The Solitaire Cipher—Part I

Solitaire, designed by cryptologist Bruce Schneier, was created to be the first truly secure hand cipher, requiring only a deck of cards for encryption and decryption of messages. In this lab, you will write the first step in encoding a message. In next week’s lab you will continue with the encryption.

Step One of Encryption

1. Given a message, discard any characters that aren’t letters;
2. Split the message into five character groups;
3. Use the character ‘X’ to pad the last group, if needed.

For example, the message string “Code in Java, live longer!” would then become

CODEINJAVA LIVELONGER

You will likely want to create a class called Message that stores the original string, implements a parsing method (with the name of your choosing) and implements the toString method which returns a String containing the altered form of the message (in 5 character blocks) as in the above example. The class members (methods and data) are otherwise up to you (though keep it simple; you shouldn’t require much for this initial task).

The Deck of Cards

Solitaire needs a full deck of 52 cards and two jokers, ‘A’ and ‘B’. Each card in the deck is assigned a value from 1 to 52. This can later be mapped to the actual cards, where a card’s value starts with the face value, Ace = 1, 2 = 2 ... Jack = 11, Queen = 12, King = 13, with a value added depending on its suit: Clubs add 0, Diamonds add 13, Hearts add 26 and Spades add 39.

Create a class called Deck that creates, and stores, a deck of cards. It is unsorted by default, so its initial state should be:

1 2 3 ... 52 A B

Eventually we will key the deck, so you can expect this order to change, i.e. we will be taking cards from one deck location and inserting them in another. The size of the deck will remain constant. Consider then what would be the best collection to use to store your deck? Notice your deck contains BOTH integers and letters. What then would be the best data type for your collection?