Trabajo Práctico 3. Programación Funcional

Puntuación

Puntaje Total: 100 puntos

Aprobación: 60 puntos

Fecha de entrega: 14/12/2020 23:55 Hs.

Condiciones de entrega

1. El presente trabajo práctico deberá resolverse en grupo de hasta 3 (tres) personas.
2. Entrega: Se realizará por medio del campus virtual de la UTN, en la tarea correspondiente al TP 3. La extensión del archivo será .zip o .rar, de acuerdo al programa de compresión usado. El nombre del archivo se consigue concatenando un prefijo del número del TP con los apellidos de los integrantes separados por guiones (Ej: Pérez y Abdala, el nombre será `tp3-abdala-perez.zip`). Note que no hay espacios en blanco ni acentos en el nombre de archivo. Dentro del archivo de entrega, deben constar los siguientes:
    - Archivo con el código fuente Scheme denominado `TP3.rkt` debidamente comentado.
    - Los casos de prueba se entregarán en un archivo de texto, no deben ser capturas de pantalla. Deberán cubrir diferentes resultados que puedan obtenerse de la evaluación de las funciones solicitadas. Se enfatiza que se adjunten casos de prueba que sean claros, válidos y suficientes para poder verificar el trabajo. Entregar este archivo con el nombre `casos-de-prueba.txt`.
    - Archivo de texto (`integrantes.txt`) con una línea para cada integrante en la cual figure el nombre del alumno/a y su dirección de email.
3. Penalización por entrega fuera de término: Si el trabajo práctico se entrega después de la fecha indicada, y hasta una semana tarde, tendrá una quita de 15 puntos. No serán recibidos trabajos luego de una semana de la fecha de entrega. Los trabajos que se deban rehacer/corregir fuera de la fecha de entrega tienen una quita de 30 puntos.

Enunciado

Lights Out[1] (LO) es un juego electrónico producido por Tiger Electronics en 1995. El juego consta de una grilla con luces, de 5 por 5 elementos. Cuando el juego inicia, algunas de las luces están encendidas; puede ser un patrón previamente almacenado en el juego, o uno aleatorio. Al presionar una de las luces, las vecinas (arriba, abajo, derecha e izquierda) y ella misma cambian su estado: la encendida se apaga y la apagada se enciende. La meta del juego es apagar todas las luces para ganar. En otra modalidad del juego, no se busca sólo apagar todas las luces, sino la menor cantidad de pulsaciones (gana el que lo hace en menos pulsaciones). En este TP, en cambio, basta que la secuencia de pulsaciones logre el apagado de todas las luces para ser considerada una solución.

Un tablero es una lista que contiene un *dotted pair* para cada luz encendida, donde el primer elemento es el renglón y el segundo es el número de columna. Esta lista tiene los pares en cualquier orden, no necesariamente por renglón o por columna. Por ejemplo, el tablero dibujado más abajo se puede representar con la lista:

```
'((1 . 3) (5 . 2) (1 . 1) (1 . 4) (3 . 5))
```

Funciones a entregar

1) Escriba la función Scheme **`tablero-inicial-1`** que devuelve el tablero fijo siguiente ("💡" indica una luz encendida):[2]

💡	□	💡	💡	□
□	□	□	□	□
□	□	□	□	💡
□	□	□	□	□
□	💡	□	□	□

El ejemplo del gráfico anterior es:

```
(tablero-inicial-1)
=> '((1 . 3) (5 . 2) (1 . 1) (1 . 4) (3 . 5))
```

[1] https://en.wikipedia.org/wiki/Lights_Out_(game)

[2] https://web.archive.org/web/20090307232754/http://www.haar.clara.co.uk/Lights/example1.html

2) Escriba la función Scheme **encendida?** que dado un tablero y las coordenadas renglón y columna (con valores entre 1 y 5), devuelve **#t** si la luz en esa posición está encendida, o **#f** si está apagada. Si las coordenadas están fuera de rango, la función devuelve **#f**.

```
(encendida? (tablero-inicial-1) 3 5)
=> #t

(encendida? (tablero-inicial-1) 1 2)
=> #f
```

3) Escriba la función Scheme **cambia-luz**, que dado un tablero y las coordenadas renglón y columna (numeradas de 1 a 5), devuelve un tablero con la luz en esa posición cambiada de estado (apagada si estaba encendida y viceversa). Si las coordenadas están fuera de rango, la función debe ignorar el cambio solicitado y devolver el tablero sin cambios.

```
(cambia-luz (tablero-inicial-1) 3 5)
=> '((1 . 3) (5 . 2) (1 . 1) (1 . 4))

(cambia-luz (tablero-inicial-1) 1  2)
=> '((1 . 2) (1 . 3) (5 . 2) (1 . 1) (1 . 4) (3 . 5))
```
(el par **(1 . 2)** puede estar en cualquier posición de la lista resultado)

4) Escriba la función Scheme **pulsa** que toma un tablero y un par de coordenadas como argumentos, y devuelve un tablero donde se ejecutó la acción de pulsado solicitada. En el tablero devuelvo están cambiadas las luces de acuerdo a las reglas del juego.

```
(pulsa (tablero-inicial-1) 1  2)
=> '((2 . 2) (1 . 2) (5 . 2) (1 . 4) (3 . 5))
```

5) Escriba una función Scheme **fin-del-juego?** Que aplicado a un tablero, devuelve **#t** cuando el tablero tiene todas las luces apagadas, y **#f** en otro caso.

```
(fin-del-juego? (tablero-inicial-1))
=> #f
```

6) Escriba una función Scheme **aplica-pulsaciones** que toma un tablero de entrada y una lista de pares con las pulsaciones a aplicar, y que devuelve el tablero resultante. Verifique que para el tablero inicial del gráfico (ver punto 1), al aplicar las pulsaciones sugeridas se obtiene un tablero ganador.

```
(aplica-pulsaciones
   (tablero-inicial-1)
   '((2 . 1) (2 . 3) (2 . 4) (3 . 1) (3 . 5)
    (4 . 2) (4 . 3) (5 . 4) ))
=> '((5 . 5) (5 . 4) (4 . 5))
```

7) Escriba una función Scheme **pulsaciones-ganadoras** que toma como argumento un tablero y devuelve una lista de pares con las pulsaciones a aplicar para ganar el juego. Si no encuentra la solución, devuelve **'no-hay-solucion**. El método de resolución del juego se denomina "*light chasing*", el cual se explica en el apéndice.

   Por ejemplo, para resolver el tablero que se presenta más arriba, pulsamos:
   Fila 2: 1, 3, 4
   Fila 3: 1, 5
   Fila 4: 2, 3
   Fila 5: 4, 5

   Entonces, al evaluar la siguiente expresión:

```
(pulsaciones-ganadoras (tablero-inicial-1))
=> ( (2 . 1) (2 . 3) (2 . 4) (3 . 1) (3 . 5) (4 . 2) (4 . 3) (5 . 4)
(5. 5) )
```

   Esperamos que el resultado obtenido sea la secuencia de pulsaciones que gana para el tablero dado, lo que se puede verificar con la expresión:

```
(let ((tab-inicial '((1 . 3) (5 . 2) (1 . 1) (1 . 4) (3 . 5))))
  (fin-del-juego?
    (aplica-pulsaciones tab-inicial
      (pulsaciones-ganadoras tab-inicial))))
```

   Dicha expresión debe evaluar a **#t**.

Apéndice: Light chasing[3] en detalle

El algoritmo se inicia apagando todas las luces de la fila 1. Para hacer esto debemos pulsar las luces que están en la fila 2, debajo de cada luz encendida de la fila 1. Al concluir este paso, todas las luces de la fila 1 estarán apagadas.

[3] https://web.archive.org/web/20100704161251/http://www.haar.clara.co.uk/Lights/solving.html
Utilizamos como base para las explicaciones esta referencia.

Repetimos este paso para apagar las luces de las filas 2, 3 y 4. En este momento, puede ser que el juego esté resuelto, si no hay luces encendidas en la fila 5.

Si hay luces encendidas en la fila 5, **y el juego tiene solución**, la fila 5 sólo puede ser una de 7 combinaciones diferentes. La jugada continúa pulsando ciertas luces de la primera fila. Cuáles luces se deben pulsar, depende de la combinación que hay en la fila 5. Luego de pulsar las luces correspondientes, se procede pulsando desde la fila 2 nuevamente, como se hizo antes, y al completar la fila 5 todas las luces se apagarán.
En la siguiente tabla se indica para cada combinación de la última fila, cuáles son las luces que deben pulsarse en la primera fila:

Encendidas en la fila 5	Pulsar en la fila 1
💡□□□💡	💡💡□□□
□💡□💡□	💡□□💡□
💡💡💡□□	□💡□□□
□□💡💡💡	□□□💡□
💡□💡💡□	□□□□💡
□💡💡□💡	💡□□□□
💡💡□💡💡	□□💡□□

Si la configuración de la última fila no coincide con ninguna presente en la tabla, entonces el juego no tiene solución. Esta forma de resolver el juego siempre tiene éxito, si se considera que la solución puede tener cualquier cantidad de pulsaciones.

Para ganar con la menor cantidad de pulsaciones hay que jugar una vez de acuerdo al método indicado para averiguar cuáles son las luces que se debe pulsar de la fila 1. En una segunda vuelta se pulsan primero las luces de la fila 1 y se sigue con el método: esta vez la cantidad de pulsaciones usadas es la mínima. Hay sitios *online*[4] donde puede elegir una configuración inicial y hacer las movidas manualmente para verificar su solución, y otros que disponen de un *solver*[5] que encuentra la solución.

[4] http://mathafou.free.fr/jeux_en/jeu305.html editar y jugar online
[5] https://www.geogebra.org/m/JexnDJpt#material/RmRUx8Yd solver