Queues
Lecture 26
Sections 8.1 - 8.3

Robb T. Koether

Hampden-Sydney College

Wed, Mar 25, 2009
Outline

1. Queues
2. The Queue Interface
3. Queue Implementation
4. Assignment
A queue is a List that operates under the principle “first in, first out” (FIFO). New elements are enqueued into the queue. Old elements are dequeued from the queue.

To enforce the FIFO principle, we enqueue and dequeue at opposite ends.
Implementation of Queues

- Which is more accurate?
  - A queue *is* a list.
  - A queue *has* a list.

- Use `pushFront()` and `popBack()`, or
- Use `pushBack()` and `popFront()`.

- Choose a List class for which enqueuing and dequeuing will be efficient.
Choose an appropriate List class as a base class.

Which are good choices?

- ArrayList
- CircArrayList
- LinkedList
- LinkedListwTail
- DoublyLinkedList
- CircLinkedList
Queue Constructors

- `Queue();` Constructs an empty queue.
- `Queue(const Queue& q);` Constructs a copy of the specified queue.
Inspectors

- `T head() const;`
  Gets a copy of the element at the head of the queue.

- `int size() const;`
  Gets the number of elements in the queue.

- `bool isEmpty() const;`
  Determines whether the queue is empty.
Mutators

- **void enqueue(const T& value);**
  Enqueues the specified value at the tail of the queue.

- **T dequeue();**
  Dequeues and returns the element at the head of the queue.

- **void makeEmpty();**
  Makes the queue empty.
Facilitators

- **void input(istream& in);**
  Reads a queue from the specified input stream.

- **void output(ostream& out) const;**
  Writes a queue to the specified output stream.
Other Member Functions

- **void isValid() const;**
  Determines whether the queue has a valid structure.
Non-Member Functions

- `istream& operator>>(istream& in, Queue& q);`  
  Reads a queue from the specified input stream.

- `ostream& operator<<(ostream& out, const Queue& q);`  
  Writes a queue to the specified output stream.
Input and Output

- Are there complications in using the List class `input()` and `output()` functions?
- Will the interpretation of head and tail be reversed between the `ArrayQueue` and the `LinkedQueue`?
- If so, then we might need to rewrite the functions for one of the two Queue classes.
Queue Implementation

- arrayqueue.h
- circarraylist.h
- linkedqueue.h
- linkedlistwttail.h
- linkedlistnode.h
- QueueTest.cpp
Assignment

Homework

- Read Section 8.1, pages 389 - 399.
- Read Section 8.2, pages 400 - 409.
- Read Section 8.3, pages 412 - 423.