

CS 101 Computer Science I (Spring 2001)
Lab 8: April 4-6, 2001

1. Write a procedure called "**largest-result**" that takes two procedures **p1** and **p2** as arguments and returns a procedure (**largest-result p1 p2**) as a result. The procedures **p1** and **p2** each take a single argument and return a number as a result. When the procedure (**largest-result p1 p2**) is applied to an argument **x**, it will apply both **p1** and **p2** to **x** and return the largest result.

```
(define add2 (lambda (n) (+ n 2)))
(define square (lambda (n) (* n n)))
((largest-result square add2) 1)      ==> 3
((largest-result square add2) 2)      ==> 4
```

2. Write a procedure called "**sumster**" that takes a procedure **fn** as an argument and returns a procedure (**sumster fn**) as a result. The procedure **fn** takes an integer as an argument and returns a number as a result. The procedure (**sumster fn**) takes two integers **low** and **high** as arguments. It returns the sum of the results of applying **fn** to the integers from **low** to **high**, including applying **fn** to **low** and **high** themselves. Your solution should make use of the **apply**, **map**, and **from-to** procedures.

```
((sumster square) 1 10)                ==> 385
((sumster (lambda (x) x)) -5 5) ==> 0
```

3. Write a procedure called "**mirror**" that takes a list **lst** as an argument and returns a procedure (**mirror lst**) as a result. The procedure (**mirror lst**) takes one argument **x** and returns a list ((**mirror lst**) **x**). The length of ((**mirror lst**) **x**) is $2 \bullet N + 1$ where **N** is the length of **lst**. The list **lst** is a prefix of ((**mirror lst**) **x**). The list (**reverse lst**) is a suffix of ((**mirror lst**) **x**). The datum **x** lies in the middle of ((**mirror lst**) **x**). Write the procedure **mirror** so that the **reverse** procedure is called when (**mirror lst**) is evaluated, but **reverse** is not called each time (**mirror lst**) is applied to a new argument.

```
((mirror '(a b c)) 'd)                  ==> (a b c d c b a)
((mirror '(1 2 3)) '(4))                ==> (1 2 3 (4) 3 2 1)
```

4. Write a procedure called "**dist-op**" that takes a procedure **fn** as an argument and returns a procedure (**dist-op fn**). The procedure **fn** takes two arguments. The procedure (**dist-op fn**) takes a value **x** and a list **lst** as arguments. The expression ((**dist-op fn**) **x lst**) evaluates to a list consisting of the results of applying **fn** to all possible combinations of **x** and a member of **lst**.

```
((dist-op +)      1 (list 10 15 20))    ==> (11 16 21)
((dist-op *)      2 (list 10 15 20))    ==> (20 30 40)
((dist-op list) 1 (list 10 15 20))      ==> ((1 10) (1 15) (1 20))
```