

Evidencia #3: Implementación de un simulador de almacén automatizado (segunda parte)

Diego Andrés Figueroa Peart
A01660987

Curso:
Implementación de métodos computacionales
(TC2037.820)

Profesores:
Román Martínez Martínez
Alberto Oliart Ros
Valentina Narváez Terán
Germán Domínguez Solís

Fecha de entrega:
13 de junio de 2023

Evidencia #3: Carrusel vertical paralelo

En este tercer periodo, la situación problema consistió en modificar la entrega de la segunda evidencia para que esta funcione en el lenguaje de programación funcional Clojure, un lenguaje basado en Lisp pero construido en la plataforma de Java. Esto le permite utilizar los módulos existentes de Java pero con las ventajas de los paradigmas funcional y concurrente.

Recordando la síntesis de la situación problema del segundo periodo, esta consiste en elaborar un simulador de carrusel vertical de almacenamiento, el cual consiste en la siguiente estructura:



Ahora bien, si el simulador entregado como segunda evidencia de este curso funcionaba para simular un solo carrusel y realizar una lista de transacciones sobre él, ahora en Clojure, gracias a sus capacidades de utilizar el paradigma concurrente/paralelo, esta nueva versión del simulador podrá realizar varias listas de transacciones sobre distintos carruseles simultáneamente.

Diseño de las estructuras de datos

Al igual que en la entrega pasada, los carruseles tendrán la siguiente estructura utilizando listas anidadas para representar las filas y los productos individuales:

```
((producto 1 cantidad precio) (producto 2 cantidad precio) (producto 3 cantidad precio))  
((producto 4 cantidad precio) (producto 5 cantidad precio) (producto 6 cantidad precio))  
((producto 7 cantidad precio) (producto 8 cantidad precio) (producto 9 cantidad precio)))
```

En esta ocasión, los carruseles serán almacenados en archivos de texto, uno para cada carrusel, los cuales tendrán los nombres “carrusel#.txt”, en donde # es el número indicador del carrusel. Estos estarán elaborados mediante un programa adicional, que generará combinaciones aleatorias de productos con precios y cantidades aleatorias también:

```
camusel1.txt x camusel2.txt x camusel3.txt x camusel4.txt x
1 (((("Pantoprazol" 240 15.1) ("Budesonida" 154 3.9) ("Cromoglicato de sodio"
248 1.8) ("5-Fluorouracilo" 175 6.9) ("Sertralina" 82 16.6)) (("Amitriptilina"
112 15.3) ("Degarelix" 275 19.3) ("Vincristina" 4 5.8) ("Carboplatino" 128
3.1) ("Ranitidina" 1 4.0)) (("Metoclopramida" 164 18.9) ("Ondansetron" 45
15.3) ("Cetirizina" 28 0.5) ("Dacarbazina" 193 10.0) ("Metformina" 292 13.8))
(("Montelukast" 271 17.1) ("Clorfenamina" 89 2.0) ("Goserelina" 210 14.0)
("Glimepirida" 51 10.0) ("Ketotifeno" 16 5.9)) (("Etanercept" 368 3.1)
("Gliclazida" 273 13.9) ("Letrozol" 1 18.6) ("Ketorolaco" 220 3.4)
("Carbamazepina" 150 16.7)) (("Bevacizumab" 69 4.2) ("Leuprorelina" 0 17.4)
("Clonazepam" 295 18.8) ("Lorazepam" 215 8.0) ("Gabapentina" 274 8.1))
(("Cisplatino" 0 19.1) ("Alprazolam" 183 0.5) ("Trimetoprima" 296 9.8)
("Azitromicina" 261 4.7) ("Insulina aspart" 382 1.6)) (("Interferon beta-1a"
293 4.0) ("Lansoprazol" 215 14.3) ("Rosuvastatina" 44 11.3) ("Vinblastina" 103
6.5) ("Cefalexina" 190 0.7)) (("Loratadina" 102 9.7) ("Ciclofosfamida" 123
15.3) ("Docetaxel" 206 6.0) ("Fexofenadina" 156 18.0) ("Amoxicilina" 257
19.9)) (("Pseudoefedrina" 137 5.3) ("Ceftriaxona" 219 0.7) ("Albuterol" 212
1.7) ("Procabazina" 183 15.3) ("Naproxeno" 165 0.3))) "Bevacizumab")
```

```
camusel1.txt x camusel2.txt x camusel3.txt x camusel4.txt x
1 (((("Desloratadina" 268 4.7) ("Trastuzumab" 210 7.2) ("Aciclovir" 252 18.4)
("Metronidazol" 243 5.0) ("Metoclopramida" 122 3.8)) (("Amitriptilina" 239
1.0) ("Carbamazepina" 117 6.2) ("Clorfenamina" 91 0.0) ("Ranitidina" 156 7.3)
("Fulvestrant" 25 8.2)) (("Leuprorelina" 69 8.7) ("Naproxeno" 252 19.5)
("Infliximab" 4 4.0) ("Ondansetron" 168 2.7) ("Interferon beta-1a" 528 19.5))
(("Insulina detemir" 91 1.7) ("Montelukast" 58 19.9) ("Vincristina" 227 7.0)
("Tamoxifeno" 123 15.2) ("Fexofenadina" 94 7.4)) (("Fluoxetina" 20 4.6)
("Ibuprofeno" 148 3.0) ("Natalizumab" 114 14.5) ("Etanercept" 745 9.0)
("Gabapentina" 0 2.7)) (("Paroxetina" 299 12.7) ("Alemtuzumab" 274 2.3)
("Domperidona" 279 16.8) ("Diazepam" 0 14.0) ("Irinotecan" 94 2.8))
(("Anastrozol" 99 10.5) ("Citalopram" 190 11.0) ("Ceftriaxona" 226 5.7)
("Trimetoprima" 174 18.8) ("Atorvastatina" 144 16.7)) (("Lansoprazol" 221 7.9)
("Ketotifeno" 1 8.4) ("Simvastatina" 12 17.3) ("Ciclofosfamida" 216 9.3)
("Bevacizumab" 92 14.0)) (("Fluticasona" 178 9.0) ("Famotidina" 64 6.6)
("Fenofibrato" 87 14.7) ("Venlafaxina" 267 20.0) ("Oseltamivir" 256 14.6))
(("Pantoprazol" 276 15.4) ("Clonazepam" 19 7.0) ("Goserelina" 100 11.9)
("Rituximab" 102 4.0) ("Doxorrubicina" 1 8.7))) "Gabapentina")
```

```
camusel1.txt x camusel2.txt x camusel3.txt x camusel4.txt x
1 (((("Gabapentina" 41 17.5) ("Atorvastatina" 110 5.3) ("Trimetoprima" 146 0.2)
("Escitalopram" 112 13.2) ("Esomeprazol" 278 3.5)) (("Metronidazol" 26 14.6)
("Lansoprazol" 140 11.6) ("Cefalexina" 16 3.9) ("Anastrozol" 173 14.2)
("Valaciclovir" 159 13.9)) (("Fenofibrato" 489 17.1) ("Salbutamol" 9 2.4)
("Cromoglicato de sodio" 99 7.9) ("Clorfenamina" 268 20.0) ("Amitriptilina"
273 3.0)) (("Ezetimiba" 295 13.3) ("Loratadina" 88 9.3) ("Rituximab" 231 8.0)
("Diazepam" 35 5.2) ("Vincristina" 20 20.0)) (("Simvastatina" 215 9.9)
("Ketorolaco" 112 10.5) ("Degarelix" 31 13.3) ("Rosuvastatina" 212 12.6)
("Interferon beta-1a" 69 10.3)) (("Aspirina" 170 12.7) ("Etanercept" 222 15.4)
("Doxorrubicina" 96 10.1) ("Letrozol" 278 12.0) ("Insulina aspart" 136 16.7))
(("Abiraterona" 183 4.9) ("Naproxeno" 27 6.8) ("Gliclazida" 51 10.1)
("Cetirizina" 219 12.9) ("Aciclovir" 284 9.5)) (("Fluoxetina" 287 3.5)
("Goserelina" 48 8.6) ("Paclitaxel" 45 6.5) ("Lorazepam" 200 16.9)
("Ketotifeno" 106 8.3)) (("Clonazepam" 267 17.1) ("Ciprofloxacino" 190 11.8)
("Ondansetron" 7 9.8) ("Insulina glargina" 184 8.2) ("Fexofenadina" 711 3.4))
(("Irinotecan" 261 4.7) ("Interferon beta-1b" 141 3.0) ("Trastuzumab" 82 5.4)
("Docetaxel" 6 5.3) ("Sertralina" 207 3.1))) "Fenofibrato")
```

```
camusel1.txt x camusel2.txt x camusel3.txt x camusel4.txt x
1 (((("Doxiciclina" 107 0.7) ("Amitriptilina" 129 12.0) ("Aspirina" 128 10.1)
("Interferon beta-1b" 6 7.3) ("Vincristina" 668 15.2)) (("Doxorrubicina" 184
5.1) ("Bleomicina" 77 19.5) ("Lamotrigina" 141 10.8) ("Clonazepam" 289 3.3)
("Lansoprazol" 136 5.4)) (("Atorvastatina" 188 19.9) ("Amoxicilina" 89 8.2)
("Metformina" 31 7.4) ("Citalopram" 101 1.4) ("Omeprazol" 168 3.7))
(("Natalizumab" 27 3.0) ("Lorazepam" 83 3.4) ("Fulvestrant" 15 10.6)
("5-Fluorouracilo" 195 7.2) ("Trimetoprima" 29 17.6)) (("Rituximab" 236 11.4)
("Cromoglicato de sodio" 140 7.8) ("Docetaxel" 138 4.2) ("Abiraterona" 251
3.4) ("Pregabalina" 22 15.7)) (("Ezetimiba" 156 6.2) ("Irinotecan" 253 4.6)
("Degarelix" 262 5.1) ("Fluoxetina" 280 0.9) ("Ketotifeno" 213 16.7))
(("Carbamazepina" 193 0.2) ("Metronidazol" 52 9.0) ("Ranitidina" 7 18.8)
("Ciprofloxacino" 259 16.5) ("Sertralina" 114 14.8)) (("Escitalopram" 2 15.0)
("Ondansetron" 262 10.0) ("Paracetamol" 60 7.1) ("Etanercept" 120 4.3)
("Ciclofosfamida" 168 11.3)) (("Desloratadina" 52 16.6) ("Paclitaxel" 131
17.5) ("Fenofibrato" 267 17.8) ("Insulina lispro" 284 11.7) ("Bevacizumab" 146
18.6)) (("Dacarbazina" 285 18.1) ("Famotidina" 193 4.1) ("Loratadina" 216
10.4) ("Glimepirida" 177 2.7) ("Cisplatino" 15 2.0))) "Vincristina")
```

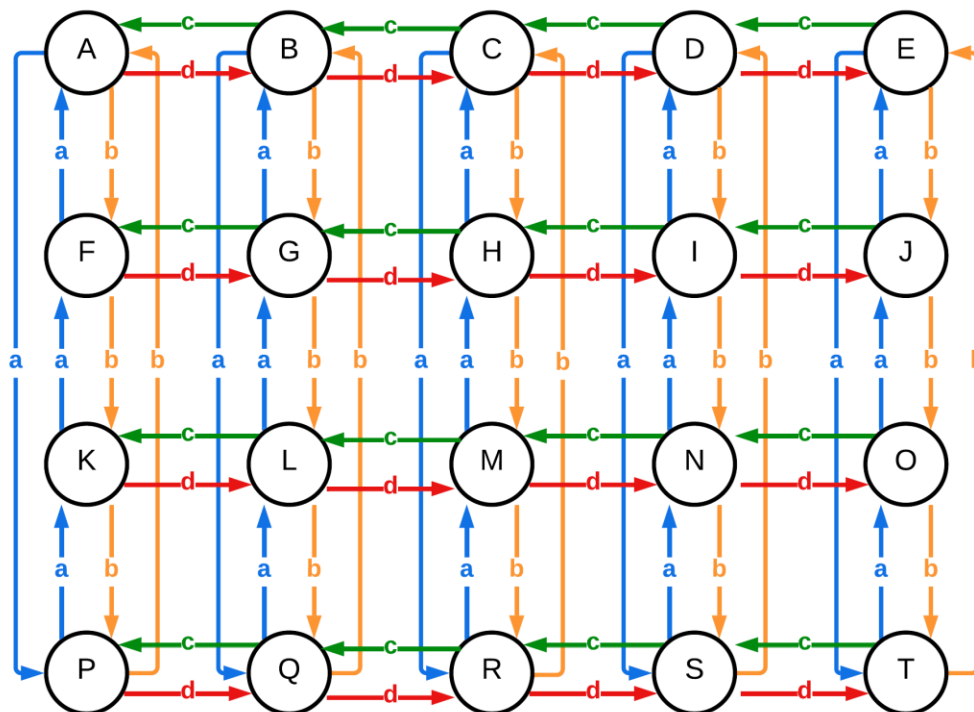
Los estados actuales de cada carrusel estarán adicionados a su respectivo archivo como el segundo elemento en una lista, el primero siendo el carrusel en si.

El movimiento del carrusel utilizará el mismo lenguaje que la entrega pasada; las funciones de movimiento serán las siguientes:

- a, hacia arriba.
- b, hacia abajo.
- c, hacia la izquierda.
- d, hacia la derecha

Diseño del autómata

La representación visual del autómata que rige el comportamiento del carrusel es el mismo, sin embargo, ahora se ejecutarán múltiples versiones de este mismo al mismo tiempo para cada carrusel:



Las transacciones a realizar en cada carrusel serán almacenadas también en archivos únicos para cada carrusel, y también serán generadas automáticamente. Los nombres de los archivos serán "transacciones#.txt", donde # corresponderá al número identificador del carrusel al que se le aplicarán las transacciones. Las transacciones posibles seguirán siendo las siguientes:

- **(a)** – El carrusel se moverá hacia arriba.
- **(b)** – El carrusel se moverá hacia abajo.
- **(c)** – El carrusel se moverá hacia la izquierda.
- **(d)** – El carrusel se moverá hacia la derecha.
- **(retirar cantidad {producto})** – Si no se especifica producto, se retirará la cantidad del producto actualmente mostrado. Si se especifica un producto existente, el carrusel se moverá a la posición del producto especificado, y mostrará la ruta más eficiente para llegar a esta posición. Si se intentan retirar más de las existencias del producto, se retirará todo y se marcará un error.
- **(agregar cantidad {producto})** – Al igual que la función retirar, si no se especifica un producto se agregará al producto actual, y si sí, se moverá el carrusel a la posición del producto utilizando la ruta más eficiente.

Ejemplos de archivos de transacciones:

```
camusel1.txt x camusel2.txt x camusel3.txt x camusel4.txt x transacciones1.txt x transacciones2.txt x
1 ((d) (retirar 43) (a) (d) (c) (c) (d) (agregar 42 "Diazepam") (c) (agregar 13)
(b) (c) (b) (retirar 49) (retirar 48) (agregar 14) (agregar 11 "Insulina
aspart") (c) (agregar 0 "Bevacizumab") (b) (retirar 43) (agregar 39
"Bevacizumab") (retirar 12) (retirar 36))
```

Al finalizar la ejecución de todas las transacciones en todos los carruseles, se desplegarán en pantalla dos datos; el valor total de todo el inventario en todas los carruseles (calculado multiplicando todos los precios unitarios por las cantidades de los productos), y el top 10% de carruseles con mayor valor de inventario.

```
3220 ; Evaluating file: Evidencia 3.clj
3221 Valor total del inventario de todos los carruseles: 3939497.4000000004
3222 Top 10% de carruseles con mayor valor de inventario: ((Carrusel 6
109264.6) (Carrusel 16 106954.29999999999) (Carrusel 27 102920.7)
(Carrusel 12 99555.3) (Carrusel 32 96769.19999999998))
```

Adicionalmente, por cada transaccion que se realice en cada carrusel, se añadirá a un archivo "log#.txt" que almacenará todas las transacciones que se realicen en su respectivo carrusel, y finalmente se desplegarán los productos bajos en inventario (menores a 20 unidades) del respectivo carrusel.

```
1 ("Leuprorelina" 56 17.4)
2 Retiradas 43 unidades de Leuprorelina.
3 ("Gliclazida" 273 13.9)
4 ("Letrozol" 1 18.6)
5 ("Gliclazida" 273 13.9)
6 ("Etanercept" 199 3.1)
7 ("Gliclazida" 273 13.9)
8 Producto Diazepam no encontrado.
9 ("Etanercept" 199 3.1)
10 Agregadas 13 unidades de Etanercept.
11 ("Bevacizumab" 186 4.2)
12 Movimiento inválido.
13 ("Cisplatino" 0 19.1)
14 Se intentaron retirar más de las existencias de Cisplatino.
15 Se intentaron retirar más de las existencias de Cisplatino.
16 Agregadas 14 unidades de Cisplatino.
17 Agregadas 11 unidades de Insulina aspart.
18 Ruta: dddd
19 ("Azitromicina" 261 4.7)
20 Agregadas 0 unidades de Bevacizumab.
21 Ruta: accc
22 ("Cisplatino" 14 19.1)
23 Se intentaron retirar más de las existencias de Cisplatino.
24 Agregadas 39 unidades de Bevacizumab.
25 Ruta: a
26 Retiradas 12 unidades de Bevacizumab.
27 Retiradas 36 unidades de Bevacizumab.
28 Productos con menos de 20 unidades: (((("Vincristina" 4 5.8) ("Ranitidina" 1 4.0) ("Ketotifeno" 16 5.9) ("Letrozol" 1 18.6) ("Leuprorelina" 13 17.4) ("Cisplatino" 0 19.1)))
```

Mutabilidad de los datos

Al igual que la entrega anterior, el simulador de carruseles es completamente dinámico y adaptable automáticamente a cualquier tamaño de carrusel; la única variable que el usuario deberá ajustar antes de ejecutarlo es el número de carruseles que se tienen. Fuera de esto, los carruseles pueden contener distintos productos, número de filas, número de columnas, etc.; mientras no varíe el número de columnas dentro del carrusel, y siga la estructura de datos anteriormente detallada, el código lo manejará sin problemas.

Ventajas y desventajas

La ventaja principal que se me ocurre al haber desarrollado este proyecto en un lenguaje de los paradigmas funcional y concurente/paralelo, como lo es Clojure, es la eficiencia en memoria y rapidez con la que es posible manejar grandes cantidades de datos gracias a la paralelización. Otra de las grandes ventajas de Clojure es que está construido sobre la plataforma de Java, permitiéndole así utilizar el extenso arsenal de módulos y librerías que existen para este lenguaje.

Sin embargo, el sintaxis poco convencional de los lenguajes funcionales, combinado con su dificultad de instalación y ejecución de programas, especialmente en plataformas como Windows, me hace dudar de su practicidad en el campo real, y honestamente me encuentro poco seguro de volver a utilizar este lenguaje más allá de lo académico.

Código

; Evidencia #3

; Implementación de un simulador de un almacén automatizado (segunda parte)

; Diego Andrés Figueroa Peart A01660987

; Se establecen los parámetros básicos de cada carrusel.

```
(defn carrusel [n]
  (first (read-string (slurp (str "carrusel" n ".txt"))))))
```

```
(defn est-actual [n]
  (second (read-string (slurp (str "carrusel" n ".txt"))))))
```

```
(defn filas [n]
  (count (carrusel n)))
```

```
(defn columnas [n]
  (count (first (carrusel n))))
```

; Se guardan los cambios al carrusel.

```
(defn escribir [n txt]
  (spit (str "carrusel" n ".txt") txt))
```

; El log de las transacciones se almacenará por carrusel.

```
(defn log [n txt]
```



```
(spit (str "log" n ".txt") (str txt "\n") :append true))
```

```
(defn coordenadas [lst col x y f]  
  (if (empty? lst) f  
      (if (<= y col)  
          (recur (rest lst) col x (+ y 1) (concat f (list (list (first (first lst)) (list x y)))))  
          (recur lst col (+ x 1) 1 f))))  
(defn diccionario [n]  
  (coordenadas (apply concat (carrusel n)) (columnas n) 1 1 '()))
```

; Se busca un producto a partir de su nombre, se regresan sus coordenadas.

```
(defn buscar [producto n]  
  (let [x (filter #(= producto (first %)) (diccionario n))]  
      (if (nil? x) false  
          (second (first x)))))
```

; Se reciben las coordenadas del producto y se regresa su nombre, cantidad y precio.

```
(defn detalles-aux [lst y]  
  (if (not (= 1 y))  
      (recur (rest lst) (dec y))  
      (first lst)))  
(defn detalles [lst x y]  
  (if (not (= 1 x))  
      (recur (rest lst) (dec x) y)
```

```
(detalles-aux (first lst) y)))
```

; Se cambia la cantidad del producto en el carrusel para luego sobrecribir el archivo.

```
(defn reemplazar-y [lst y r f]
```

```
(cond
```

```
(empty? lst) f
```

```
(= y 1) (recur (rest lst) (dec y) r (concat f (list r)))
```

```
:else (recur (rest lst) (dec y) r (concat f (list (first lst)))))
```

```
(defn reemplazar-x [lst x y r f]
```

```
(cond
```

```
(empty? lst) f
```

```
(= 1 x) (recur (rest lst) (dec x) y r (concat f (list (reemplazar-y (first lst) y r '()))))
```

```
:else (recur (rest lst) (dec x) y r (concat f (list (first lst)))))
```

```
(defn reemplazar [n x y r]
```

```
(reemplazar-x (carrusel n) x y r '()))
```

```
(defn arriba [x1 x2 lst filas]
```

```
(if (= x1 x2) lst
```

```
(if (= x1 1)
```

```
(recur filas x2 (concat lst (list 'a)) filas)
```

```
(recur (- x1 1) x2 (concat lst (list 'a)) filas))))
```

```
(defn abajo [x1 x2 lst filas]
```

```
(if (= x1 x2) lst
```

```

(if (= x1 filas)
  (recur 1 x2 (concat lst (list 'b)) filas)
  (recur (+ x1 1) x2 (concat lst (list 'b)) filas))))

```

; Se calcula la distancia entre el producto actual y el producto al que se desea llegar, se determina la ruta más rápida.

```

(defn distancia-y [y1 y2 lst]
  (if (= y1 y2) lst
    (if (> y1 y2)
      (recur (- y1 1) y2 (concat lst (list 'c)))
      (recur (+ y1 1) y2 (concat lst (list 'd'))))))

```

```

(defn distancia [x1 y1 x2 y2 lst filas]
  (cond
    (< x1 x2) (if (> (- x2 x1) (/ filas 2))
      (recur 0 y1 0 y2 (arriba x1 x2 lst filas) filas)
      (recur 0 y1 0 y2 (abajo x1 x2 lst filas) filas))
    (> x1 x2) (if (> (- x1 x2) (/ filas 2))
      (recur 0 y1 0 y2 (abajo x1 x2 lst filas) filas)
      (recur 0 y1 0 y2 (arriba x1 x2 lst filas) filas))
    :else (distancia-y y1 y2 lst)))

```

; Se regresan los productos con menos de 20 unidades.

```

(defn productos-bajos [n]
  (let [prods (filter #(> 20 (second %)) (apply concat (carrusel n)))]
    (log n (str "Productos con menos de 20 unidades: " (list prods)))))

```

```
(defn valor-total [n]
  (apply + (map #(* (second %) (last %)) (apply concat (carrusel n))))))
```

; Movimiento hacia arriba.

```
(defn a [n]
  (let [coords (buscar (est-actual n) n)]
    (if (zero? (dec (first coords)))
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) (filas n) (second coords)))))
        (log n (detalles (carrusel n) (filas n) (second coords)))))
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) (dec (first coords)) (second coords)))))
        (log n (detalles (carrusel n) (dec (first coords)) (second coords)))))))
```

; Movimiento hacia abajo

```
(defn b [n]
  (let [coords (buscar (est-actual n) n)]
    (if (< (filas n) (inc (first coords)))
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) 1 (second coords)))))
        (log n (detalles (carrusel n) 1 (second coords)))))
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) (inc (first coords)) (second coords)))))
```

```
(log n (detalles (carrusel n) (inc (first coords)) (second coords))))))
```

; Movimiento hacia la izquierda.

```
(defn c [n]
  (let [coords (buscar (est-actual n) n)]
    (if (zero? (dec (second coords)))
      (log n "Movimiento inválido.")
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) (first coords) (dec (second coords))))))
        (log n (detalles (carrusel n) (first coords) (dec (second coords)))))))))
```

; Movimiento hacia la derecha

```
(defn d [n]
  (let [coords (buscar (est-actual n) n)]
    (if (< (columnas n) (inc (second coords)))
      (log n "Movimiento inválido.")
      (do
        (escribir n (list (carrusel n) (first (detalles (carrusel n) (first coords) (inc (second coords))))))
        (log n (detalles (carrusel n) (first coords) (inc (second coords)))))))))
```

(defn agregar

[cantidad n] ; Se agrega la cantidad al producto que se muestra actualmente.

```
(let [coords (buscar (est-actual n) n)
      prod (detalles (carrusel n) (first coords) (second coords))
```

```

    reem (list (first prod) (+ (second prod) cantidad) (last prod))]
(escribir n (list (reemplazar n (first coords) (second coords) reem) (est-actual n)))
(log n (str "Agregadas " cantidad " unidades de " (first prod) "."))))
([cantidad producto n] ; Se agrega la cantidad al producto especificado.
(let [origen (buscar (est-actual n) n)
      destino (buscar producto n)]
  (if (not destino)
    (log n (str "Producto " producto " no encontrado."))
    (let [prod (detalles (carrusel n) (first destino) (second destino))
          reem (list (first prod) (+ (second prod) cantidad) (last prod))]
      (escribir n (list (reemplazar n (first destino) (second destino) reem) producto))
      (log n (str "Agregadas " cantidad " unidades de " (str (first prod)) "."))
      (if (not (= origen destino))
        (log n (str "Ruta: " (apply str (distancia (first origen) (second origen) (first destino) (second destino) '() (filas n))))))))))

```

(defn retirar

([cantidad n] ; Se retira la cantidad del producto que se muestra actualmente.

```

(let [coords (buscar (est-actual n) n)
      prod (detalles (carrusel n) (first coords) (second coords))
      reem (list (first prod) (- (second prod) cantidad) (last prod))]
  (if (> 0 (second reem))
    (do
      (escribir n (list (reemplazar n (first coords) (second coords) (list (first reem) 0 (last reem))) (est-actual n)))
      (log n (str "Se intentaron retirar más de las existencias de " (first prod) "."))))

```

```

(do
  (escribir n (list (reemplazar n (first coords) (second coords) reem) (est-actual n)))
  (log n (str "Retiradas " cantidad " unidades de " (first prod) ".")))))

([cantidad producto n] ; Se retira la cantidad del producto especificado.
(let [origen (buscar (est-actual n) n)
      destino (buscar producto n)]
  (if (not destino)
    (log n (str "Producto " producto " no encontrado."))
    (let [prod (detalles (carrusel n) (first destino) (second destino))
          reem (list (first prod) (- (second prod) cantidad) (last prod))]
      (if (> 0 (second reem))
        (do
          (escribir n (list (reemplazar n (first destino) (second destino) (list (first reem) 0 (last reem))) producto))
          (log n (str "Se intentaron retirar más de las existencias de " (first prod) ".")))
        (do
          (escribir n (list (reemplazar n (first destino) (second destino) reem) producto))
          (log n (str "Retiradas " cantidad " unidades de " (first prod) ".")))
          (if (not (= origen destino))
            (log n (str "Ruta: " (apply str (distancia (first origen) (second origen) (first destino) (second destino) '() (filas n))))))))))

```

; Se leerán y ejecutarán las transacciones línea por línea.

```

(defn main-aux [n transacciones]
  (eval (concat (first transacciones) (list n)))
  (if (not (empty? (rest transacciones)))

```

```
(recur n (rest transacciones))))
```

```
(defn main-p [n]
```

```
(main-aux n (read-string (slurp (str "transacciones" n ".txt"))))))
```

(def num_carruseles 50) ; El número de carruseles a procesar es configurable aquí.

```
(defn main []
```

```
(pmap #(main-p %) (range 1 (inc num_carruseles))) ; Se procesan los carruseles de forma paralela.
```

```
(Thread/sleep 500)
```

```
(pmap #(productos-bajos %) (range 1 (inc num_carruseles))) ; Se añade al log los productos con menos de 20 unidades.
```

```
(Thread/sleep 500)
```

```
(println "Valor total del inventario de todos los carruseles:" (apply + (map #(valor-total %) (range 1 (inc num_carruseles))))) ; Se calcula el valor total del inventario de todos los carruseles.
```

```
(println "Top 10% de carruseles con mayor valor de inventario:" (take (/ num_carruseles 10) (reverse (sort-by second (map #(list (str "Carrusel " %) (valor-total %)) (range 1 (inc num_carruseles))))))))
```

```
(main)
```

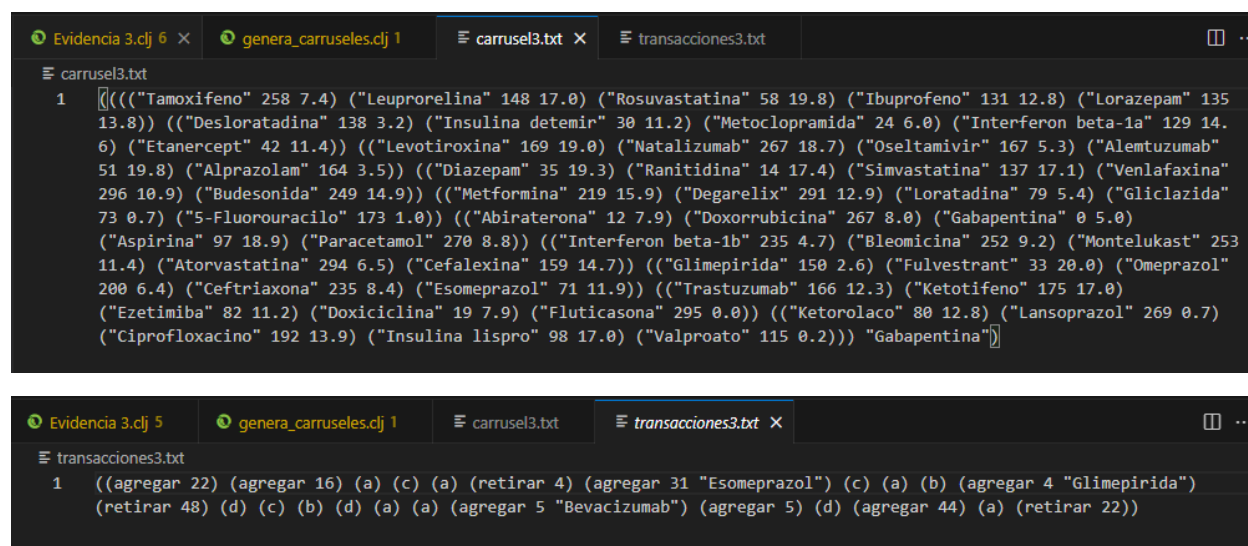

Pruebas realizadas

Para demostrar la efectividad de la paralelización en Clojure, realicé dos pruebas generando 30 carruseles aleatorios y 30 listas de transacciones aleatorias también; una utilizando pmap y la otra con un map regular, midiendo el tiempo que se tomó cada una. Me aseguré que todos los carruseles se encontraran en exactamente el mismo estado inicial para que las condiciones de ejecución de ambas funciones fueran idénticas. Estos fueron los resultados, el primero con map y el segundo con pmap:

```
3358 "Elapsed time: 1.1446 msecs"
3359 Valor total del inventario de todos los carruseles: 2263125.9999999995
3360 Top 10% de carruseles con mayor valor de inventario: ((Carrusel 12
106290.600000000002) (Carrusel 10 94929.400000000002) (Carrusel 1 91165.
29999999997))
3361 nil
3362 clj:user:>
3363 ; Evaluating file: Evidencia 3.clj
3364 "Elapsed time: 0.1545 msecs"
3365 Valor total del inventario de todos los carruseles: 2261423.3999999994
3366 Top 10% de carruseles con mayor valor de inventario: ((Carrusel 12
106190.300000000003) (Carrusel 10 95292.500000000003) (Carrusel 1 91165.
29999999997))
3367 nil
```

Aquí expongo algunos de los carruseles en esta prueba, sus respectivas listas de transacciones y sus archivos log:

Carrusel 3:



```
Evidencia 3.clj 6 x  genera_carruseles.clj 1  carrusel3.txt x  transacciones3.txt ...
carrusel3.txt
1 [((((("Tamoxifeno" 258 7.4) ("Leuprorelina" 148 17.0) ("Rosuvastatina" 58 19.8) ("Ibuprofeno" 131 12.8) ("Lorazepam" 135
13.8)) (("Desloratadina" 138 3.2) ("Insulina detemir" 30 11.2) ("Metoclopramida" 24 6.0) ("Interferon beta-1a" 129 14.
6) ("Etanercept" 42 11.4)) (("Levotiroxina" 169 19.0) ("Natalizumab" 267 18.7) ("Oseltamivir" 167 5.3) ("Alemtuzumab"
51 19.8) ("Alprazolam" 164 3.5)) (("Diazepam" 35 19.3) ("Ranitidina" 14 17.4) ("Simvastatina" 137 17.1) ("Venlafaxina"
296 10.9) ("Budesonida" 249 14.9)) (("Metformina" 219 15.9) ("Degarelix" 291 12.9) ("Loratadina" 79 5.4) ("Gliclazida"
73 0.7) ("5-Fluorouracilo" 173 1.0)) (("Abiraterona" 12 7.9) ("Doxorrubicina" 267 8.0) ("Gabapentina" 0 5.0)
("Aspirina" 97 18.9) ("Paracetamol" 270 8.8)) (("Interferon beta-1b" 235 4.7) ("Bleomicina" 252 9.2) ("Montelukast" 253
11.4) ("Atorvastatina" 294 6.5) ("Cefalexina" 159 14.7)) (("Glimepirida" 150 2.6) ("Fulvestrant" 33 20.0) ("Omeprazol"
200 6.4) ("Ceftriaxona" 235 8.4) ("Esomeprazol" 71 11.9)) (("Trastuzumab" 166 12.3) ("Ketotifeno" 175 17.0)
("Ezetimiba" 82 11.2) ("Doxiciclina" 19 7.9) ("Fluticasona" 295 0.0)) (("Ketorolaco" 80 12.8) ("Lansoprazol" 269 0.7)
("Ciprofloxacino" 192 13.9) ("Insulina lispro" 98 17.0) ("Valproato" 115 0.2))) "Gabapentina"]

Evidencia 3.clj 5  genera_carruseles.clj 1  carrusel3.txt  transacciones3.txt x  ...
transacciones3.txt
1 ((agregar 22) (agregar 16) (a) (c) (a) (retirar 4) (agregar 31 "Esomeprazol") (c) (a) (b) (agregar 4 "Glimepirida")
(retirar 48) (d) (c) (b) (d) (a) (a) (agregar 5 "Bevacizumab") (agregar 5) (d) (agregar 44) (a) (retirar 22))
```

```
Evidencia 3.clj 6  genera_carruseles.clj 1  carrusel3.txt  transacciones3.txt  log3.txt x  ...
log3.txt
1  Agregadas 22 unidades de Tamoxifeno.
2  Agregadas 16 unidades de Tamoxifeno.
3  ("Ketorolaco" 80 12.8)
4  Movimiento inválido.
5  ("Trastuzumab" 170 12.3)
6  Retiradas 4 unidades de Trastuzumab.
7  Agregadas 31 unidades de Esomeprazol.
8  Ruta: adddd
9  ("Ceftriaxona" 235 8.4)
10 ("Atorvastatina" 294 6.5)
11 ("Ceftriaxona" 235 8.4)
12 Agregadas 4 unidades de Glimepirida.
13 Ruta: ccc
14 Retiradas 48 unidades de Glimepirida.
15 ("Fulvestrant" 33 20.0)
16 ("Glimepirida" 150 2.6)
17 ("Trastuzumab" 166 12.3)
18 ("Ketotifeno" 175 17.0)
19 ("Fulvestrant" 33 20.0)
20 ("Bleomicina" 247 9.2)
21 Producto Bevacizumab no encontrado.
22 Agregadas 5 unidades de Bleomicina.
23 ("Montelukast" 209 11.4)
24 Agregadas 44 unidades de Montelukast.
25 ("Gabapentina" 4 5.0)
26 Se intentaron retirar más de las existencias de Gabapentina.
27 Productos con menos de 20 unidades: (("Ranitidina" 14 17.4) ("Abiraterona" 12 7.9) ("Gabapentina" 0 5.0)
("Doxiciclina" 19 7.9))
```

Carrusel 15:

```
Evidencia 3.clj 6  genera_carruseles.clj 1  carrusel15.txt x  ...
carrusel15.txt
1  (((("Amoxicilina" 256 18.5) ("Doxiciclina" 265 3.7) ("Vinblastina" 8 1.6) ("Duloxetina" 120 14.8) ("Fluoxetina" 176 11.
7))) (("Docetaxel" 105 1.2) ("Lansoprazol" 6 6.2) ("Simvastatina" 220 2.7) ("Loratadina" 145 3.3) ("Lorazepam" 136 8.2))
(("Interferon beta-1a" 212 5.5) ("Fexofenadina" 134 6.2) ("Montelukast" 126 5.5) ("Letrozol" 215 9.0) ("Ketotifeno" 200
11.2)) (("Infliximab" 274 0.4) ("Paracetamol" 212 5.9) ("Pantoprazol" 102 4.3) ("Oseltamivir" 283 5.1) ("Albuterol" 283
2.9)) (("Azitromicina" 175 2.6) ("Goserelina" 74 4.8) ("Natalizumab" 102 19.5) ("Sertralina" 212 2.1) ("Ciclofosfamida"
28 11.8)) (("Metoclopramida" 93 5.0) ("Insulina aspart" 165 2.6) ("Ibuprofeno" 204 7.9) ("Fenofibrato" 40 13.9)
("Venlafaxina" 256 7.7)) (("Cisplatino" 232 6.0) ("Famotidina" 124 11.2) ("Carboplatino" 57 7.0) ("Doxorrubicina" 298
11.8) ("Metronidazol" 244 1.8)) (("Ondansetron" 242 11.1) ("Diazepam" 221 13.1) ("Ceftriaxona" 268 3.3) ("Insulina
lispro" 51 10.0) ("Rosuvastatina" 72 9.5)) (("Aciclovir" 265 16.5) ("Paroxetina" 85 16.5) ("Clonazepam" 298 1.0)
("Trimetoprima" 205 2.0) ("Oximetazolina" 171 17.5)) (("Vincristina" 127 10.7) ("Dacarbazina" 231 2.6) ("Cetirizina" 56
16.0) ("Clorfenamina" 130 19.8) ("Pseudoefedrina" 154 19.1))) "Paroxetina")
```

```
Evidencia 3.clj 6  genera_carruseles.clj 1  transacciones15.txt x  ...
transacciones15.txt
1  ((c) (agregar 29 "Fexofenadina") (a) (retirar 35) (b) (retirar 49) (retirar 46) (retirar 47) (c) (d) (a) (retirar 46)
(d) (d) (c) (agregar 16 "Diazepam") (retirar 3) (c) (c) (d) (b) (retirar 3) (a) (b) (agregar 12))
```

```
Evidencia 3.cj 6  genera_carruseles.cj 1  log15.txt X
log15.txt
1  Movimiento inválido.
2  Agregadas 29 unidades de Fexofenadina.
3  Ruta: bbd
4  ("Lansoprazol" 87 6.2)
5  Retiradas 35 unidades de Lansoprazol.
6  ("Fexofenadina" 276 6.2)
7  Retiradas 49 unidades de Fexofenadina.
8  Retiradas 46 unidades de Fexofenadina.
9  Retiradas 47 unidades de Fexofenadina.
10 ("Interferon beta-1a" 212 5.5)
11 ("Fexofenadina" 134 6.2)
12 ("Lansoprazol" 52 6.2)
13 Retiradas 46 unidades de Lansoprazol.
14 ("Simvastatina" 220 2.7)
15 ("Loratadina" 145 3.3)
16 ("Simvastatina" 220 2.7)
17 Agregadas 16 unidades de Diazepam.
18 Ruta: aaaac
19 Retiradas 3 unidades de Diazepam.
20 ("Ondansetron" 242 11.1)
21 Movimiento inválido.
22 ("Diazepam" 221 13.1)
23 ("Paroxetina" 76 16.5)
24 Retiradas 3 unidades de Paroxetina.
25 ("Diazepam" 221 13.1)
26 ("Paroxetina" 73 16.5)
27 Agregadas 12 unidades de Paroxetina.
28 Productos con menos de 20 unidades: (((("Vinblastina" 8 1.6) ("Lansoprazol" 6 6.2)))
```

Carrusel 28:

```
Evidencia 3.cj 6  genera_carruseles.cj 1  carrusel28.txt X
carrusel28.txt
1  (((("Bleomicina" 201 13.0) ("Etanercept" 69 7.9) ("Aspirina" 0 15.4) ("Lorazepam" 193 15.1) ("Escitalopram" 266 18.9))
  ("Naproxeno" 173 19.4) ("Irinotecan" 84 1.8) ("Albuterol" 266 1.2) ("Clorfenamina" 28 8.5) ("Vincristina" 255 9.8))
  ("Pseudoefedrina" 182 17.5) ("Amitriptilina" 241 16.6) ("Rosuvastatina" 170 11.1) ("Tamoxifeno" 5 12.3) ("Insulina
  aspart" 172 10.0)) (("Pantoprazol" 168 10.4) ("Famotidina" 28 19.9) ("Adalimumab" 171 15.3) ("Aciclovir" 142 8.1)
  ("Atorvastatina" 46 3.8)) (("Azitromicina" 124 7.3) ("Ketotifeno" 280 18.3) ("Leuprorelina" 0 14.9) ("Alemtuzumab" 29 8.
  0) ("Budesonida" 28 6.1)) (("Procarbazina" 196 2.0) ("Valproato" 141 2.7) ("Rituximab" 85 9.0) ("Letrozol" 117 16.0)
  ("Oximetazolina" 275 11.6)) (("Diazepam" 276 19.0) ("Levotiroxina" 200 5.7) ("Glimepirida" 222 0.7) ("Natalizumab" 50
  13.1) ("Metronidazol" 175 18.5)) (("Salbutamol" 112 12.2) ("Clonazepam" 239 3.8) ("Abiraterona" 0 7.0) ("Cefalexina"
  285 4.0) ("Metformina" 197 8.3)) (("Interferon beta-1a" 84 1.1) ("Paracetamol" 43 10.9) ("Dacarbazina" 296 15.0)
  ("Cetirizina" 53 1.8) ("Fluticasona" 49 11.7)) (("Interferon beta-1b" 168 1.0) ("Bevacizumab" 231 11.0) ("Infliximab"
  140 11.1) ("Paclitaxel" 143 1.4) ("Degarelix" 61 8.5))) "Escitalopram")
```

```
Evidencia 3.cj 6  genera_carruseles.cj 1  transacciones28.txt X
transacciones28.txt
1  ((d) (d) (retirar 33) (retirar 41) (agregar 42 "Levotiroxina") (d) (c) (c) (retirar 12) (agregar 23) (b) (a) (c)
  (retirar 9) (retirar 31) (agregar 5) (d) (d) (agregar 39 "Degarelix") (agregar 34 "Lamotrigina") (agregar 10
  "Desloratadina") (b))
```

```
Evidencia 3.clj 6  genera_carruseles.clj 1  log28.txt x
log28.txt
1  ("Etanercept" 69 7.9)
2  ("Aspirina" 63 15.4)
3  Retiradas 33 unidades de Aspirina.
4  Se intentaron retirar más de las existencias de Aspirina.
5  Agregadas 42 unidades de Levotiroxina.
6  Ruta: aaaac
7  ("Glimepirida" 222 0.7)
8  ("Levotiroxina" 200 5.7)
9  ("Diazepam" 300 19.0)
10 Retiradas 12 unidades de Diazepam.
11 Agregadas 23 unidades de Diazepam.
12 ("Salbutamol" 112 12.2)
13 ("Diazepam" 311 19.0)
14 Movimiento inválido.
15 Retiradas 9 unidades de Diazepam.
16 Retiradas 31 unidades de Diazepam.
17 Agregadas 5 unidades de Diazepam.
18 ("Levotiroxina" 200 5.7)
19 ("Glimepirida" 222 0.7)
20 Agregadas 39 unidades de Degarelix.
21 Ruta: bbbdd
22 Producto Lamotrigina no encontrado.
23 Producto Desloratadina no encontrado.
24 ("Escitalopram" 266 18.9)
25 Productos con menos de 20 unidades: (((("Aspirina" 0 15.4) ("Tamoxifeno" 5 12.3) ("Leuprorelina" 0 14.9) ("Abiraterona"
0 7.0)))
```

Experiencia de aprendizaje

Honestamente, esta evidencia me resultó increíblemente difícil de completar; tuve problemas en todos lados, desde instalar y correr Clojure en mi computadora, probar mi código en Replit, lograr que funcionara su integración con Visual Studio Code y Calva/Leiningen, y finalmente la lógica en sí de programar en paralelo. Fue un trabajo arduo y complejo pero me siento increíblemente orgulloso de mi mismo por haber logrado perseverar y solucionar la evidencia de forma completa.