Programming languages (TC-2006)

Homework 07

In this homework, you will practice higher-order functions in the Racket language. Please consider that the purpose of this homework is to allow you to practice and identify strengths and weaknesses. Then, implement these functions as requested and avoid using any built-in functions that already do what you are requested to implement. Since this homework focuses on higher-order functions, consider that, in all the functions requested, you must use, at least, one-higher order function such as `map`, `apply`, `filter`, among others. Failing to use at least one higher-order function in each of the requested functions will cancel any points related to such a function.

1 `combine` (20%)

Implement a function in Racket that receives two matrices as arguments and returns a new matrix where each element contains the respective elements from the arguments (as a list).

For example, combining the following matrices[1]:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Will produce the matrix:

$$\begin{pmatrix} (1,a) & (2,b) & (3,c) \\ (4,d) & (5,e) & (6,f) \\ (7,g) & (8,h) & (9,i) \end{pmatrix}$$

For representing the matrices in Racket, use a 'by row' representation.

2 `product` (20%)

Write a function in Racket that returns the Cartesian product of two lists. That is, for lists A and B, the Cartesian product $A \times B$ is the list of all ordered pairs (a, b) where $a \in A$ and $b \in B$.

3 `unique` (20%)

Write a function in Racket that receives a list as input and returns a list that contains only unique elements (all repeated elements are removed).

---

[1]Please note that these matrices are not represented as they will be in Racket.

4 `multiples` **(20%)**

Write a function in Racket that receives two parameters: a list of numbers and an integer. The function returns a list with all the elements in the list which are multiples of the integer provided as second argument.

5 `sum` **(20%)**

Write a function in Racket that sums all the elements in a matrix of numbers. For this assignment, use a 'by row' representation. For example, invoking the function with the argument '((1 2 3) (4 5 6) (7 8 9)) will return 45 (the sum of the numbers 1 to 9).

Deliverables

Prepare an RKT file that contains the functions requested and submit it to Canvas. **Please, do not submit other formats but RKT**. To prepare your RKT file, use the code template distributed along with this document. The template contains some test cases for each function to help you verify that your codes work as requested.

I promise to apply my knowledge, strive for its development, and not use unauthorized or illegal means to complete this activity, following the Tecnológico de Monterrey Student Code of Honor.