Ejercicios de repaso – I2

Clase #20

IIC1103 – Introducción a la Programación

Marcos Sepúlveda (marcos@ing.puc.cl)

Contenidos a evaluar en la I2

- Todo!

- Listas
- Archivos
- Programación Orientada a Objetos

Ejercicio 1: midterm – 2014.2 – pregunta 4

- Una aerolínea nacional guarda sus vuelos en una lista llamada **`vuelos`**. Cada entrada de la lista es a su vez una lista que contiene el número y la fecha de un vuelo. Cada fecha, a su vez, es representada por una lista **`[año, mes, día]`**. A continuación se muestra una posible lista **`vuelos`**.

```
vuelos = [['AF10', [2014, 1, 2]], ['AF11', [2014, 1, 2]],
          ['AA12', [2014, 1, 3]], ['DE13', [2014, 5, 1]],
          ['DE14', [2014, 5, 1]], ['LA14', [2014, 7, 1]]]
```

- Por otra parte, cada vuelo puede tener varios destinos. Esta información se guarda en una lista llamada **`destinos`**, en la que cada entrada es una lista que contiene el número de vuelo y una lista de destinos. Cada destino, a su vez, se almacena en una lista **`[ciudad, país]`**. A continuación se muestra una posible lista **`destinos`**.

```
destinos = [
['AF10', [['Lima', 'Peru'], ['San Jose', 'Costa Rica'], ['Los Angeles','USA']]],
['AF11', [['San Jose', 'Costa Rica'], ['C. de Panama', 'Panama']]],
['AA12', [['Sao Paulo', 'Brasil'], ['San Jose', 'Costa Rica']]],
['DE13', [['Lima','Peru'], ['San Jose', 'Costa Rica']]],
['DE14', [['San Jose', 'Costa Rica'], ['Buenos Aires', 'Argentina']]],
['LA14', [['San Jose', 'Costa Rica']]]]
```

Ejercicio 1: midterm – 2014.2 – pregunta 4

En esta pregunta deberás escribir dos funciones.

a) **`vuelos_destino(vuelos, destinos, destino, fecha)`**. Esta función debe retornar una lista con los vuelos que salen hacia **`destino`** en la fecha **`fecha`**.

Por ejemplo,

`vuelos_destino(vuelos, destinos, ['San Jose', 'Costa Rica'], [2014, 5, 1])`

debe retornar **`['DE13', 'DE14']`**.

b) **`destinos_repetidos(destinos)`**. Esta función debe retornar una lista de listas con los destinos que figuran en todos los vuelos.

Por ejemplo,

`destinos_repetidos(destinos)` debe retornar **`[['San Jose', 'Costa Rica']]`**

2015 – segundo semestre – examen – P4

Problema 4 (60 puntos) --- 2015 2 - examen

Debes completar el programa que se muestra más abajo. Este programa modela la interacción entre los siguientes objetos: dos **cocineros**, un **mesón**, y dos **mozos** del restorán *Don Yadran*.

Los **cocineros** ponen platos en el **mesón** para que sean retirados por los **mozos**. El mesón tiene un espacio limitado y los **cocineros** no pueden poner más platos cuando el **mesón** está lleno. *Don Yadran* es muy famoso y se especializa en solo dos tipos de platos: "*lomitos*" y "*completos*". El **cocinero1** solo hace "*lomitos*" y el **cocinero2** sólo hace "*completos*".

Los **mozos** sacan platos del **mesón** (si es que los hay) y los ponen en una bandeja distribuidora que los llevan a los hambrientos clientes. El **mozo1** sólo saca del **mesón** platos "*lomitos*" y el **mozo2** sólo saca platos "*completos*".

El siguiente programa ejecuta muchas iteraciones de la interacción entre **cocineros**, **mesón** y **mozos**. En cada iteración, cada **cocinero** prepara y pone en el **mesón** una cantidad aleatoria de (entre cero y **P**) platos. En el programa, **P** vale **6** ("*lomitos*") para **cocinero1** y **8** ("*completos*") para **cocinero2**. Si un cocinero debe poner **x** platos en el **mesón**, pero en éste sólo caben **k**, con **k < x**, sólo pone **k** platos.

En cada iteración, cada **mozo** saca del **mesón** una cantidad aleatoria de (entre cero y **S**) platos. En el programa, **S** vale **5** ("*lomitos*") para **mozo1** y **4** ("*completos*") para **mozo2**. Si un mozo debe sacar **x** platos del **mesón**, pero en éste sólo quedan **k**, con **k < x**, sólo saca **k** platos.

Debes completar este programa escribiendo las clases **Meson**, **Cocinero**, y **Mozo**, asegurando que funcione a cabalidad. No puedes modificar el código presentado. Debes definir las clases, atributos y métodos necesarios. El método `lleno()` retorna `True` si el **mesón** está lleno y `False` en caso contrario. El atributo `faltan` indica la cantidad de platos que no estaban disponibles en el **mesón** para ser retirados por un **mozo** (no se acumulan de una iteración a otra). El atributo `tipo` indica si es "*lomito*" o "*completo*". El atributo `lom` indica la cantidad de "*lomitos*" en el **mesón**. El atributo `comp` indica la cantidad de "*completos*" en el **mesón**.

2015 – segundo semestre – examen – P4

```
### programa principal ###
meson=Meson(20)                              # crea un mesón de capacidad 20 platos
cocinero1 = Cocinero("lomito",6,meson)       # crea cocinero con máx. 6 lomitos
cocinero2 = Cocinero("completo",8,meson)     # crea cocinero con máx. 8 completos
mozo1=Mozo("lomito",5,meson)                 # crea mozo que retira máx. 5 lomitos
mozo2=Mozo("completo",4,meson)               # crea mozo que retira máx. 4 completos

t=0     # ejecuta 50 iteraciones
while t < 50:
    cocinero1.agregaPlatos()
    cocinero2.agregaPlatos()
    print("  Mesón lleno : ", meson.lleno())
    mozo1.retiraPlatos()
    print("  Faltan : ", mozo1.faltan," ",mozo1.tipo)
    mozo2.retiraPlatos()
    print("  Faltan : ", mozo2.faltan," ",mozo2.tipo)
    t = t + 1
    print ("t = "+str(t) + " Meson: " + str(meson.lom) + ", " + str(meson.comp))
# fin de la iteración
```