Initial "UML" for **queue** class (revised 1-27-05)

adapted from Ch. 8, Savitch and Main, "Data Structures and Other Objects Using C++"

Template Class: `queue`

/* a collection of items such that entries can be inserted at one end (called the **rear**) and removed at the other end (called the **front**). */

Member data and related details:

* contains elements of type **`value_type`**; this is set to be the value of template parameter **Item**

Constructors:

/* postcondition: creates an empty **queue** instance */

```
queue( );
```

Accessors and other constant member functions:

/* postcondition: returns **true** if queue is empty, and returns **false** otherwise */

```
bool     is_empty( ) const;
```

/* precondition: is_empty() == false */
/* postcondition: returns the value of the front item of the queue, BUT the queue is unchanged. */

```
Item     get_front( ) const;
```

Modifiers and other modifying member functions:

/* postcondition: a new copy of **entry** has been inserted at the **rear** of the queue */

```
void     enqueue(const Item& entry);
```

/* precondition: is_empty() == false */
/* postcondition: the front item of the queue has been removed, and a reference to it is returned */

```
Item&    dequeue( );
```