UTN – FRBA – Paradigmas de Programación

Ejercicios – Objetos – guía 3

Colecciones

Ejercicio 1 – alquiler de vehículos

Una empresa de alquiler de vehículos maneja bicicletas, motos y autos.
De cada vehículo interesa el gasto cada 100 km, la velocidad máxima y la cantidad de pasajeros que pueden llevar.

Para las bicicletas:

- la velocidad máxima es (rodado de la bicicleta 1.2), p.ej. para una bicicleta de rodado 20 la velocidad máxima es 24 km/h.
- el gasto cada 100 km es 1 peso
- pueden llevar un pasajero.

Para las motos:

- la velocidad máxima es (cilindrada / 5), p.ej. para una moto de 1000 cm3 de cilindrada la velocidad máxima es 200 km/h.
- el gasto cada 100 km es 5 pesos + (cilindrada / 200), p.ej. para una moto de 1000 cm3 de cilindrada el gasto cada 100 km es 10 pesos.
- las motos de hasta 150 cm3 de cilindrada pueden llevar un pasajero, las de mayor cilindrada pueden llevar dos pasajeros.

Para los autos:

- la velocidad máxima y la cantidad de pasajeros que pueden llevar son específicas de cada auto.
- el gasto cada 100 km es 20 pesos + (capacidad en pasajeros 10), p.ej. para un auto que lleva 4 pasajeros el gasto cada 100 km es 60 pesos.

Se pide conocer

1. los vehículos con velocidad máxima mayor a un parámetro.
2. los vehículos que cada 100 km consuman menos que un parámetro.
3. el vehículo con mayor coeficiente de eficiencia, que se calcula así
   (cant. pasajeros * velocidad máxima) / gasto cada 100 km
4. la cantidad total de pasajeros que pueden transportar los vehículos de la empresa a más de cierta velocidad, p.ej. "¿cuántos pasajeros podemos llevar a más de 80 km/h?".

Ejercicio 2 – más sobre atención de animales

Agregar las siguientes funcionalidades al ejercicio sobre atención de animales de la guía de trabajo sobre clases.

a.
Agregar lo que haga falta para que algún objeto pueda responder, dado un dispositivo:

- si un animal o no fue atendido
- cuántos animales fueron atendidos
- el conjunto de los animales atendidos que conviene vacunar
- el animal más pesado que atendió
- el peso total de los animales atendidos

También debe existit la capacidad de hacer, mediante un mensaje a un dispositivo, que atienda de nuevo a todos los animales que atendió.

b.
Cómo harías, desde un workspace que conoce a dos dispositivos, para obtener:

- el conjunto de animales que fue atendido en los dos
- el conjunto de animales que fue atendido en uno sí y en el otro no

Pensar en operaciones de conjuntos.

c.
Cómo harías, desde un workspace que conoce a un conjunto de dispositivos, para obtener el conjunto con el animal más pesado de cada dispositivo.

d.
Para cada estación de servicio, pasar de 3 dispositivos a una cantidad indeterminada de dispositivos, y agregar estos requerimientos:

- saber cuáles de los dispositivos de una estación necesitan recarga.
- obtener para una estación, el último animal atendido en cada dispositivo (se espera un conjunto de animales como respuesta).

Ejercicio 3 – carga de camiones

Una empresa de transporte de cargas necesita un software que la ayude a organizarse con el llenado de las camiones que maneja. La empresa tiene varios depósitos, en cada depósito hay varios camiones.

La empresa puede recibir bultos, cajas sueltas y bidones que transportan líquido.

Un bulto es una estructura de madera que arriba tiene un montón de cajas, se envuelve todo con plástico para que no se desbanden las cajas. Todas las cajas de un bulto pesan lo mismo.
El peso de un bulto es

(peso de cada caja * cant. cajas) + peso de la estructura de madera que va abajo.

P.ej. un bulto de 50 cajas de 12 kg cada una con una estructura de 70 kg pesa en total 670 kg.
Para cada bulto se informa qué llevan las cajas, p.ej. ketchup. Todas las cajas de un mismo bulto llevan lo mismo.

De cada caja suelta se informa el peso indvidualmente, son todas distintas. También se informa qué llevan, igual que los bultos.
El peso de un bidón es su capacidad en litros por la densidad (o sea, cuántos kg pesa un litro) del líquido que se le carga. P.ej. si a un bidón de 200 l lo lleno de aceite, y el aceite tiene densidad 0,8; entonces su peso es 160 kg. Los bidones van siempre llenos hasta el tope.

Cada camión puede llevar hasta una carga máxima medida en kg. Además, cada camión puede: estar disponible para la carga (en cuyo caso ya puede tener cosas cargadas), estar en reparación, o estar de viaje.

Se pide:

a. cargar el camión con un "coso", donde el coso puede ser un bulto, una caja suelta, o un bidón.
b. saber si un camión puede aceptar un coso. Un camión puede aceptarlo si con lo que le preguntan + lo que ya tiene cargado no supera su carga máxima, y además está disponible para la carga.
c. modificar la carga del camión para que solo se cargue si puede hacerse. Por ejemplo si un camión con capacidad para 150 kg que ya tenia 140 kg se le pide que cargue un bulto de 10 kg lo haga, pero si se pide que cargue un bidon de 25 kg no lo haga porque con este superaría su capacidad.
d. registrar estos eventos:
    - un camión sale de reparación, en cuyo caso queda disponible para la carga.
    - un camión entra en reparación
    - un camión sale de viaje.
    - un camión vuelve de viaje, en cuyo caso queda disponible para la carga.
e. saber si un camión está listo para partir, que es: si está disponible para la carga, y el peso total de lo que tiene cargado es de al menos 75% de su carga máxima.
f. saber para un depósito el total de carga que está viajando, o sea la suma de lo que llevan todos los camiones de ese depósito que están de viaje.
g. saber los elementos que están cargados en un determinado camión.
h. saber en qué camiones de un depósito se están cargando un determinado elemento, p.ej. ketchup o aceite. Decimos que un camión se está cargando cuando no está listo para partir.
i. dados dos camiones obtener los elementos que se transportan en los dos. Por ejemplo, si yo tengo un camión que transporta galletitas, raids, alfajores y aceite y tengo otro que transporta agua, galletitas, dentífrico y aceite lo que debería obtener es un conjunto con aceite y galletitas.
j. para un camión saber el coso (bulto, caja o bidón) más liviano que está siendo transportado.
k. para un depósito saber el camión que transporta mayor cantidad de cosos.
l. obtener para dos depósitos el conjunto de productos que están almacenados en los dos ordenados por el peso de cada producto en forma ascendente.
m. lo complicamos un poquito mas
    - ahora hay camiones reutilizables que tienen uno o muchos destinos, de cada destino saben que cosos (bulto, bidón o caja) de los que transportan a cada uno, modelar esta situación.
    - realizar las operaciones correspondientes para que un camión reutilizable llegue a destino y descargue sólo lo que le corresponde a ese destino (tengan en cuenta que en el destino se tiene que actualizar los cosos que tiene almacenado), luego que siga viaje a otro depósito, teniendo en cuenta que puede ir a cualquier depósito de los que todavía no entregó la mercadería
    - ahora también hay camiones frigoríficos que pueden transportar productos hasta una cierta temperatura maxima (además de la restricción de peso que se sigue

manteniendo), la temperatura máxima es la misma para todos los camiones frigoríficos pero puede cambiar, una vez que cambia para uno cambia para todos.
- todos los productos tienen una temperatura máxima que pueden soportar.
- la temperatura máxima soportada por un coso es la menor temperatura de los productos que contiene. Por ejemplo si en un bulto hay lechuga cuya temperatura maxima es 17º y manteca cuya temperatura máxima son 10º, la temperatura maxima de ese bulto será de 10º.
- para que un camión frigorífico pueda cargar un coso la temperatura máxima de ese coso tiene que ser menor a su propia temperatura máxima, y además cumplirse el resto de condiciones que se cumplen para los camiones comunes.

Ejercicio 4 – reservas aéreas

a.
Definir e implementar los objetos que modelan las reservas de un vuelo en una aplicación para una aerolínea, de acuerdo a los requerimientos que se indican.
La aerolínea maneja vuelos, cada vuelo tiene una cantidad de asientos.

Lo que se necesita es:
- saber cuántos asientos disponibles tiene un vuelo.
- reservar una cantidad de asientos para un vuelo. El único efecto hasta acá es que baja la cantidad de asientos disponibles.

b.
Agregar la organización de asientos en filas y columnas, ahora
- cada asiento tiene una identificación dada por fila y columna; p.ej. fila 14 columna "A".
- también se distingue entre ventanilla y pasillo.
- cuando se hace una reserva, se indica cuántos de los asientos son para ventanilla y cuántos para pasillo. Los asientos se asignan en forma automática, sin que nos importe en qué orden se asignan, respetando las indicaciones indicadas en la reserva respecto de ventanilla y pasillo.

Agregar la posibilidad de averiguar
- si una reserva puede o no hacerse, p.ej. si pido 3 de ventanilla y 5 de pasillo, y el vuelo tiene 2 lugares libres la reserva no puede hacerse, y tampoco si 20 lugares libres pero todos en pasillo.
- qué asientos fueron asignados a una reserva.
- los asientos disponibles de un vuelo. No la cantidad sino el conjunto de asientos.
- si una fila de un vuelo (p.ej. la fila 14) está llena.
- cuántos asientos de ventanilla libres tiene un vuelo.
- Las filas en las que hay al menos un asiento disponible.

c.
Agregar métodos que permitan saber cantidad de asientos disponibles y reservar tomando en cuenta sólo los asientos que cumplen una determinada condición (p.ej. sólo entre las filas 8 y 15), además de mantener la condición "tantos de ventanilla y tantos de pasillo".
¿Qué objeto usarían para representar una condición que debe cumplir un asiento?

d.
Implementar la clase VueloBuilder cuyas instancias entienden los siguientes mensajes:

```
enCadaFilaPasillo: cuantosPasillo ventanilla: cuantosVentanilla
   "me dicen cuantos asientos tienen que tener en cada fila
    los vuelos que instancie"
filas: cuantasFilas
   "me dicen cuantas filas tienen que tener los vuelos que instancie"
buildVuelo
   "creo un vuelo y lo devuelvo"
```

e.
La complicamos un poco más.
Ahora sabemos que muchos vuelos se hacen para el mismo avión, entonces para todos los vuelos que se hagan con el mismo avión, los asientos disponibles son los mismos.
Los reservados claramente no, las reservas se hacen para un vuelo, no para un avión.
También al hacer una reserva se indica si las personas que van a viajar van a comer comida normal, comida vegetariana, o comida kosher[1]. Todas las personas incluidas en la misma reserva comen el mismo tipo de comida.
Y ya que estamos, se registra el mail de la persona que hizo la reserva.
Entonces:

- arreglar lo que se hizo para los asientos los tengan los aviones, no los vuelos, pero que siga andando todo lo anterior. El VueloBuilder se transforma en AvionBuilder.
- saber cuántos menúes normales, vegetarianos y kosher hay que subir a un vuelo.
- obtener el conjunto de mails de los que hicieron las reservas de un vuelo.

[1] Preparada de acuerdo a las normas de alimentación judías.