Homework 2: LISP

You should hand in a printout of your programs. To do this, edit the programs into a file `hw2.l` on Unix. (We will want to be able to see the `hw2.l` file in your directory.)

Test your programs using the `lisp` command on Unix, which is the interpreter described in Appendix C.

Try to write programs without using `set` and `while`, as this will help you both to get the problems right and later in the course when we study functional languages.

(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. Do problem 1(e) on page 60 of Kamin's book (sublists).

2. Do problem 2 on page 60 (sets).

3. Consider the following function:

```
(define list->array (lst)
  (cons 'array lst))
```

This function turns a list into an "array", by representing the array as a list whose car is the symbol `array` and whose cdr is the original list. We will call such a list an array for the purposes of this problem. Your task is to program the following functions.

(a) `(array-get arr n)` returns the `n`th element of `arr`, and the symbol `get-error` otherwise. For example,

```
(array-get (list->array '(3 5 7 11 13)) 3)
```

returns 7.

(b) `(array-size arr)` returns the number of elements in `arr`. For example,

```
(array-size (list->array '(3 5 7 11 13)))
```

returns 5.

(c) `(array-update arr n x)` returns a new array that is like `arr` except that its `n`th element is the value of `x`. For example,

```
(array-get (array-update (list->array '(3 5 7 11 13)) 2 99) 2)
```

returns 99.

(d) `(array-add arr1 arr2)`, where `arr1` and `arr2` have the same size and contain only integers, returns a new array whose nth element is the sum of the nth elements of `arr1` and `arr2`. For example,

```
(array-add (list->array '(1 2 3)) (list->array '(4 5 6)))
```

should return an array equivalent to `(list->array '(5 7 8))`.

4. (extra credit only) How does the efficiency of the arrays in the previous problem compare with the usual efficiency of arrays? What are the causes of any inefficiency? 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.

5. 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 `()`.

6. Do problems 5 (a) and (b) on page 61.

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. The country Lolga uses the following secret code. The message, written as a LISP list, such as `(a t t a c k - a t - d a w n)`, is transformed twice, first by reversing all sequences of nonvowels (including - and other punctuation), `(a t t a - k c a d - t a n w)`, and then the entire list is reversed, in this case yielding `(w n a t - d a c k - a t t a)`. Write a LISP function `code` that both encodes and decodes messages. (The same function will do both, since the coding process is its own inverse.) The case of letters does not matter; i.e., `a` vs. `A`. The vowels are a, e, i, o, and u.

9. (extra credit only) What is the real name of the country in the previous problem?

10. (extra credit only) 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) (extra credit only) `(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))`.

11. (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.