Ejercicio de parcial de Rep& Abs

Elías Cerdeira & Román Gorojovsky

4 de Junio de 2025

Introducción

Hoy íbamos a resolver en clase este ejercicio de Rep & Abs, que es el que tomamos en el recuperatorio del 2do parcial del cuatrimestre pasado. Los ejercicios tradicionales de este tema consisten en presentar un TAD, una implementación y pedirles que escriban el invariante y la función de abstracción. Ejemplos de esto son los ejercicios 1 a 3 de la práctica 7. Desde el cuatrimestre pasado estamos tomando ejercicios como este que presentamos hoy, (4 de la práctica) que reducen tanto el tiempo que ustedes tienen que usar para resolver el ejercicio en el parcial como el tiempo que tenemos nosotros para corregir.

Dicho eso, la mecánica general para escribir tanto un Invariante como su función de abstracción es siempre la misma:

- Invariante de representación
  - No hay una "receta" para saber si se especificó el invariante completo, pero se pueden seguir algunos patrones en orden hasta que le tomen la mano.
  - Pueden tomar todas las variables individualmente y ver si necesitamos especificar algo de ellas, alguna restricción que tenga sentido en el contexto de uso. Por ejemplo, si una variable de tipo entero tiene un rango de valores específico, si un vector debería estar ordenado, si un diccionario debería tener todas significados con cierta característica o, como se vio en el ejercicio de la Agenda, que los arreglos que corresponden a cada día tengan 24 horas.
  - Luego, tomar todas las variables de a pares y, si corresponde, escribir cómo se relacionan los datos en cada una. Repetir este procedimiento de a tríos y demás.
  - Puede pasar que queden cosas redundantes. En ese caso, lo correcto es sacarlas. Pero tengan cuidado, porque por ejemplo, cuando decimos que dos variables deben tener los mismos elementos (por ejemplo, las claves de un diccionario deben coincidir con los elementos en una cola de prioridad) hay que decirlo "en los dos sentidos".
  - **Importante:** Es un error común especificar en el invariante de representación de un módulo cosas que corresponden a los módulos que usan. Por ejemplo, si se usa un Conjunto, no hace falta especificar que los elementos no son repetidos.
- Función de abstracción
  - Cuando les pedimos implementar un módulo a partir de un TAD, es común que los observadores tiendan a relacionarse uno a uno, pero no es el caso en estos ejercicios
  - El ABS requiere que se cumpla el invariante del módulo. Por lo tanto si hay información que en el módulo está redundante y en el TAD está una sola vez, alcanza relacionar una de las variables del módulo con el observador correspondiente

Enunciado

Ejercicio 4. Dados el siguiente TAD y su implementación

Ingrediente, *Receta* es $\mathbb{Z}$

```
TAD Cocina {
   obs recetario: dict⟨Receta, conj⟨Ingrediente⟩⟩
   obs stock: dict⟨Ingrediente, int⟩
   proc iniciarActividad() : Cocina
         asegura {Ambos diccionarios comienzan vacíos}
   proc nuevaReceta(inout c : Cocina, in r : Receta, in is : conj⟨Ingrediente⟩)
         requiere {La receta no está registrada previamente y el conjunto no es vacío}
         asegura {Se registra la receta con los ingredientes requeridos}
   proc comprarIngrediente(inout c : Cocina, in i : Ingrediente)
         asegura {Se le agrega una existencia al ingrediente comprado}
}
```

```
Módulo CocinaImpl implementa Cocina <
  var ingredientesPorReceta: Diccionario<Receta, Conjunto<Ingrediente>>
   // Todas las recetas con los ingredientes que son necesarios. Cada ingrediente consume una sola existencia
  var existencias: Array<Ingrediente>
   // Todos los ingredientes que se poseen. Cada ingrediente aparece tantas veces como existencias se tengan
  var ingMasAbundante: ColaDePrioridadMax<Tupla<Ingrediente, int>>
   // Cola de prioridad ordenada de acuerdo a la cantidad que se tiene de cada ingrediente
  var abundancias: Diccionario<int,Conjunto<Ingrediente>>
   // Todas las cantidades con el conjunto de los ingredientes que tienen esas existencias
>
```

a) Indique si cada una de las sentencias propuestas pertenecen al invariante de representación para la estructura propuesta
   i) Dadas dos recetas distintas en **ingredientesPorReceta** sus conjuntos de ingredientes son disjuntos
   ii) Todo ingrediente asociado a una receta en **ingredientesPorReceta** debe estar presente en **existencias**
   iii) Todas las claves registradas en **abundancias** son mayores que cero
   iv) Dados dos cantidades registradas en **abundancias** sus conjuntos de ingredientes son disjuntos
   v) Todo ingrediente presente en **existencias** debe ser ingrediente de alguna receta registrada en **ingredientesPorReceta**

b) Proponga todas las sentencias en castellano que falten para incluir todas las restricciones correspondientes a **ingMasAbundante** en el módulo

c) Escribir la función de abstracción en lógica de primer orden

Resolución

a) Como ven, en este ejercicio no hay que escribir el invariante en lógica, sólo marcar las opciones correctas y luego completarlo en castellano

   i) **Incorrecta**: Pueden haber recetas que compartan ingredientes.

   ii) **Incorrecta**: El enunciado dice que los ingredientes aparecen repetidos tantas veces como corresponda.

   iii) **Correcta**: No guardamos ingredientes que no tengamos en existencia.

   iv) **Correcta**: No podemos tener para el mismo ingrediente distintas cantidades.

   v) **Incorrecta**: Puedo tener ingredientes que no se usen en ninguna receta. No hay requerimientos ni en *nuevaReceta* ni en *comprarIngrediente* que lo fuercen.

b) Falta agregar que:

   i) Una tupla pertenece a `ingMasAbundante` si y sólo si su primera coordenada es un ingrediente que aparece en `existencias` y su segunda coordenada tiene la cantidad de veces que aparece ese ingrediente en `existencias`.

   ii) Si una tupla pertenece a `ingMasAbundante` su segunda componente es clave de `abundancias` y su primera componente debe estar en el conjunto asociado a esa clave. Esta condición debe ser recíproca entre las dos variables.

c) En este caso el Abs sí lo pedimos en lógica. La primer cláusula relaciona *recetario* y `ingredientesPorReceta` y la segunda relaciona *existencias* y `stock`:

`pred Abs` $(t : Cocina, m : CocinaImpl)$ {

$$
\begin{array}{l}
(\forall r : Receta)(r \in t.recetario \leftrightarrow \\
\quad r \in m.ingredientesPorReceta.elems \wedge_L \\
\quad\quad r \in t.recetario \rightarrow_L t.recetario[r] = m.ingredientesPorReceta.Obtener(r)) \\
\wedge \\
(\forall i : Ingrediente)(i \in t.stock \leftrightarrow \\
\quad \#apariciones(m.existencias, i) > 0 \wedge_L \\
\quad\quad i \in t.stock \rightarrow_L \#apariciones(m.existencias, i) = t.stock[i])
\end{array}
$$

}