelete(node_pointer node, node_pointer deleted)
{
    if (node==deleted) printf("Deletion of head node
                              not permitted.\n");
    else {
        (1) deleted->llink->rlink= deleted->rlink;
        (2) deleted->rlink->llink= deleted->llink;
            free(deleted);
    }
}
```


head node
(1)
llink
item
rlink
(2)