Due on Friday 10/28, 10:00 p.m. Don't wait until the last second!

Check that the language level in Dr.Scheme is set to R5RS. Then put all the procedures that you write into one Scheme file named **hw2.scm**. **Be sure to name the procedures as requested in the question**. This way it is easy to run standard test cases.

Write your name and UW net id at the beginning of your hw2.scm file.

1. Let's define the following sequence of natural numbers:

$x_0 = 1$

$x_1 = 2$

$x_{n+2} = x_{n+1} + x_n + x_{n+1} * x_n, \forall n \geq 0$

a.) Write a function (**sequence-rec** n) that calculates x_n while generating a *recursive process.*

b.) Write a function (**sequence-iter** n) that calculates x_n while generating an *iterative process.*

Recall that your process will be iterative if you use tail recursion.

You can assume that $n \geq 0$.

2. a) Define a function (**all** p list) that takes as input a unary predicate and a list. A predicate is a function that takes a single argument and returns either true (#t) or false (#f), depending on whether that argument has a certain property. Your function (called **all**) should return true if all the elements in the given list satisfy predicate p. Otherwise, it should return false.

If the list is empty, then the function **all** should return #t (if a list is empty, we can say that all of its elements satisfy p).

b) Define a function (**exists** p list) that takes as input a unary predicate and a list and returns true if there exists at least one element in the given list that satisfies predicate p. Otherwise, it should return false.

If the list is empty, the function **exists** should return #f (if a list is empty, we cannot say that there exists an element that satisfies p).

3. First, define a variable FIXED-LENGTH and give it a value. Example:

```
(define FIXED-LENGTH 3)
```

Then, define a function (**fixed-length-even-list** L) that receives a list of lists of integer numbers, L. It returns:

#t, if all the lists inside the list L have the length FIXED_LENGTH and there is at least one list inside list L that has only even numbers.

#f, otherwise.

You can assume that the argument list is in the form:

'((1 2) (4 2 4) (5 6 7 4) (2 2 1 3 1)), or '((2 3 4)), etc. Your program will only be tested with lists in the "list of lists of integers" form.

For example, if fed with the above list as argument, the fixed-length-even-list must return #f (for any FIXED-LENGTH). But, (fixed-length-even-list '((1 2 4) (2 4 4) (3 43 3) (3 2 1))) should return #t, as all the lists inside of it have the length as 3 and there's one list that has only even numbers.
Note that you can use (and are advised to use) the functions you defined in parts a) and b) of this problem.

Examples of responses for FIXED-LENGTH = 3:
> (fixed-length-even-list '((1 2 3) (2 2 2)))
#t
> (fixed-length-even-list '((1 2) (2 2 2)))
#f
> (fixed-length-even-list '())
#f
> (fixed-length-even-list '((2 4 2) (1 2 3)))
#t
> (fixed-length-even-list '((2 2 2)))
#t

4. Write a function (**f-to-n** f n) that returns a function that takes one integer argument x and applies f to x n times. You can assume that f is a function that expects one integer argument.
Note that (f-to-n f 0) is the identity function.

5. Define a function (**integral** f n) that would approximate the bounded integral of a function through the trapezoid formula:

$$\int_0^n f(x)dx \approx \sum_{k=0}^{n-1} \frac{f(k)+f(k+1)}{2}$$

Your function (integral f n) should return the approximation of the integral of function f from 0 to n.