Homework 2: LISP

This homework is due February 6, 1991, (by 5:00 PM).

Test your programs using the `lisp` command on Unix, which is the interpreter described in Appendix C. Programs that use `set` and `while` will receive less credit than programs that do not use these features. Put your LISP code for the standard problems in `hw2.l`. Hand in a print-out of the code, and please put your name on it. (If you do extra-credit problems, please write the names of the file(s) where the source code for your solutions can be found on your print-out.)

1. (6 points) Do problem 1(e) on page 60 of Kamin's book (sublists).
2. (3 points) Write a program `(append ls1 ls2)` that returns a list that contains the elements of `ls1` followed by the elements of `ls2`.
3. (9 points) Do problem 2 on page 60 (sets)
4. (20 points) Do problems 5 (a) and (b) on page 61.
5. (15 points) This problem involves showing how to represent arrays as lists. Write the following functions:
   (a) `(list->array lst)` returns the list `lst` as an array.
   (b) `(array-fetch arr n)` returns the `n`th element of `arr`. (You decide what it should do if this element doesn't exist.)
   (c) `(array-size arr)` returns the number of elements in `arr`.
   (d) `(array-update arr n x)` returns a new array that is like `arr` except that its `n`th element is the value of `x`.

   How does the efficiency of your implementation compared with the usual efficiency of arrays? What are the causes of any inefficiency?

   (extra credit only) Implement the above operations in the most efficient manner you can. Define carefully what efficiency measures you are using, and show why your programs are as efficient as possible.
6. (10 points) Write a program `(subst new old ls)` that builds a list obtained by replacing each (top-level) occurrence of the item `old` in the list `ls` with the item `new`. You may assume that an "item" is either a number or a symbol. For example, `(subst 'z 'a '(a b a c a))` returns `(z b z c z)`, `(subst 0 1 '(0 1 0 1))` returns `(0 0 0 0)`, `(subst 'dog 'cat '(my dog is fun))` returns `(my dog is fun)`, and `(subst 'two 'one '())` returns `()`.
7. (Extra credit only) Do (one or more of) problem 4 on page 60, and problems 5 (c), 5 (d), or 6 (any part) on page 61.
8. (extra credit) Program the following:
   (a) `(remove-2nd what where)` returns a the list that is like `where`, except that the second occurrence of `what` in `where` is removed.

(b) `(describe s)` returns a s-expression that, when its printed form is input to the interpreter, will yield an s-expression equal to `s`, assuming that `(quote x)` was the same as `'x` for symbols. For example: `(describe 27)` returns 27, `(describe 'x)` returns `(quote x)`, `(describe '(a b))` returns `(cons (quote a) (quote b))`.

9. (extra credit only) Do (one or more of) the problems 7, 8, or 12 in section 2.7.2 of Kamin's book. You will find the source for the LISP interpreter in

   ```
   /home/cs342/public/src/chap.2/lisp.p
   ```

   Make a copy using the Unix `cp` command and then edit your copy. Turn in differences (using the Unix command `diff`) as before.