PEP 1 Paradigmas de Programación (2021-2)

NOMBRE:_____ RUT: _____ Profesor: _____

Instrucciones

- Tiempo estimado de respuesta: 90 minutos
- El enunciado estará disponible desde las 13:00 hrs del 7 de Diciembre
- Tiempo disponible para entregar respuestas desde que tuvo acceso al enunciado: 3 horas hasta las 13:00 PM del 8 de Diciembre de 2021 (es opción de cada estudiante si quiere entregar la prueba durante el periodo habilitado del 8 de Diciembre, considerando que se trata de un día feriado).
- 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 escanee 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.

Sobre la evaluación. Todo ítem se evalúa en una escala de 5 puntos: 0 (sin respuesta o respuesta no aborda el problema), 0.25 (respuesta con errores mayores), 0.5 (respuesta con errores), 0.75 (respuesta con errores menores), 1 (respuesta completa sin errores), según grado de cumplimiento.

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. 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 cada pregunta, puede usar las siguientes funciones de Scheme: car, cdr, if, cond, define, let, null?, =, >, <, eqv?, eq? equal?, +, *, /, -, list?, null, lazy, lambda, force, not, and, or, cons,

list, round (para redondear), length. Cualquier otra función no indicada en esta lista deberá ser implementada.

Pregunta 1 (30 pts)

Se planea construir un sistema de administración/cliente de correos electrónicos (ej: Thunderbird) a través del cual un usuario (individual registrado con su propio correo) puede redactar, enviar, responder, reenviar, buscar y recibir correos. Los correos se clasifican en una bandeja de entrada y de salida. Además, los correos tienen un remitente, un destinatario, un asunto y un cuerpo. Por otro lado, el usuario maneja una lista de contactos con los nombres, organización y dirección de correo: A partir de estos antecedentes:

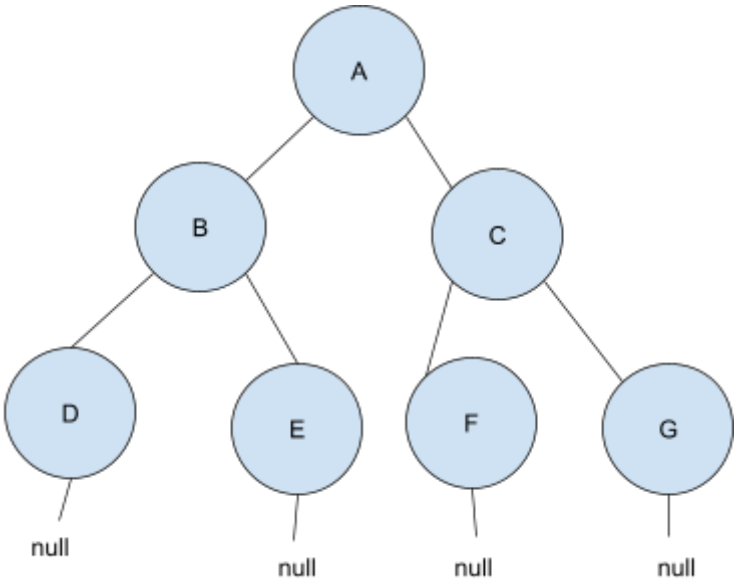
a) (**2 pts**) Especificar el o los TDA(s) necesarios para abordar la construcción del sistema. Sólo considere lo absolutamente necesario para cubrir los requerimientos de esta pregunta.
b) (**3 pts**) Implementar el o los TDA(s) a partir de la especificación propuesta en el ítem (a)
c) **(5 pts)** Implementar una función que permita enviar un correo.
d) **(5 pts)** Implementar una función que permita responder un correo, procurando añadir al comienzo del asunto del mismo el texto “Re: “.
e) **(5 pts)** Implementar una función que permita reenviar un correo, procurando añadir al comienzo del asunto del mismo el texto “Fwd: “.
f) **(8 pts)** Implementar una función recursiva (de cola o natural - debe señalar en la documentación de la función cuál usa) que permita buscar la dirección de correo eléctronico de un contacto a partir del nombre de la persona. Si existen varias coincidencias, la función retorna el correo electrónico del primer resultado. Para esta función puede usar *(string-contains? s contained)*, que permite buscar el string contained en el string s.
g) (**2 pts**) Muestre con un ejemplo completo el uso de las funciones c, d, e y f.

Pregunta 2 (30 pts)

Considere la implementación del TDA árbol binario a partir de la siguiente representación:

árbol = null | árbol X string X árbol

a partir de la cual es posible construir árboles como el siguiente:



Para el TDA árbol binario y la representación indicada para su implementación

a) (**3 pts**) Implementar los selectores naturales: hijoIzquierdo, raiz, hijoDerecho.
b) (**9 pts**) Implementar una función que permita recorrer un árbol en orden prefijo. El resultado de la función es un string con elementos concatenados. Para el árbol de la figura el resultado sería “ABDECFG”.
c) (**9 pts**) Implementar una función que permita recorrer un árbol en orden infijo. El resultado de la función es un string con elementos concatenados. Para el árbol de la figura el resultado sería “DBEAFCG”.
d) (**9 pts**) Implementar una función que permita recorrer un árbol en orden postfijo. El resultado de la función es un string con elementos concatenados. Para el árbol de la figura el resultado sería “DEBFGCA”.

Tip: Para las funciones b, c y d puede usar la función (string-append str …) que permite concatenar dos o más strings.