CS1007 lecture #24 notes

tue 30 apr 2002

- news
- data structures
- reading: ch 12

news.

- Please complete an on-line course evaluation. See the NEWS page for the link.
- homework #6 due Thu May 2
- final exam:
  - Tue May 14 9am-12noon (AM class, section 002)
  - Thu May 16 1pm-4pm (PM class, section 001)

data structures.

- a *data structure* is essentially an *abstract data type*
- it maps a virtual model of data to a real data type
- like an array or a Vector or a class
- there are several classic data structures in computer science
- we'll look at a few:
  - linked list
  - doubly linked list
  - queue
  - stack
- each has rules about how to *add* and *remove* elements
- examples:
  - a queue is also called FIFO — first in, first out
  - a stack is also called LIFO — last in, first out.

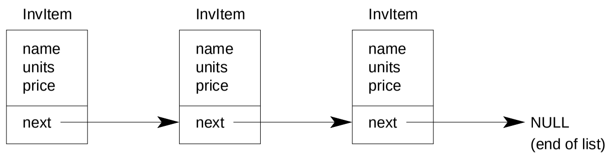
implementation vs interface.

- when talking about data structures, there is a distinction between *implementation* and *interface*
- implementation — refers to the actual underlying implementation; like *linked list* or *doubly linked list*
- interface — refers to an abstract view of the data structure, overlying the implementation; like *queue* or *stack*
- for example, a *queue* can be implemented using a *linked list*
- the interfaces govern how items can be added and removed from the data structure

linked list.

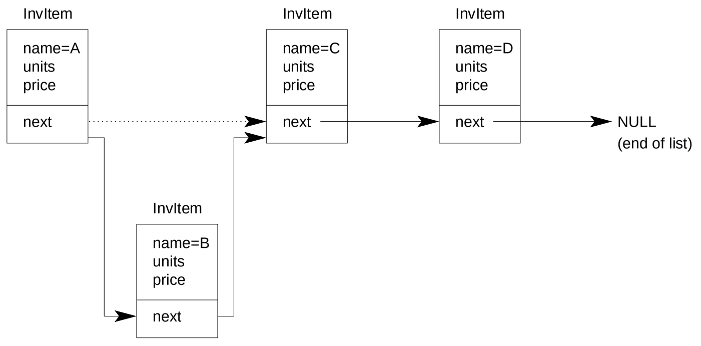
- a *linked list* chains instances of a class together using a field called “next”, which points to the next instance of the class in the chain
- yes, this looks like another type of array or Vector

```
public class InvItem {
  private String name;
  private int units;
  private float price;
  InvItem next; // points to the next InvItem in the chain
}
```



adding an item to a linked list (1).

- to add an item to a linked list, you simply instantiate one instance of the InvItem and then set the “next” fields in the new item and the item in the linked list after which the new item is being inserted:



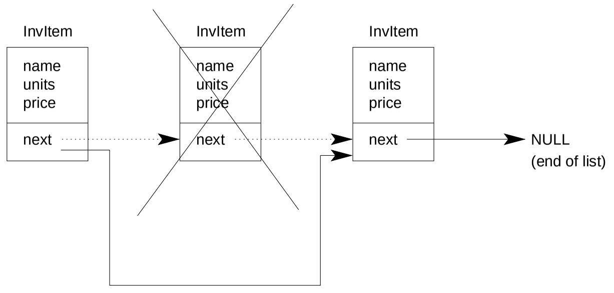
adding an item to a linked list (2).

- here’s a code fragment:

```
// top-level declaration
InvItem linkedList;
.
.
.
// somewhere inside a method...
InvItem newItem = new InvItem( name,units,price );
InvItem listItem; // pointer to list item after which
                  // newItem will be inserted
newItem.next = listItem.next;
listItem.next = newItem;
.
.
.
```

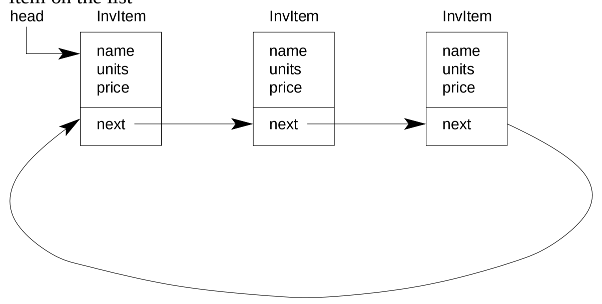
removing an item from a linked list.

- to remove an item to a linked list, you simply move the “next” field pointers around



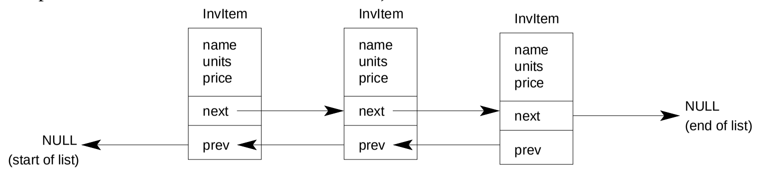
circular linked list.

- sometimes linked lists are *circular*, where the “next” field from the last item points back to the first item
- in this case, there is typically an external “pointer” called “head” which points to the first item on the list



doubly linked list (1).

- a *doubly linked list* chains instances of a class together using two fields called “next” (which points to the next instance of the class in the chain) and “prev” (which points to the previous instance of the class in the chain).



doubly linked list (2).

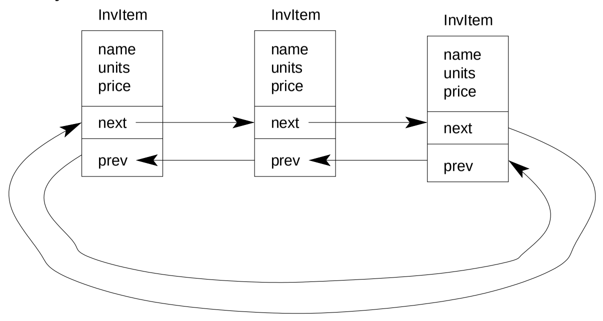
- for example:

```
public class InvItem {
  private String name;
  private int units;
  private float price;
  InvItem next; // points to the next InvItem in the chain
  InvItem prev; // points to the previous InvItem in the chain
}
```

- when adding and removing items to a doubly linked list, you need to set the “next” and “prev” fields, just as like with the (singly) linked list

circular doubly linked list.

- doubly linked lists can also be circular:



queue.

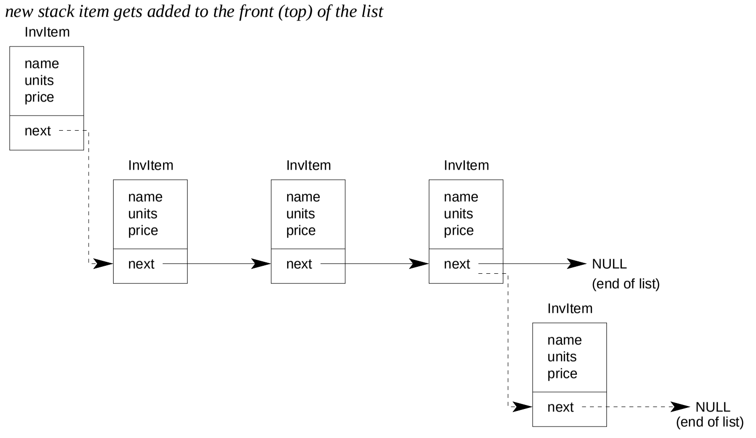
- a *queue* is like a check-out line at a store
- you can only add items (people) to the end of the line
- you can only remove items (people) from the front of the line
- hence, a queue is also called a FIFO (first in, first out)
- following are the typical names for queue routines:
  - **enqueue**: for adding items to a queue
  - **dequeue**: for removing items from a queue

stack.

- a *stack* is like a stack of plates
- you can only add items (plates) to the top of the stack
- you can only remove items (plates) from the top of the stack
- hence, a stack is also called a LIFO (last in, first out)
- following are the typical names for stack routines:
  - **push**: for adding items to a stack
  - **pop**: for removing items from a stack

stack vs queue: adding items.

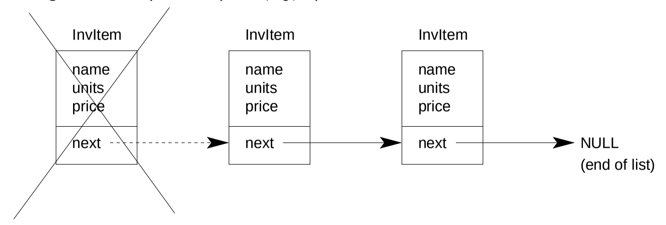
new stack item gets added to the front (top) of the list



new queue item gets added to the end of the list

stack vs queue: removing items.

stack items get removed from the front (top) of the list



queue items get removed from the front of the list