# CMPSC 461: Programming Language Concepts, Spring 2024
## Assignment 5 Practice Notes Packet

Prof. Suman Saha

October 14, 2025

## Problem 1: Racket Basics : Knowledge check

What do the following expressions evaluate to:

1. (* 2 (+ 4 5))

2. (= 3 (+ 1 3))

3. (car '(elmer fudd daffy duck))

4. (cdr '(elmer fudd daffy duck))

5. (and (= 1 2) (= 5 (/ 5 1)))

**Problem 2: Racket Basics : Knowledge check**

1. What does the following expression evaluate to:

```
1  (let (( x (+ 2 8))
2            (y 100)) (/ y x) )
```

2. What does the following expression evaluate to:

```
1  (let (( x 100) (y 5) )
2      (let ((x 1))
3            (+ x y) ) )
```

## Problem 3: Racket programs: Recursion

Define a recursive program **remove-at** which removes Kth element from a list and returns the remaining elements of the list in order. Additionally, when K is negative the goal is to remove they Kth element from the right. The function should work as follows:

```
1       (remove-at '() 3) returns '()
2       (remove-at '(a b c d) 2) returns '(a c d)
3       (remove-at '(a b c d) -2) returns '(a b d)
4       (remove-at '(a b c d) 0) returns '(a b c d)
```

## Problem 4: Map practice : Deviation

"Squared deviations from mean" (or SDM) is a metric commonly used in statistical analysis, and is defined as the squared difference of the value from the mean of a given list. Implement a function "sdm" to calculate the squared deviations from mean for each value in a list, and return the answer as a list of SDMs. Use the **map** function to accomplish the same. You can use the *average* function such that "(average lst)" would return the average value of the list of numbers *lst*. The function should work as follows :

```
1 (sdm '(1 3 2 4 5)) returns '(4 0 1 1 4)
2 (sdm '(5 5 5 5 5)) returns '(0 0 0 0 0)
```

## Problem 5: Foldl practice : New Filter

Consider the following scheme programs:

```scheme
(define (foo l) (map (lambda (x) (/ x 2)) l))

(define (goo f l)
    (cond ((null? l) '())
        (else (let ((result (goo f (cdr l))))
            (append (f (car l)) result)))))
```

What will be the output of the following code and describe what does this function return in one sentence:

1. (foo '(2 4 6 8))

2. (goo (lambda (x) (list x x)) '(1 4 9 16))?

## Problem 6: Map practice : Addition

Using *map*, define a function *addN* that takes a list of numbers and a constant "N" as input and returns the list with N added at each element. Using a lambda function can help with the same.

**Problem 7: Building Up 1**

Using any way you like, define a function called *sum_cubes* that takes in a list of numbers and output the summation of each item cubed. If the list is empty, return 0. Assume that the input list would only be numbers and no other input that would make the function dysfunctional. Assume all cases and you may create helper function.

```
1  (sum_cubes '()) ; returns 0
2  (sum_cubes '(2)) ; returns 8
3  (sum_cubes '(2 4 3)) ; returns 99
4  ; Note this is not all possible cases
```

**Problem 8: Building Up 2**

Extending from the previous question for *sum_cubes*. The result and score is not ideal. Plans for a curve is in the making. Define a function or functions *curve* with *foldl* and *length* function to determine the curve for the assignment. The curve is relevant to the class average. If the average is greater than 70, no curve. If the average is equal to 70, no curve. If the average is lower than 70, the curve is equal to 70 minus the class average. You can safely assume all scores are between 0 to 100. Return 0 when scores are empty.

```
1  (curve '()) ; returns 0
2  (curve '(61)) ; returns 9
3  (curve '(70 70 70)) ; returns 0
4  (curve '(46 61 47 73)) ; returns 13.25
5  ; Note this is not all possible cases
```

1. The average function computes the average of a list of numbers.

2. If the list is empty (null? lst), it returns 0.

3. Otherwise, it uses foldl + 0 lst to sum all elements in the list and divides by the length of the list using (length lst) to compute the average.

4. The curve function calculates the appropriate curve based on the class average.

5. If the list is empty (null? lst), the result is 0.

6. If the average is 70, no curve is applied, and the result is 0.

7. If the average is greater than 70, no curve is applied, and the result is 0.

8. If the average is less than 70, the curve is calculated as *70 - average*.

## Problem 9: Building Up 3

After the *curve* function is complete, it is in time to apply the curve. But nobody wants to manually add the curve. Define a function *apply* to the take in a list of scores, and add the curve to each score in the list, then output the list. Use the *map* function and feel free to use your curve function previous defined. If scores are empty, return 0. Assume all scores are from 0 to 100.

```
1  (apply '()) ; returns '()
2  (apply '(61)) ; returns '(70)
3  (apply '(70 70 70)) ; returns '(70 70 70)
4  (apply '(46 61 47 73)) ; returns '(59.25 74.25 60.25 86.25)
5  ; Note this is not all possible cases
```

**Problem 10: Building Up 4**

Define a function `discount`, which takes the name of an organization that someone belongs to and produces the discount (as a factor) that the person should receive on a purchase. Members of BBB get 25%, members of ACM or IEEE get 35%, and members of UPE get 20%. All other organizations get no discount.

| Sample Input | Sample Output |
|:---:|:---:|
| `"IEEE"` | 0.35 |
| `"UPE"` | 0.20 |