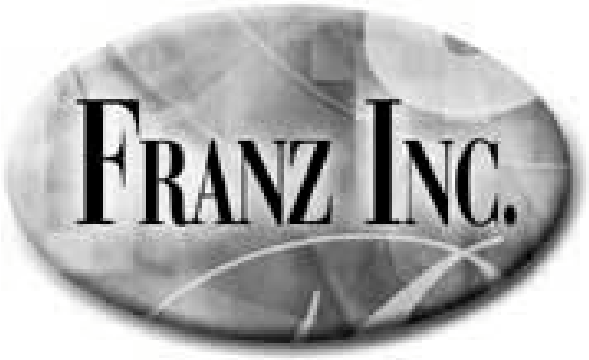
Allegro CL Certification Program

Lisp Programming Series Level 2

Session 1

Homework



Functions

- Write the hello-world function. Pass the stream as an optional argument
- Pass the stream as a keyword argument
- Write the function SUM that returns the sum of all its arguments. Write it such that it can take any number of arguments.

setf

- Write a setf function on 3RD that does this:

```
(setq list '(1 2 3 4 5 6))
(setf (3rd list) 7)
list
⇒ (1 2 7 4 5 6)
```

Functions

- Write a function EXPENSIVE that calculates the square of a number
- Write a function FRUGAL that returns the same answer, but only calls EXPENSIVE when the given argument has not been seen before

Mapping

- Use mapping functions to sum the elements of a list

Multiple Values

- TRUNCATE takes two arguments and returns two values. Write a function that calls it and returns only its second value (the remainder).

Hash Tables

- Using a hash table, write the following:
  - (occurrences '(a b r a ca d a b r a))
  - Returns ((A . 4) (R . 2) (B . 2) (D . 1) (CA . 1))

Macros

- Write some macros that help generate HTML
- Send output to *standard-output*
- (as center “Lisp Class”)
  - Lisp Class
- (with center (princ “Lisp”) (princ “ Class”))
  -
  - Lisp Class
  -

Macro Lab 2

- Implement rotatef as a macro
  - (let ((a 1) (b 2)) (rotatef a b) a) => 2
- Implement "mydefun" as a macro that works like defun
- Implement "mytypecase" as a macro that works like typecase (hint: use typep and cond)
  - (typecase x
  - (symbol (print ‘symbol))
  - (string (print ‘string)))

Closures

```
;;;Where is the closure?

(defun add1 (list)
  (mapcar #'(lambda (n) (+ n 1)) list))

(defun sum (list)
  (let ((sum 0))
     (mapcar #'(lambda (n)
                 (setq sum (+ sum n)))
               list)
      sum))
```