Doubly Linked Lists

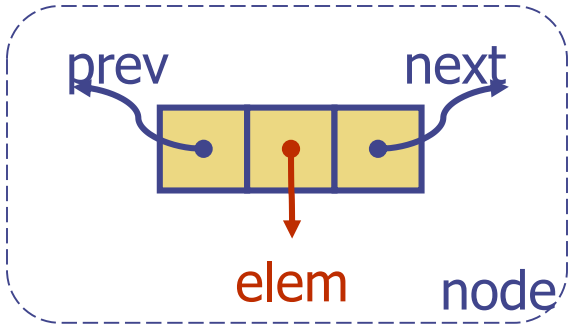
Jyh-Shing Roger Jang (張智星)
CSIE Dept, National Taiwan University

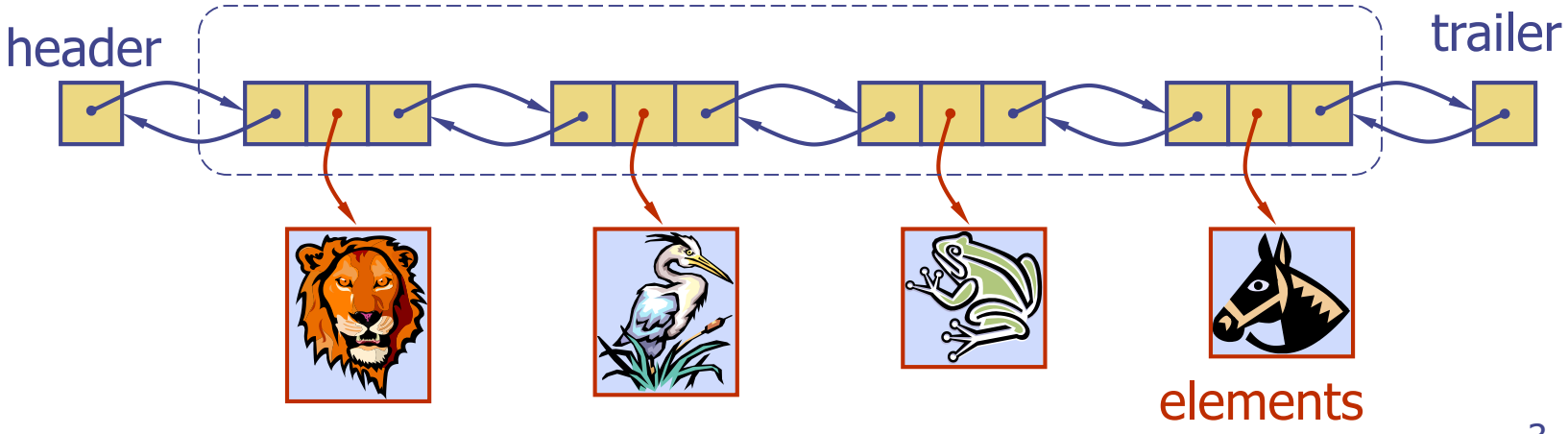
Why Doubly Linked Lists

- Why?
  - To facilitate efficient insertion and removal at any positions
- How?
  - By having two links pointing to the previous and the next nodes, respectively

Doubly Linked List

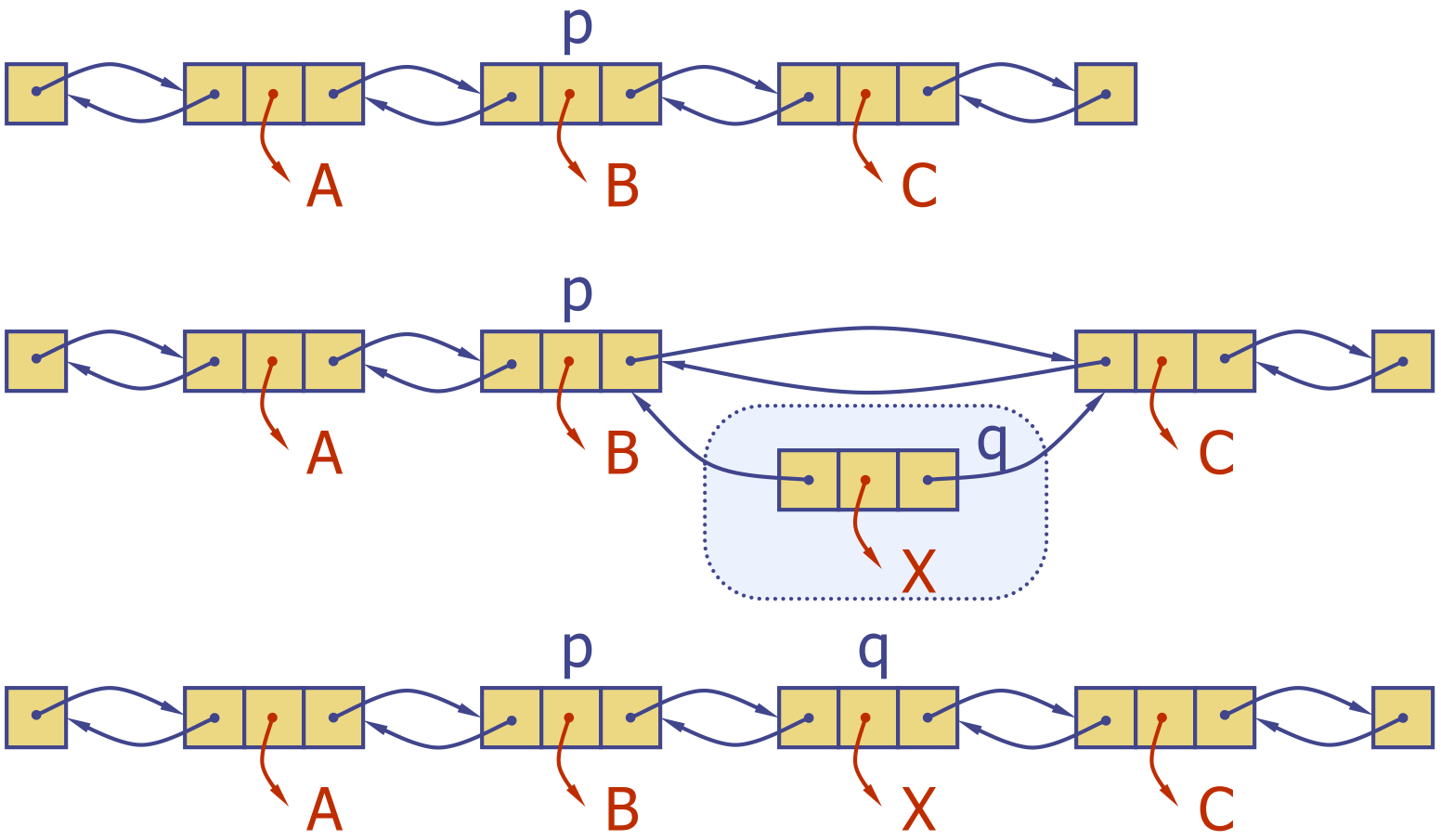
- A doubly linked list (DLL) allows us to traverse the list in either directions quickly.
- A node has the following fields
  - element
  - link to the previous node
  - link to the next node
- Special trailer and header nodes





Insertion after a Node

- Operations of insertAfter(p, X)



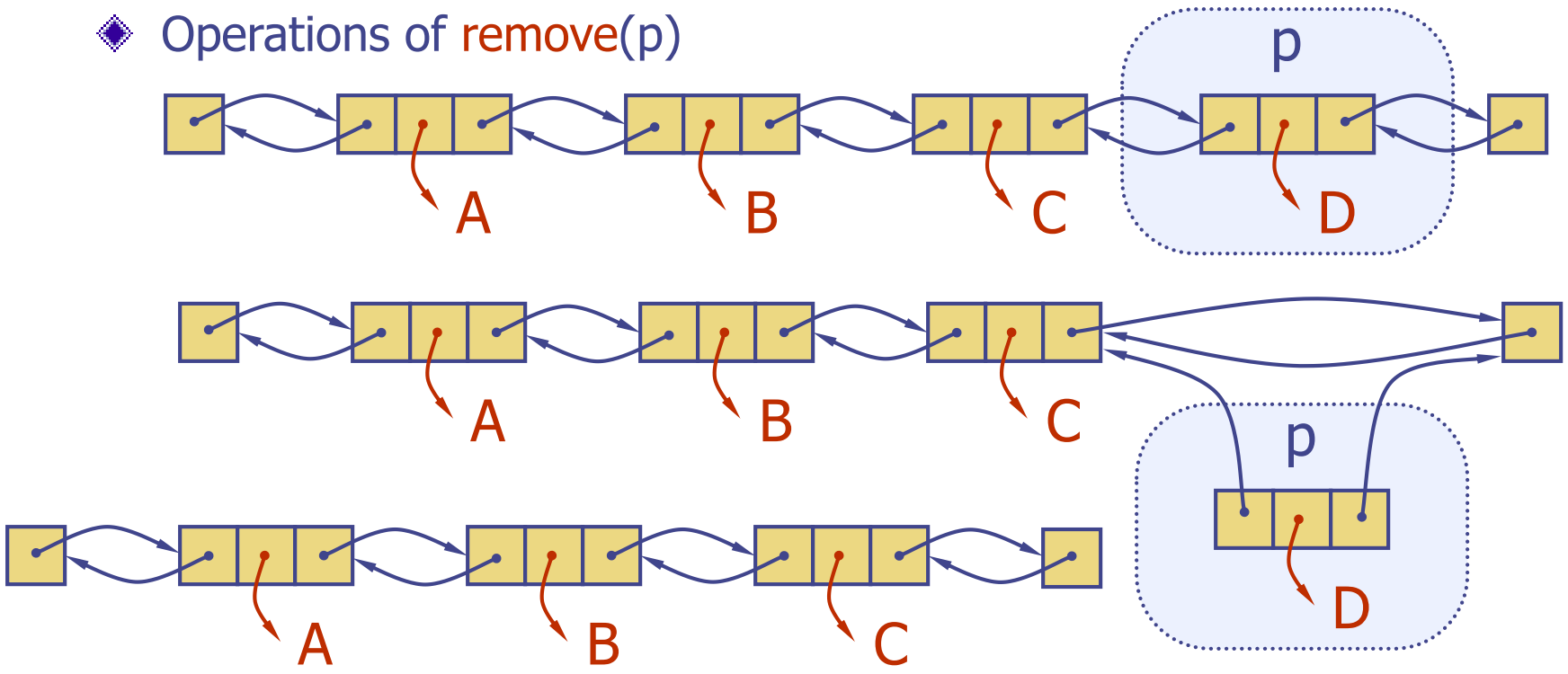
Insertion before a Node

- Operations of insertBefore(v, e)

```
                                                    // insert new node before v
void DLinkedList::add(DNode* v, const Elem& e) {
  DNode* u = new DNode; u->elem = e; // create a new node for e
  u->next = v;                      // link u in between v
  u->prev = v->prev;                // ...and v->prev
  v->prev->next = v->prev = u;
}
```

Deletion

- Operations of remove(p)




```
void DLinkedList::remove(DNode* v) {      // remove node v
  DNode* u = v->prev;                     // predecessor
  DNode* w = v->next;                     // successor
  u->next = w;                            // unlink v from list
  w->prev = u;
  delete v;
}
```

Example of Generic DLL

- By using generic DLL, you can put any data of any types into the data part of a node.
  - Example