Homework #3: Functions and Closures
CSC 358/458, Prof. Robin Burke
Spring 2006
Due: 4/17/2006

Objective:
Gain experience using functional objects in Lisp.

What to do:
Download cards.lsp and eights.lsp from the course web site. Load eights.lsp into your Lisp environment. You may need to edit the path for cards.lsp so that it loads. Finish eights.lsp by writing the following Lisp code:

1. Currently, the game structure is stored in a global variable called *game*. Change the code to get rid of *game* and replace it with a lexical variable game, using let. The closure should include: new-game, deal-hands, upcard-suit, two-pile, set-two-played, discard, draw, hand, set-hand, and the defsetf. You will need to define two new functions: upcard (that returns the upcard), and hands (that returns the list of hands), since the structure will not longer be accessible.

2. Rewrite draw as a recursive function.

3. Write a function playable-card that takes a hand and an upcard and returns a list of all the playable cards in the hand. This function must use one of the list processing functions and an anonymous lambda expression. It cannot use a loop.

4. Write a function move-selector that, given a hand, returns a card to play. It should call playable-card from step 3 and then select randomly from the cards returned. If there are no cards playable, it should return nil. If it selects an eight, have it pick a suit randomly.

5. Write a function two-move-selector that, given a hand, returns a card to play. The only legal play is a two, so the function should return one of the twos in the hand if there are any.

6. You should now be able to play a game of Crazy Eights against the computer. It will play badly (plus you get to see its hand.) Start by calling (crazy-eights 2) or if you want more computer players (crazy-eights 4).

7. (Extra credit) Write a more sophisticated move selector (and two-move selector) for Crazy Eights. Your function should have a subfunction that scores each card for desirability and then selects the one most preferred. It is up to you to consider how this should work, but you should not look at your opponent's hands except to determine how many cards they hold. (That would be cheating!)

What to turn in:
Upload your .lsp file to the DLWeb website for the course. All assignments are due BEFORE CLASS TIME on the due date. We will discuss the homework in class, so there can be no late assignments.

Hints and Notes:
- Rules of Crazy Eights:
  The dealer deals eight cards to each player. The undealt stock is placed face down on the table, and the top card of the stock is turned face up and placed beside the stock to start the discard pile.

  Starting with the player to dealer's left, and continuing clockwise, each player in turn must either play a legal card face up on top of the discard pile, or draw a card from the undealt stock. The object of the game is to get rid of all of your cards. The following plays are legal:

  - if the top card of the discard pile is not an eight, you may play any card which matches the rank or suit of the previous card (for example if the top card was the king of hearts you could play any king or any heart);
  - an eight may be played on any card, and the player of the eight must nominate a suit, which must be played next;
• if an eight is on top of the pile, you may play any card of the suit nominated by the person who played the eight.
• if a two is played, the next player must either draw two cards or play another two. If several consecutive twos have been played the next player must either play another two or draw two cards for each two in the sequence.
• You may find it useful to have your deck be sorted initially. This creates a predictable set of cards and choices and makes debugging easier.
• If your program gets hung up in the user input loop (it sometimes happens if you evaluate an expression while it is in this mode), go to "View -> Processes" and "Abort" the Listener process.
• You will want to use the random function to choose cards randomly. Examine its usage in the shuffle-deck function to see how it works.