POR Paradigmas de Programación (2021-1)

Agosto

NOMBRE: _____ RUT: _____ Profesor: _____

Instrucciones

- **El siguiente enunciado tiene 3 partes correspondientes a las POR de la PEP1, PEP2 y PEP3. Responder sólo la parte correspondiente a la PEP en la que obtuvo la calificación más baja.**
- Tiempo estimado de respuesta: 90 minutos
- Tiempo disponible para entregar respuestas desde que tuvo acceso al enunciado: 3 horas hasta las 23:59 del 25 de Agosto de 2021.
- Lugar de entrega: Campus Virtual.
- Escriba sus respuestas en hojas separadas
- Puede escribir sus respuestas directamente en un documento electrónico o bien puede hacerlo en papel. Escoja el que más le acomode.
- Recomendación: Para asegurar acceso continuo a este enunciado en caso de cortes de suministro eléctrico o en el acceso a Internet, procure descargar este enunciado y si puede, imprimalo.
- Identifique cada hoja de respuestas con su nombre y rut, lo que sirve de complemento a la declaración que ha aceptado en Campus Virtual.
- Identifique cada hoja de respuesta con el profesor de su sección.
- Al terminar la evaluación,
    - Si desarrolló sus respuestas con lápiz y papel, saqué una fotografía o escaneé cada una de sus hojas de respuestas. Asegúrese que las imágenes de sus capturas son legibles (abra el archivo y verifique). Luego, agrupe todas las imágenes en un PDF (en Windows basta con seleccionar las imágenes desde el explorador de archivos -> clic derecho -> Imprimir -> Escoger Impresora PDF de Microsoft u otra que disponga) o bien en un archivo comprimido *.zip* o *.rar*.
    - Si desarrolló sus respuestas directamente en formato digital (ej: documento Word, Google Docs, Latex, etc.) guardar el documento como PDF. Se recomienda este formato para evitar problemas en la visualización de estilos.
- Finalmente, subir el archivo al espacio creado para tales efectos en Campus Virtual. Una vez subido el archivo, descárguelo y asegúrese que el archivo se puede abrir (para evitar situaciones de archivos dañados).
- Excepción: En la eventualidad que no pueda subir su trabajo a Campus Virtual en el plazo disponible, enviarlo inmediatamente vía correo a su profesor.
- Se reitera la importancia de su integridad académica (código de honor al que usted se ha suscrito) al momento de realizar estas evaluaciones no presenciales. Copias o solicitar ayuda a terceros serán sancionadas en base al reglamento institucional. Sus respuestas deben basarse únicamente en base a su estudio de la materia.

Sobre la evaluación. Todo ítem se evalúa en una escala de 3 puntos: 0 (sin respuesta o respuesta no aborda el problema), 0.5, 1 (respuesta completa sin errores), según grado de cumplimiento.

- **NOTA 1:** Procure hacer un adecuado uso del paradigma funcional. La aplicación inadecuada del paradigma tendrá una penalización de hasta 2 puntos por cada ítem afectado.

PARTE I - POR PEP 1 (Paradigma Funcional)

1) **(10 pts)** Modifique el siguiente código para que se utilice recursión de cola y la función entregue el mismo resultado. Mantenga la misma cabecera de la función para encapsular el tipo de recursión empleada.

```
(define (algo x)
       (if (null? x)
              1
              (+ 1 (algo (cdr x)))))
```

2) **(20 pts)** Implementar una función en pseudo-scheme que busque un elemento en una lista a partir de un criterio definido mediante una función booleana ***f*** currificada y entregue la posición en la lista (comenzando con la posición 0 para el primer elemento) donde se encuentra la primera ocurrencia del elemento buscado de izquierda a derecha. La función requerida se debe invocar de la siguiente forma:

```
(busqueda '(5 2 88 "hola" -4)  (f  "hola"))
```

donde f en este caso es

```
(define (f p1 p2)
       (equal? p1 p2)
)
```

Luego, para el ejemplo anterior el retorno esperado es 3. En caso de no encontrarse el elemento a partir del criterio definido por la función ***f*** se debe retornar -1. De ser necesario puede crear más funciones además de la función 'búsqueda'.

3) **(30 pts)** Considere el TDA Vector que permite las siguientes operaciones: suma de vectores y ponderación de vectores por un escalar. Para efectos de esta pregunta, los vectores quedan representados por listas de números y las operaciones a abordar como parte de la implementación del TDA se especifican de la siguiente manera:

a) **(5 pts)** Ponderación de vectores: Se multiplica el ponderador por valores en cada eje:

ej: 4*(a1,b1) = (4*a1, 4*b1)

b) **(10 pts)** Suma de dos vectores: Se suman eje a eje cada vector. ej (a1,b1)+(a2,v2)=((a1+a2),(b1+b2))

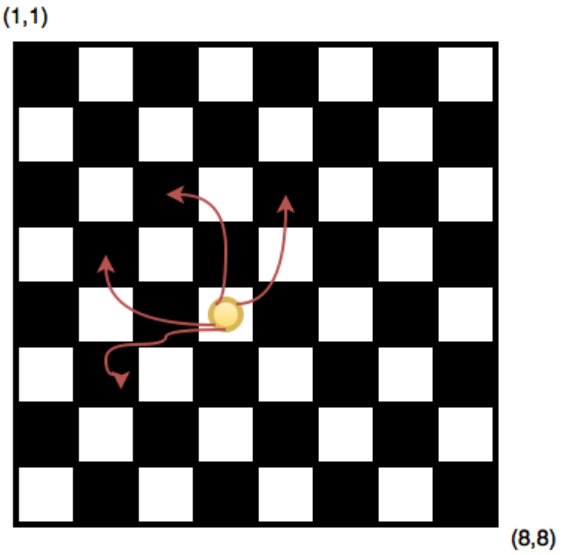
c) **(15 pts)** Suma de n vectores: Función que puede sumar n vectores (función n-variable). Ejemplo (sumaVectores v1v2v v3) (sumaVectores v1v2v v3 v4 v5). Para esto puede usar encabezados de argumentos variables (define (f . params)) donde params es una lista de argumentos.

Nota: Solo puede usar los recursos básicos de scheme: define, lambda, +, -, *, if, let, cond, <, >, =, equal?, eqv?, eq?, cons, pair?, null, null?, list. Si decide usar funciones como map, filter, sort, append, etc. debe implementar estas funciones.

PARTE II - POR PEP 2 (Paradigma Lógico)

Pregunta 1 (30 pts)

Supongamos que los cuadros del tablero de ajedrez los representamos por pares de números [X,Y] con X e Y entre 1 y 8.



1) Definir la relación salta(+C1,?C2) que se verifica si el caballo puede pasar en un movimiento del cuadrado C1 al cuadrado C2 **(9 ptos)**. Por ejemplo,

```
?- salta([1,1],S). S=[3,2]; S=[2,3];
No
```

2) Definir la relación camino(L) que verifique si L (lista de tamaño N) es una lista de cuadrados representando el camino recorrido por un caballo sobre un tablero vacío **(9 ptos)**. Por ejemplo,

```
?-camino([[1,1],C]). C=[3,2];
C=[2,3];    No
```

3) Usando la relación `camino`, escribir una consulta para determinar los caminos de longitud 4 por los que puede desplazarse un caballo desde cuadro [2,1] hasta el otro extremo del tablero (Y=8) de forma que en el segundo movimiento pase por el cuadro [5,4] **(6 ptos)**.

4) Calcular el menor número de movimientos necesarios para desplazar el caballo del cuadro [1,1] al [2,2]. ¿Cuántos caminos de dicha longitud hay de [1,1] a [2,2]? **(6 ptos)**

Nota: Si no conoce las reglas de ajedrez, considere que un caballo se puede desplazar por tres espacios en forma de L. Ejemplo, estando el caballo en [1,2] se puede desplazar a [3,1], [3,3] o [2,4]. La figura ilustra precisamente algunas posibilidades de movimientos para un caballo ubicado en la posición [5,4].

Pregunta 2 (30 pts).

Considere una representación de conjuntos de elementos a través de listas. En este contexto, interesa contar con predicados para responder a las siguientes preguntas:

1) miembro(E,C), que es verdadero si y sólo si X está contenido en L. **(1 pts)**
2) cardinalidad(C,T), que es verdadero si y sólo si T corresponde a la cantidad de elementos contenidos en C. **(2 pts)**
3) subconjunto(C1,C2), que es verdadero si y sólo si C1 es subconjunto de C2. **(3 pts)**
4) disjunto(C1,C2), que es verdadero si y sólo si C1 es disjunto con C2 (no hay elementos en común). **(3 pts)**
5) diferencia(C1,C2,C3), que es verdadero si y sólo si C3 es la diferencia entre C1 y C2. **(3 pts)**
6) union(C1,C2,C3), que es verdadero si y sólo si C3 es la unión de C1 y C2. **(4 pts)**
7) interseccion(C1,C2,C3), que es verdadero si y sólo si C3 es la intersección entre C1 y C2. **(4 pts)**
8) unionSinRepetición(C1,C2,C3): que es verdadero si y sólo si C3 es la unión de C1 y C2 sin elementos repetidos. **(5 pts)**
9) unionOrdenada(C1,C2,C3), que es verdadero si y sólo si C3 es la unión de C1 y C2 con elementos ordenados de menor a mayor sin repetición. Solo este predicado asume que los elementos de la lista son números. **(5 pts)**

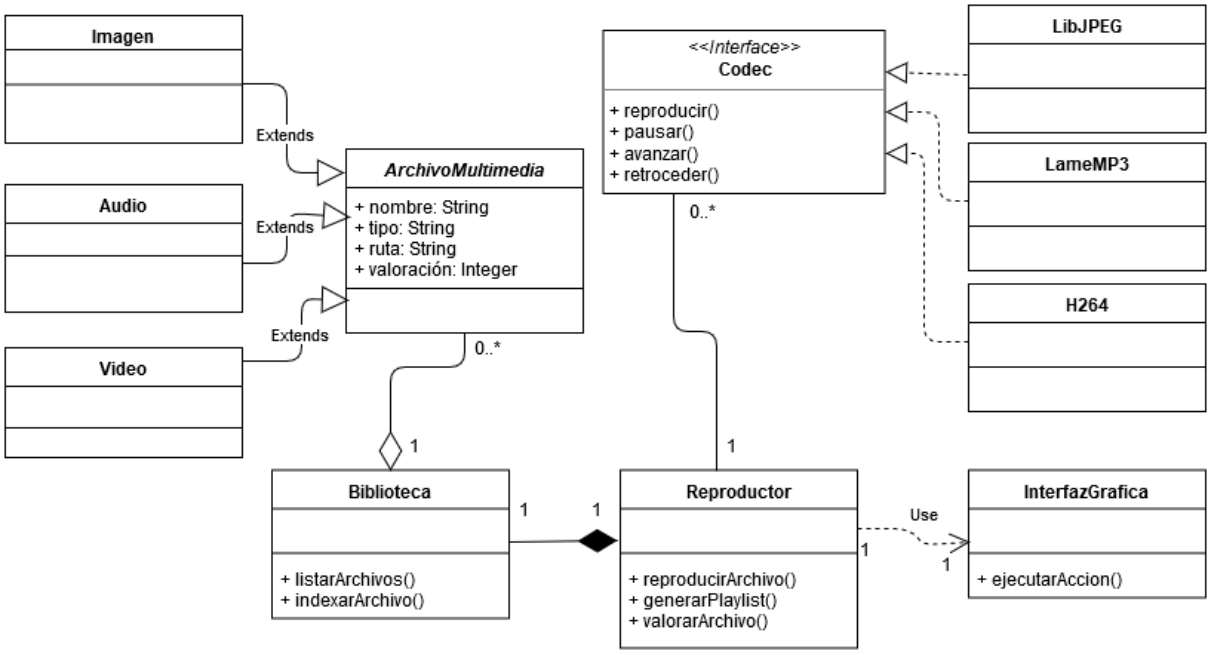
Luego, bajo el paradigma de programación lógico, implementar estos predicados en pseudo-Prolog. De manera sucinta, para cada predicado como meta principal procure especificar los dominios, metas secundarias, cláusulas (hechos y reglas). La puntuación de esta documentación está contemplada dentro del puntaje de cada ítem.

NOTA: Si bien varios de estos predicados ya están implementados en Prolog, usted debe implementarlos desde cero empleando los recursos elementales de la programación lógica

PARTE III - POR PEP 3 (Paradigma Orientado a Objetos)

Pregunta 1 (30 pts).

Considere el siguiente diagrama de clases UML:



A partir de este diagrama, realice lo siguiente:

1) Describa (en no más de 10 líneas) una descripción coherente y concisa del problema que representa este diagrama (una narrativa). Procure en su relato especificar las relaciones entre las clases del diagrama y mencionar todos los comportamientos que pueden realizar éstas **(10 ptos)**
2) Escriba el esqueleto del programa (en pseudo-Java o pseudo-C#) mencionando las clases, sus atributos y los tipos de entrada y salida de los métodos, sin especificar la lógica de estos **(20 ptos)**

Pregunta 2 (30 pts)

La compañía de transportes TransPort cuenta con una variada flota de vehículos para servir a la comunidad en los traslados dentro y fuera de la ciudad. Dentro de los vehículos que posee TransPort, se tiene automóviles, bicicletas, motocicletas, camiones y drones, entre otros. Cada medio transporte tiene características únicas respecto del tipo de servicio que pueden prestar dependiendo generalmente de las capacidades de espacio, distancia y autonomía en los recorridos. TransPort ofrece a sus clientes la posibilidad de llevarlos a ellos mismos o enviar alguna encomienda (paquete, carta, etc.) desde un punto de origen a un punto de destino a través de alguno de sus medios de transporte. Dicho traslado tiene un costo que se calcula en base a la distancia del tramo (en Km.) y la tarifa por Km recorrido que aplique según el horario en que se ha efectuado el viaje. Luego el cálculo para determinar el monto a pagar es simplemente *DistanciaRecorrida*Tarifa*. Al respecto, existen tres tipos de tarifa, Normal, Rebajada y Alta cada una de las cuales puede estar asociada a uno o más horarios. Cada servicio prestado a un cliente se registra en el historial de cada cliente. Además, cada transacción por servicio tiene vinculado algún tipo de recibo por concepto de pago, siendo este recibo una boleta o una factura. Respecto de la diferencia principal entre ambos tipos de documentos se tiene que la Factura ofrece un desglose del cargo por servicio contratado y el IVA, además de información respecto del Giro y RUT de la empresa a quién va dirigida la factura. Para el caso de traslado de encomiendas es preciso dejar registro de la descripción general del paquete, valor aproximado del contenido, dimensiones aproximada y peso del paquete. El origen y destino en los traslado se especifica a través de un sistema de coordenadas georeferenciadas, lo que a través del uso de servicios de mapas externos (ej: Google Maps) ofrecidos a través de una API local llamada *Maps*, permite fácilmente calcular la distancia entre dos puntos a través del servicio *getDistance* que opera sobre el origen y el punto de destino.

a) Expresar la narrativa anterior a través de un diagrama de clases (UML). Procure incluir todas las clases, interfaces, atributos, métodos y relaciones necesarias ajustándose únicamente a lo expresado en el texto anterior. (**8 pts.**)
b) Indicar con comentarios en el mismo diagrama los tipos de relaciones expresadas. (**3 pts.**)
c) A partir de su diagrama de clases, implementar en pseudo-C# de forma completamente consistente su modelo. Centrarse solo en los esqueletos de clases, atributos, métodos y las relaciones presentes. No implementar algoritmos (**7 pts.**)
d) ¿Dónde ubicaría en su modelo y código los comportamientos relativos a la determinación del monto a pagar para un traslado? (**3 pts**)
e) Implementar en pseudo-C# y donde corresponda el o los algoritmos necesarios para determinar el costo por traslado (**6 pts.**)
f) Implementar en pseudo-C# un método *main* donde quede en evidencia cómo se comunicarán los objetos para que un cliente solicite a través de *TransPort* el traslado como pasajero o bien el envío de encomiendas y donde el cliente pueda consultar sobre el total a pagar. Para este caso considere los siguientes antecedentes que deben quedar evidenciados en la instanciación y comunicación de objetos realizado en el *main* (**3 pts.**). (i) Las tarifa Normal se sitúa en el horario de las 10:00 a 18:00. La tarifa Rebajada se da en el horario de las 21:00 a 6:00. Finalmente, la tarifa alta se da entre las 6:00 y 10:00 y entre las 18:00 y 21:00. (ii) Respecto