**Ejercicios PEP 1**

**Pregunta 1 (30 pts):**

La biblioteca CoolBooks ofrece el servicio de préstamo de libros. Como parte de una mejora en su oferta están evaluando la implementación de un sistema para préstamos online. Se le ha encargado a usted desarrollar un prototipo del nuevo sistema en pseudo-Scheme bajo el paradigma de programación funcional.

En particular, usted debe implementar un prototipo del sistema de préstamo tomando en consideración los siguientes antecedentes. Todo préstamo de material bibliográfico tiene la siguiente información: **un RUT del cliente, ID del recurso facilitado, fecha de préstamo, fecha de retorno, estado de retorno (boolean).**

**Para su simplicidad, asuma que las fechas son simplemente un string.**

1. **(5 pts)** Diseñe una representación de él o los TDAs vinculados al registro de los préstamos de CoolBooks. Luego implemente sus constructores y funciones de pertenencia.
2. **(5 pts)** Implemente una función que permita añadir nuevos préstamos al registro de préstamos de CoolBooks.
3. **(20 pts)** A partir de los TDAs construidos en las preguntas 1 y 2, se solicita implementar las siguientes funciones que permitan lo siguiente:
   1. **(5 pts)** Recuperar todos los préstamos para un rut dado.
   2. **(5 pts)** Recuperar todos los préstamos con estado de retorno no devuelto (false).
   3. **(10 pts)** Recuperar todas las muestras de acuerdo a un criterio personalizado en una función *f* aplicable sobre los datos (individuales o conjunto) de un préstamo, lo cual debe ser indicada en el momento de ejecutar la función.

**NOTA 1:** Recuerde documentar todas sus funciones indicando dominio, recorrido, descripción de su funcionalidad, tipo de recursión (si aplica) y ejemplo de uso. El no hacerlo puede significar un descuento de hasta 25% del puntaje.

**NOTA 2:** Para esta pregunta, puede usar las siguientes funciones de Scheme: car, cdr, if, cond, define, null?, list?, null, let, lazy, lambda, force, not, and, or, equal, cons, list. Cualquier otra función no indicada en esta lista deberá ser implementada.

**SOLUCIÓN**

[**https://github.com/gmartinezramirez/paradigmas-programacion/blob/main/2021-01/01-FP/PEPS/ejercicio\_tipo\_pep1.rkt**](https://github.com/gmartinezramirez/paradigmas-programacion/blob/main/2021-01/01-FP/PEPS/ejercicio_tipo_pep1.rkt)

**Pregunta 2**

SMart es una tienda especializada en la venta de videojuegos está en el proceso de construcción de un sistema utilizando el paradigma funcional que permita visualizar su lista de videojuegos a los clientes. Los videojuegos contienen los siguientes datos específicos: id, nombre, plataforma, precio y calificación (de 1 a 10). Para la construcción de este sistema se ha decidido implementar el módulo encargado de la búsqueda y adición de juegos utilizando Scheme.

1. **Diseñe una representación (sólo la representación)** para un TDA de la lista de videojuegos y especifique un ejemplo de su representación con videojuegos y lista de videojuegos**.** **Luego implemente un constructor, función de pertenencia y selectores naturales.**
2. A partir de la implementación de su TDA, implemente las funciones necesarias que permita agregar un nuevo videojuego a la lista de videojuegos.
3. A partir de la implementación de su TDA, implementar funciones que permitan:  
   1. Recuperar todos los videojuegos según un nombre ingresado.
   2. Recuperar todos los videojuegos que se encuentren en cierto rango de precios (min, max).
   3. Haga la misma función anterior pero currificada.
   4. Recuperar todos los videojuegos de una plataforma en específico
   5. Recuperar todos los videojuegos en base a un criterio personalizado expresado a través de una función *f*. Este criterio se puede aplicar sobre cualquiera de los campos del videojuego, pero es la función *f* la que resuelve esto.
   6. Haga la misma función anterior pero currificada.
4. Implementar una función currificada que permita buscar videojuegos según los criterios indicados en los ítems (b) y (d) de la parte (2) de manera conjuntiva (que se deben cumplir todos los criterios a la vez). Si lo estima conveniente, puede apoyarse en las funciones ya implementadas en la parte (2).

**SOLUCIÓN**

**REPOSITORIO:** [**https://github.com/gmartinezramirez/paradigmas-programacion/blob/main/2021-01/01-FP/PEPS/ejercicio\_tipo\_2.rkt**](https://github.com/gmartinezramirez/paradigmas-programacion/blob/main/2021-01/01-FP/PEPS/ejercicio_tipo_2.rkt)

1. RacketFlix™ es el nuevo sistema de moda para ver películas por *streaming*. El sistema de RacketFlix™ dispone de “películas”, las cuales son compradas por los “usuarios” y son agregadas al “perfil del usuario”. Por lo que cada usuario consta con su propio perfil que despliega las películas adquiridas en este sistema.

Cada película se identifica con un id (*int*), nombre (*string*), duración en minutos (*int*), año (*int*) y una lista de tags (*string*). Los tags corresponden a etiquetas asociadas a cada película. Por ejemplo, “comedia”, “clasico”, “acción”. Asimismo, cada usuario se identifica con su rut (*string*), username (*string*), una lista de preferencias de estilo.

RacketFlix utiliza su propio sistema de criptomonedas, el cual es llamado Racket Dollars™ que corresponde a un número entero. Cada usuario tiene una cartera de Racket Dollars, que indica la cantidad disponible a gastar. Del mismo modo, cada película tiene un precio asociado en este tipo de moneda.

Dentro de la compañía quieren evaluar la posibilidad de migrar RacketFlix a una implementación basada en el paradigma funcional.

1. **(2 pts) Especificar TDAs que permitan cubrir con lo necesario** para cubrir con las demandas de RacketFlix de acuerdo a los requerimientos de las siguientes preguntas. Luego para la implementación del TDA señalar las representaciones escogidas para cada uno.
2. **(5 pts)** Implementar una función que permita registrar un nuevo usuario con nombre de usuario y contraseña tomando como base la representación escogida.
3. **(5 pts)** Implementar una función que permita comprar contenido procurando descontar los montos de su cartera de RacketDollars.
4. **(5 pts)** Implementar una función que permita seleccionar películas de acuerdo a un criterio C determinado.
5. Implementar **SOLO UNA** de las siguientes funciones que pueden ser usadas como criterios de la función del apartado d.
   1. **(5 pts)** Películas cuya duración se encuentre en un rango de minutos [min,max]
   2. **(5 pts)** Películas que contengan TODAS las etiquetas presentadas en una lista L de entrada (etiqueta1, etiqueta2, etiqueta3). Las películas pueden tener otras etiquetas, pero debe contener todas las de la lista para ser seleccionada.
6. **(8 pts)** Implementar una función que permita aplicar una rebaja dada por una función *f* a un conjunto de películas que cumplan con un criterio C. A modo de ejemplo la función *f* podría ser una rebaja del 30% y el criterio de selección aplicaría para todas las películas del catálogo etiquetadas como comedia.

**Pregunta 1 (30 pts)**

Como parte del equipo desarrollador del software de reproducción de medios vía streaming “PlayFunk”, se le ha encomendado desarrollar una solución para el módulo de playlists (listas de reproducción). Los componentes de un medio (elemento en la playlist) son los siguientes: Id del Medio, Artista, Título, Álbum, Género Musical, Disquera, Tipo (“canción” o “videoclip”), Rating (entero entre 1 y 5). El módulo a diseñar e integrar debe ser diseñado en lenguaje Scheme utilizando el Paradigma Funcional.

1. Especificar el o los TDAs necesario(s) para abordar el problema (solo especificación, no implementación) **(2 pts)**.
2. Diseñe la representación para el o los TDAs **(3 pts)**. Luego implemente sus respectivos constructores y selectores **(3 pts)**.
3. De acuerdo a sus TDAs, implemente las siguientes funciones:
   1. Recuperar todos los medios de un cierto género musical **(2 pts)**
   2. Recuperar todas las canciones de una playlist cuyo rating esté entre X e Y **(3 pts)**
   3. Recuperar todos los videoclips de una disquera dada **(2 pts)**
   4. Recuperar todas los medios de un artista, incluso si se ingresa un string incompleto (ej: “beat” debería coincidir con “the beatles”) **(3 pts)**
   5. Recuperar todos los medios en base a un criterio personalizado en una función *f* ya existente, la cual se encarga de aplicar el criterio. **(4 pts)**
4. Diseñe una función que recupere medios a partir de los siguientes parámetros: Un playlist, una consulta sobre el Artista o el Título (string completo o incompleto), un género musical y un tipo (4 parámetros de entrada). La respuesta de esta función no debe incluir medios repetidos **(8 pts)**.

**Nota 1:** *Puede hacer uso de la función string-search-all que permite buscar un string dentro de otro string. Esta función entrega una lista con las posiciones dónde se encuentra el primer string en el segundo. Ejemplo:* (string-search-all "rat" "pirate rating") => ‘(2 7)

**Nota 2:** *Para cada una de sus funciones procure documentar el dominio, recorrido, tipo de recursión empleada (si aplica), casos base y descomposición recursiva. No realizar esto implica un descuento de 0,25 (en la escala de evaluación de 0, .25, .5, .75, 1 que se aplicará en el proceso de evaluación) por cada función implementada.*

**Nota 3:** *Además de las funciones provistas en la Nota 1, solo puede usar funciones como car, cdr, null?, if, cons, null, define, let, cond, equal?, eqv? eq? list, list?, cons?. Otras funciones las debe implementar o bien preguntar al profesor.*

**Pregunta 1 (30 pts)**

Una Matriz Dispersa es aquella matriz en la que solo se almacenan elementos que existen. Para el caso de una matriz dispersa de números, se asume que sus elementos están conformados por los números distintos de 0. Por tanto, los elementos con valor igual a 0 no se almacenan. En base a esta definición y al paradigma funcional utilizando una notación basada en cálculo lambda o pseudo-scheme,

1. Especificar el o los TDAs para poder trabajar con Matrices dispersas (considere solo hasta la capa de modificadores). **(5 pts).**
2. Señalar una representación para poder implementar el o los TDAs especificados en la parte a) **(5 pts).**
3. Implementar los TDAs en base a la especificación de la parte a) **(5 pts)**.
4. Implementar una función para ponderar los elementos de una matriz dispersa a\*M **(5 pts).**
5. Implementar una función para sumar dos matrices dispersas (**10 pts).**

**Nota:** Procure documentar adecuadamente cada función implementada (dominio, recorrido, descripción, tipo de recursión).

**Pregunta 2 (30 pts)**

TicTacToe (popularmente conocido como Gato) es un juego de mesa donde dos jugadores se enfrentan a través de un tablero de 2x2 hasta NxN. El objetivo es lograr disponer el tipo de pieza asignada a cada jugador (**X** o **O**) de manera contigua siguiendo un patrón horizontal, vertical o diagonal. Para el caso de tableros de dimensiones bajas (ej: 2x2 y 3x3) generalmente la longitud del patrón es N. Luego, en este caso un jugador cuya pieza es la X resulta ganador si logra poner de forma contigua 3 Xs de forma horizontal, vertical o diagonal.

1. Especificar el o los TDAs necesario(s) para abordar el problema (solo especificación, no implementación) **(2 pts)**.
2. Especificar una representación para el tablero **(2 pts)**.
3. Implementar una función que construya un tablero vacío de tamaño NxM **(2 pts)**.
4. Implementar selectores naturales para su TDA **(2 pts)**.
5. Implementar una función que permita realizar una jugada en el tablero (disponer una X o O en una posición dada) procurando controlar los turnos. En otras palabras, un jugador (X o O) no puede realizar más de una jugada consecutiva **(8 pts)**.
6. Implementar una función que permita retornar el ganador de un juego (X o O) en base a jugadas horizontales, verticales y diagonales de tamaño L (cantidad de registros consecutivos de X o O en una horizontal, vertical o diagonal). Si no hay ganador, entrega un elemento predefinido dentro del recorrido válido de la función **(14 pts)**.

**Nota 1:** *Para cada una de sus funciones procure documentar el dominio, recorrido, tipo de recursión empleada (si aplica), casos base y descomposición recursiva. No realizar esto implica un descuento de 0,25 (en la escala de evaluación de 0, .25, .5, .75, 1 que se aplicará en el proceso de evaluación) por cada función implementada.*

**Nota 2:** *Además de las funciones provistas en la Nota 1, solo puede usar funciones como car, cdr, null?, if, cons, null, define, let, cond, equal?, eqv? eq? list, list?, cons?. Otras funciones las debe implementar o bien preguntar al profesor.*

**Pregunta 2 (30 pts)**

Considere el juego arcade Outlaw en el que dos pistoleros se enfrenta a un duelo a muerte (Figura 1). En el juego que puede ser de 1 o 2 jugadores, cada jugador o CPU controla a un jugador con desplazamientos que van de izquierda a derecha o de arriba a abajo. En esta versión del juego (para la prueba) los disparos son siempre horizontales y su trayectoria puede ser interrumpida por uno o más obstáculos (cactus, carreta, etc.) que pueden servir de escudo a los personajes. En el juego, cada impacto en el jugador contrario entrega un punto al jugador que ejecutó el disparo. Gana el jugador que primero llega a una cantidad **A** de aciertos, pudiendo darse también una situación de empate. Estos estados correspondientes a Derrota, Victoria y Empate, además de JUGANDO corresponden a los estados del juego.

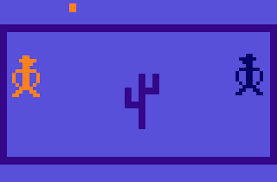


Figura 1: Captura de Outlaw.

En base a esta descripción y al paradigma funcional utilizando una notación basada en cálculo lambda o pseudo-scheme,

1. Especificar el o los TDAs para poder desarrollar el juego Outlaw. (considere solo hasta la capa de modificadores). **(5 pts).**
2. Señalar una representación para poder implementar el o los TDAs especificados en la parte a) **(5 pts).**
3. Implementar los TDAs en base a la especificación de la parte a) **(5 pts)**.
4. Implementar una función para la modalidad de 1 player que permita al jugador desplazarse y disparar y que por otro lado permita al enemigo controlado por CPU realizar su desplazamiento y correspondiente disparo. El retorno de la función debe permitir al jugador seguir realizando jugadas mientras el estado del juego sea JUGANDO. **(5 pts).**
5. Implementar la misma función de la parte d) donde el resultado se expresa como una lista infinita con los resultados parciales de una jugada en cada instante o cuadro de tiempo. La lista se genera solo hasta un tiempo t que entrega el usuario como parámetro de entrada y que sirve para calcular la posición del disparo en cada instante de tiempo (i.e., t0, t1,t2, t3, ……. t) (**10 pts).**

**Nota:** Procure documentar adecuadamente cada función implementada (dominio, recorrido, descripción, tipo de recursión).

**Pregunta 2 (30 pts):** Considere la implementación parcial del TDA imágen en cuya representación cada pixel es expresado a través de una lista de 5 elementos de tipo entero positivo + {0} (enteros positivos incluido el cero) correspondientes a la posición en los ejes X,Y y los colores RGB (X, Y, Red, Green, Blue). Luego una imagen queda expresada por una lista de N píxeles (puede asumir que la imagen es cuadrada, largo = ancho o sqrt(N), sin embargo este dato no afecta en absoluto las preguntas ni sus respuestas).

Tomando como base esta representación, implementar las funciones señaladas a continuación:

1. (**7 pts**) Función que comprime una imagen eliminando todos los pixeles blancos (i.e., cuando R=255 y G=255 y B=255) de la imagen. El resultado es una imagen comprimida cuya representación es distinta a la de una imagen sin comprimir ya que mantiene un registro de la dimensión original de la imagen en la primera posición de la lista.

El prototipo para la función debe ser: (comprimir imagen)

1. (**7 pts**) Función que convierte una imagen a color a una blanco y negro. Para esto, en cada pixel, reemplaza los valores R,G,B por el resultado de la siguiente operación:

redondear ((R + G + B) / 3.0) / 255) \* 255

El prototipo para la función debe ser: (colorToBW imagen)

1. (**16 pts**) Función que permite ajustar los colores de una imagen en base a una función f que permite manipular de forma independiente cada color de la imagen (R, G, B).
2. **(5 pts)** El prototipo para la función debe ser: (ajustarColor f imagen)
3. **(8 pts)** Implementar funciones currificadas para ajustar cada canal (i.e., R, G y B) (Aclaración: El ajuste se da en base a un ponderador como se aprecia en la siguiente pregunta). Estas funciones se entregan a la función ajustarColor a través del parámetro *f*.
4. **(3 pts)** Muestre cómo haría uso de la función ajustarColor para cambiar el color azul (Blue) de una imagen I1 (preexistente) al 30% (ponderador = 0.3).

**NOTA 1:**  Procure hacer un adecuado uso del paradigma funcional (principalmente declarativo en la medida de lo posible) de manera de no redundar en esfuerzos de implementación. Si lo estima conveniente, puede implementar funciones propias del TDA y otras funciones complementarias (incluyendo las de la pregunta 1). La implementación innecesaria y/o redundante de funciones que desaprovechen el paradigma funcional tendrán una penalización de hasta **2 puntos** por cada ítem afectado.

**NOTA 2:** Recuerde documentar todas sus funciones indicando dominio, recorrido, descripción de su funcionalidad, tipo de recursión (si aplica) y ejemplo de uso (una sola línea, es un ejemplo de cómo se evalúa ej: (función 3 4 5)). El no hacerlo puede significar un descuento de hasta **1 punto** por cada ítem afectado.

**NOTA 3:** Para esta pregunta, puede usar las siguientes funciones de Scheme: car, cdr, if, cond, define, null?, =, eqv?, eq? equal?, +, \*, /, -, list?, null, let, lazy, lambda, force, not, and, or, eq?, cons, list, round (para redondear). Cualquier otra función no indicada en esta lista deberá ser implementada.

**Pregunta 1 (30 pts):** El laboratorio médico VeriHealth S.A. se encarga de recolectar y evaluar exámenes de laboratorio de pacientes de varios hospitales y clínicas de la ciudad, con el fin de determinar las posibles patologías que puedan tener (ej: diabetes, hipotiroidismo, anemia, etc.). Con el advenimiento de las enfermedades de invierno, se hace necesario implementar en VeriHealth S.A. un nuevo sistema para el etiquetado de muestras. Se le ha encargado a usted desarrollar un prototipo del sistema en pseudo-Scheme bajo el paradigma de programación funcional.

Toda muestra posee un RUT del paciente (string), ID de muestra (entero), su edad (entero), su sexo biológico (string, “M” o “F”), centro hospitalario de origen (string), si la muestra ya ha sido evaluada (booleano), si el paciente consume medicamentos (booleano), sus enfermedades preexistentes (string) y las enfermedades detectadas por la misma muestra (string).

1. Diseñe una representación para los TDAs Lista de Muestras y Muestra, añadiendo un ejemplo de uso. Luego implemente sus constructores y funciones de pertenencia. **(6 ptos)**
2. Implemente una función para el TDA Lista de Muestras que dado un RUT de un paciente y su ID de muestra, permita cambiar el estado de evaluación de la muestra y además cambiar las enfermedades que fueron detectadas en dicha muestra. **(4 ptos)**
3. A partir de sus implementaciones en las preguntas 1 y 2, se solicita implementar funciones que permitan lo siguiente:
   1. Recuperar todas las muestras de un paciente determinado (por RUT). **(3 ptos)**
   2. Recuperar todas las muestras de pacientes dentro de un rango etáreo (especificando edad mínima y máxima). **(3 ptos)**
   3. Recuperar todas las muestras cuya enfermedad detectada es la indicada como parámetro de entrada. Esta función también debe aceptar coincidencias parciales (Ej: “abe” debiese retornar “Diabetes”). **(4 ptos)**
   4. Recuperar todas las muestras de acuerdo a un criterio personalizado en una función *f*, la cual debe ser indicada en el momento de ejecutar la función. **(5 ptos)**
   5. Hacer una función currificada que permita obtener las muestras bajo los criterios indicados en los ítems (b) y (c) de manera simultánea. **(5 ptos)**

**NOTA 1:**  Procure hacer un adecuado uso del paradigma funcional (principalmente declarativo en la medida de lo posible) de manera de no redundar en esfuerzos de implementación. Si lo estima conveniente, puede implementar funciones propias del TDA y otras funciones complementarias (incluyendo las de la pregunta 2). La implementación innecesaria y/o redundante de funciones que desaprovechen el paradigma funcional tendrán una penalización de hasta **2 puntos** por cada ítem afectado.

**NOTA 2:** Recuerde documentar todas sus funciones indicando dominio, recorrido, descripción de su funcionalidad, tipo de recursión (si aplica) y ejemplo de uso. El no hacerlo puede significar un descuento de hasta **1 punto** por cada ítem afectado.

**NOTA 3:** Para esta pregunta, puede usar las siguientes funciones de Scheme: car, cdr, if, cond, define, null?, =, eqv?, eq? equal?, +, \*, /, -, list?, null, let, lazy, lambda, force, not, and, or, eq?, cons, list, **<, >, integer?, string?, boolean?**. Cualquier otra función no indicada en esta lista deberá ser implementada.

**NOTA 4:** Además de las funciones indicadas en la nota anterior, puede hacer uso de la función (previamente implementada - asuma su existencia, no la implemente) llamada *partial-match* que permite buscar un string dentro de otro, retornando como resultado una lista indicando si hay coincidencias (booleano) además del número de coincidencias detectadas.

Ejemplo: (partial-match “ma” “Cáncer de mama”) => ‘(#t 2)

(partial-match “ami” “Anemia”) => ‘(#f 0)