FINAL PARADIGMAS DE PROGRAMACION LEGAJO:

28/02/2009 NOMBRE:

Objetos

1) Mejor Efectividad

Existe el siguiente sistema de clubes de fútbol, donde cada club conoce algunos datos y estadísticas de los jugadores que forman parte de su plantel durante el presente campeonato.

```
Clase Club
vi: denominacion, jugadores

mejorEfectividad
^(jugadores asSortedCollection:
   [:a :b | a efectividad > b
efectividad]) first

Clase Jugador
vi: nombre, fechaNacimiento,
golesConvertidos, partidosJugados

efectividad
^golesConvertidos/partidosJugados
```

Como el sistema anduvo bien en un campeonato se lo quiere seguir utilizando en los siguientes, pero manteniendo el registro histórico de la información, es decir las estadísticas de cada jugador en cada campeonato. Tener en cuenta que entre un campeonato y otro los jugadores pueden cambiar de club o retirarse, o un nuevo jugador puede incorporarse al club. Asumir que hay un solo campeonato por año.

Por ejemplo:

- *Palacios, que nació el 20/02/86, jugó en Boca en el campeonato del 2008. Hizo 8 goles y jugó 16 partidos.*
- *Palermo, que nació el 1/05/80, jugó en Boc, en el campeonato del 2008, hizo 1 gol y jugó 3 partidos. En Estudiantes, en el campeonato del 1999, hizo 18 goles y jugó 18 partidos.*

Ahora se quiere conocer el jugador de mayor efectividad del club en el campeonato de un año dado: por ejemplo, si se le pregunta a Boca quién tuvo mayor efectividad en el 2008, la respuesta es Palacios.

Hacer:

a) Identificar los problemas principales que presenta la solución actual teniendo en cuenta los nuevos requerimientos.
b) Realizar los cambios necesarios para adaptar el sistema.
c) Dibujar un diagrama de objetos con los datos del ejemplo.

2) Resultados de los partidos

En el mismo sistema (sin las modificaciones hechas en el punto anterior) también se registra el resultado de los partidos:

Por ejemplo:
Los partidos de Boca

- *Contra River, salió 2 a 1.*
- *Contra Banfield, 1 a 1.*

Para calcular los puntos que obtuvo el equipo en el partido, está el siguiente método:

```
Clase Partido
puntos
 (self golesPropios > self
golesAdversario ) ifTrue: [^3].
 (self golesPropios = self
golesAdversario) ifTrue: [^1].
 ^0
```

El club tiene el siguiente método

```
totalPuntos
^partidos inject: 0 into: [:p :tot
| tot + p puntos]
```

Si bien muchos partidos usan este esquema, hay otros partidos en los que se registra la información con más detalle. Es posible encontrar partidos registrados de la primera forma y otros de la segunda.

Por ejemplo:

Siguiendo con Boca, en el partido contra River
Para boca, Palermo hizo un gol en el minuto 23 y Palacios otro en el minuto 79.
Para el adversario, Buenanote hizo un gol en el minuto 40.

Hacer:

a) Realizar los cambios necesarios para registrar esta información y seguir calculando el total de puntos que hizo un club.
b) Justificar los conceptos aplicados.
c) ¿Fue necesario/conveniente la utilización de self y/o super? Explicar.

3) Sistema integrado

¿Cómo integraría la nueva solución del punto 1 con los cambios que realizó en el punto 2? No es necesario codificar, sino explicar qué conceptos entran en juego

Lógico

En la base de conocimiento está la siguiente información sobre los préstamos y pagos de clientes a sus bancos. También se sabe qué bancos son extranjeros.

```
prestamo(nacion, juan, 1000).
prestamo(ciudad, ana, 2000).
prestamo(bnpParis, carlos, 1000).
pago(juan, nacion, 300).
pago(ana, ciudad, 100).
pago(carlos, bnpParis, 400).
extranjero(bnpParis).
```

Se necesita obtener los clientes morosos. Un banco extranjero considera morosos a los que pagaron menos de la mitad del préstamo, mientras que los restantes bancos sólo a los que pagaron menos de la cuarta parte (se asume que todos pagaron algo).

Para ello, se construyó la siguiente solución:

```
moroso(Cli):-
      prestamo(Bco, Cli,
Prestamo),
      extranjero(Bco),
      pago(Cli, Bco, Pago),
      Pago < Prestamo / 4.
moroso(Cli):-
      prestamo(Bco, Cli,
Prestamo),
      pago(Cli, Bco, Pago),
      Pago < Prestamo / 2.
```

1) Análisis de la solución

¿La solución hace lo que tiene que hacer?

- En caso positivo, si hay alguna mejora que se le pueda hacer, hacerla.
- En caso negativo, corregirla para que funcione.

2) Nuevos requerimientos

Ahora se quiere contemplar a los clientes que sacaron préstamos y nunca pagaron.
Una forma sería introducir para estos casos hechos del predicado pago/3, con importe 0.

a) ¿De qué otra forma se puede resolver correctamente?
b) Comparar ambas soluciones e indicar ventajas y desventajas.

Funcional

1) Local de ventas

Para un sistema de cálculo de premios en un local de venta de jeans, se define lo siguiente:

```
empleados = [
("marta", [600, 200, 300]),
("cristian", [1000, 50, 700,
250]), ("daniela", [250, 250,
250])
]
```

Es una lista de duplas, donde la primera componente es el nombre del empleado y la segunda una lista de ventas totales por día.

```
cobraBonoVentas [] = []
cobraBonoVentas (x:xs)
   |sum (snd x ) >= 2000
      = x: cobraBonoVentas xs
   |otherwise
      = cobraBonoVentas xs

cobraBonoRegularidad [] = []
cobraBonoRegularidad (x:xs)
   |all (>= 250) (snd x)
      = x : cobraBonoRegularidad
xs
   |otherwise
      = cobraBonoRegularidad xs
```

a) Reescribir las definiciones utilizando funciones de orden superior.
b) Detallar qué ventajas presenta una solución donde se utilizan y reutilizan funciones de orden superior versus a una que no
(por ejemplo, qué ocurre con la expresividad del software escrito?)