```
public interface ShoppingCart {
  /// <summary>
  /// Add this many of this item to the
  /// shopping cart.
  /// </summary>
  /// <exception cref="ArgumentOutOfRangeException">
  /// </exception>
  void AddItems(Item anItem, int quantity);

  /// <summary>
  /// Delete this many of this item from the
  /// shopping cart
  /// </summary>
  /// <exception cref="ArgumentOutOfRangeException">
  /// </exception>
  /// <exception cref="NoSuchItemException">
  /// </exception>
  void DeleteItems(Item anItem, int quantity);

  /// <summary>
  /// Count of all items in the cart
  /// (that is, all items x qty each)
  /// </summary>
  int ItemCount { get; }

  /// Return iterator of all items
  IEnumerable GetEnumerator();
}
```

ShoppingCart.cs

Answer on 214

3. **A fax scheduler.** This code will send faxes from a specified filename to a U.S. phone number. There is a validation requirement; a U.S. phone number with area code must be of the form *xnn-nnn-nnnn*, where *x* must be a digit in the range `[2..9]` and *n* can be `[0..9]`. The following blocks are reserved and are not currently valid area codes: *x*11, *x*9*n*, 37*n*, 96*n*.

   The method's signature is as follows:

```
///
/// Send the named file as a fax to the
/// given phone number.
/// <exception cref="MissingOrBadFileException">
/// </exception>
/// <exception cref="PhoneFormatException">
/// </exception>
/// <exception cref="PhoneAreaCodeException">
/// </exception>
public bool SendFax(String phone, String filename)
```

   Given these requirements, what tests for boundary conditions can you think of?

Answer on 215

4. **An automatic sewing machine that does embroidery.** The class that controls it takes a few basic commands. The co-

x and *y* increase as you move toward the upper-right corner, whose coordinates are `x = TableSize.Width - 1` and `y = TableSize.Height - 1`.

Coordinates are specified in fractions of centimeters.

```
public void MoveTo(double x, double y);
public void SewTo(double x, double y);
public void SetWorkpieceSize(double width,
                             double height);
public Size WorkpieceSize { get; }
public Size TableSize { get; }
```

Some real-world constraints might be interesting: you can't sew thin air, of course, and you can't sew a workpiece bigger than the machine.

Given these requirements, what boundary conditions can you think of?

5. **Audio/video-editing transport.** A class that provides methods to control a DVD or media player. There's the notion of a "current position" that lies somewhere between the beginning (historically, BOT for "beginning of tape") and the end (EOT).

Answer on 216

You can ask for the current position and move from there to another given position. *Fast-forward* moves from current position toward the EOT by some amount. *Rewind* moves from current position toward the BOT by some amount. When media is first loaded, it is positioned at BOT automatically.

```
using System;
public interface AVTransport {
  /// Move the current position ahead by this many
  /// seconds. Fast-forwarding past EOT
  /// leaves the position at EOT
  void FastForward(double seconds);

  /// Move the current position backwards by this
  /// many seconds. Rewinding past zero leaves
  /// the position at zero
  void Rewind(double seconds);

  /// Return current time position in seconds
  double CurrentTimePosition();

  /// Mark the current time position with label
  void MarkTimePosition(String name);

  /// Change the current position to the one
  /// associated with the marked name
  void GotoMark(String name);
}
```

AVTransport.cs