Ingeniería de Software 1

Práctica 6 – Testing funcional

Parte 1 – Casos de test a partir de una descripción textual o informal

Ejercicio 1

Diseñe el test de unidad de una funcionalidad que, dados tres enteros que se interpretan como la longitud de cada uno de los lados de un triángulo, dice si el triángulo resultante es isósceles, escaleno o equilátero.

Ejercicio 2

Se quiere testear una máquina expendedora de boletos a ser instalada en la línea 36 de colectivos. El colectivo cuenta con un sensor de tipo GPS que le indica a la máquina expendedora en qué tramo del recorrido se encuentra como un entero entre 0 y 99 que debe ser interpretado como el porcentaje completado del recorrido. La expendedora le solicita al pasajero que indique su destino final y en función del mismo (y la posición actual del colectivo) calcula la tarifa.

La tarifa es de 50 centavos si el pasaje corresponde a un recorrido menor o igual al 10 % del recorrido total; $1 si el tramo recorrido está entre el 11 % y el 50 %; $1,50 si es mayor al 51 %.

Ejercicio 3

Se tiene una clase `PilaProductos` que implementa una pila de objetos `Producto` y tiene los siguientes métodos:

- `iniciar(capacidad: integer)`
- `apilar(p: Producto)`
- `desapilar(): Producto`
- `estaIniciada(): boolean`
- `capacidad(): integer`
- `estaVacia(): boolean`
- `estaLlena(): boolean`

No contamos con una especificación, pero sabemos que se trata de una pila acotada, donde inicialmente hay que indicar la cantidad máxima de elementos permitidos dentro de la pila. Mientras no se haya iniciado, el únicos método que se puede invocar es `estaIniciada()`.

Se pide diseñar el test del método `apilar()`.

Ejercicio 4

Un sistema de pagos está diseñado de manera de tener un registro de cada empleado. Cada dos semanas, se imprimen reportes con el número de horas que trabajó cada empleado en lo que va del año. Para cada empleado se establece su honorario por hora. Cada fin de mes, se imprime el cheque de cada empleado.

Diseñe el test funcional para el problema descripto, detallando los casos de test y un conjunto de datos de test correspondiente a cada caso.

Ejercicio 5

Se desea testear una funcionalidad destinada a la concesión de un permiso para establecer un puesto comercial (de un cierto tipo) en una estación de una línea subterránea por parte de un Ente Municipal a un individuo determinado o a algún tipo de sociedad (en ambos casos reconocidos con personería jurídica).

El proceso de la concesión de dicho permiso debe ser llevado a cabo por un usuario logueado al sistema del municipio, con perfil del grupo de registraciones. Este usuario ingresa el DNI o CUIT del titular de la nueva concesión. Si el DNI o CUIT corresponde a un titular no registrado, la concesión no se efectúa. El titular puede ser una persona jurídica individual o una sociedad. En el caso de ser una persona jurídica individual, no puede tener más de dos concesiones simultáneas.

Toda concesión se hace efectiva en una estación de una línea subterránea. Las estaciones son conocidas por su nombre, y no existen estaciones de distintas líneas subterráneas con el mismo nombre. Las estaciones que pueden tener este tipo de puestos son solamente las estaciones bidireccionales (que poseen andenes de trenes que se dirigen en ambas direcciones).

Los tipos de puestos que se pueden concesionar son kioscos y bancos. Sólo puede existir a lo sumo un banco concesionado entre todas las estaciones pertenecientes a una misma línea subterránea.

En el caso de que la concesión de un banco ocurra satisfactoriamente, el sistema emite un reporte con los datos de la concesión al grupo de *marketing* con el objeto de hacer publicidad.

Se solicita lo siguiente:

a) Indicar los factores, categorías y elecciones que intervienen en el diseño de los casos de test de la unidad funcional.
b) En base al punto anterior, diseñar los casos de test en base al método de partición por categorías indicando el resultado esperado para cada caso.

Ejercicio 6

La empresa de transporte de pasajeros TerremotoTerrestre ofrece servicios de media y corta distancia. Se considera larga distancia cuando el punto de partida y el de destino están separados más de 100 km. Los micros pueden tener uno o dos pisos, usándose estos últimos únicamente para el transporte de larga distancia.

La funcionalidad a testear es la de la logística de asignación de micros a recorridos. Más precisamente, dado un conjunto de recorridos (especificado como una *lista* de triplas ⟨origen, destino, distancia⟩) y un conjunto de micros (especificado como una *lista* de tuplas ⟨patente, cantidad de pisos⟩), se debe asignar un micro a cada uno de los recorridos (la función debe devolver una lista de tuplas ⟨patente, origen, destino⟩).

Proponga una partición de categorías y un conjunto de tests para validar la implementación de la funcionalidad pedida. El conjunto de tests debe ser *adecuado*, es decir que debe contemplar todos los casos interesantes de ser testeados.

Ejercicio 7

La entidad bancaria NoHayUnMango ha lanzado una nueva línea de créditos para la vivienda (hipotecarios) con el objetivo de que cada vez más personas puedan acceder a su propia casa. Para esto desarrollaron un programa `EvaluaciónPréstamo` que, en función de algunas características del solicitante y del dinero necesario para la compra, determina si el crédito es otorgado o no y, en caso afirmativo, informa el valor de la cuota inicial.

Esta nueva línea otorga créditos a 15 ó 20 años de plazo de devolución.

El monto del préstamo solicitado debe ser superior a 20.000 pesos y, para que finalmente el crédito sea otorgado, el solicitante deberá demostrar ingresos netos mayores a 5.000 pesos. Además, estos ingresos deberán ser superiores al 10 % del valor del préstamo.

Para incentivar la solicitud de créditos, el Banco lanzó una promoción en la que para los créditos a 20 años la cuota sufre un 15 % de descuento.

Proponga una partición de categorías y un conjunto de tests para validar la implementación de la funcionalidad pedida. El conjunto de tests debe ser *adecuado*, es decir que debe contemplar todos los casos interesantes de ser testeados.

Ejercicio 8 *(Testing funcional)*

Un conocido club de fútbol desea implementar la función `calcularPrecio`. La misma se utiliza para computar el precio final que los socios del club deben pagar por las entradas para cada partido.

A continuación se detallan los aspectos que debe tener en cuenta la función para realizar descuentos sobre el precio base de la entrada:

- Las entradas para la popular cuando se juegan de local son gratis (el precio es 0).
- Si un jubilado (edad mayor a 60 años) quiere comprar una entrada para platea y se juega de local, el club hace un descuento del 40 % sobre el precio base.
- Si un menor (edad menor a 12 años) quiere comprar una entrada para platea y se juega de local, el club hace un descuento del 25 % sobre el precio base.
- Finalmente, cuando se juegue de visitante el precio de la entrada para la popular es un 20 % menos que el precio base y la platea no sufre descuentos.

Realice el diseño de los casos de test de la función `calcularPrecio` antes descripta utilizando el método de partición por categorías.

Parte 2 – Casos de test a partir de especificaciones

Ejercicio 9

Diseñe el test funcional correspondiente al ejercicio sobre venta de electrodomésticos de la práctica de casos de uso. Indique:

- Casos de test.
- Datos de test, incluyendo datos de input y datos del entorno necesarios para la futura ejecución del test.
- Resultado esperado.

Ejercicio 10

a) Dado del enunciado y su resolución del ejercicio "Máquina para la venta y expedición de bebidas en forma automática" de la práctica de diagramas de actividad, derive casos de testing funcional utilizando la técnica de partición de categorías.
b) Explique los pros y los contras de hacer testing sobre cada actividad del diagrama y hacerlo sobre el diagrama en general. ¿En qué casos aplicaría qué técnica? ¿Aplicaría las dos juntas? Justifique.

Ejercicio 11

Derive casos de test funcional utilizando el diagrama de su resolución del ejercicio del teléfono celular de la práctica de FSM.

Ejercicio 12

Un microemprendimiento dedicado a la elaboración y venta de cervezas cuenta con un software dedicado a la gestión comercial. Este software brinda una funcionalidad de facturación rápida de cervezas, donde se generan facturas de un único ítem. El software respeta la siguiente especificación:

Casos de uso:

Caso de uso: Registrando factura rápida.

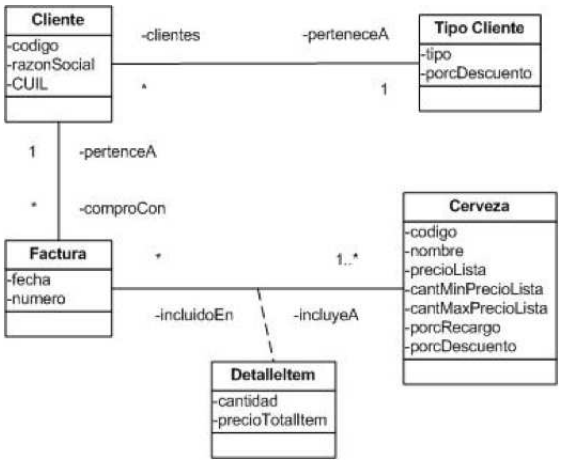
Actor: Facturador.

Precondición: Usuario logueado.

Postcondición: Queda registrada la nueva factura con sus valores correctos.

Curso normal	Curso alternativo
1. El usuario ingresa el código de cliente, el código de una cerveza y la cantidad a facturar.	
2. El sistema valida los datos.	**2.1.** Datos inválidos. Si algún dato es inválido, muestra un mensaje explicativo. Fin del caso de uso.
3. El sistema registra la factura correspondiente a los datos ingresados, previo cálculo del monto correspondiente (ver DA "Registrar factura rápida").	
4. El sistema muestra un mensaje indicando el éxito de la operación. Fin del caso de uso.	

Modelo conceptual / OCL:

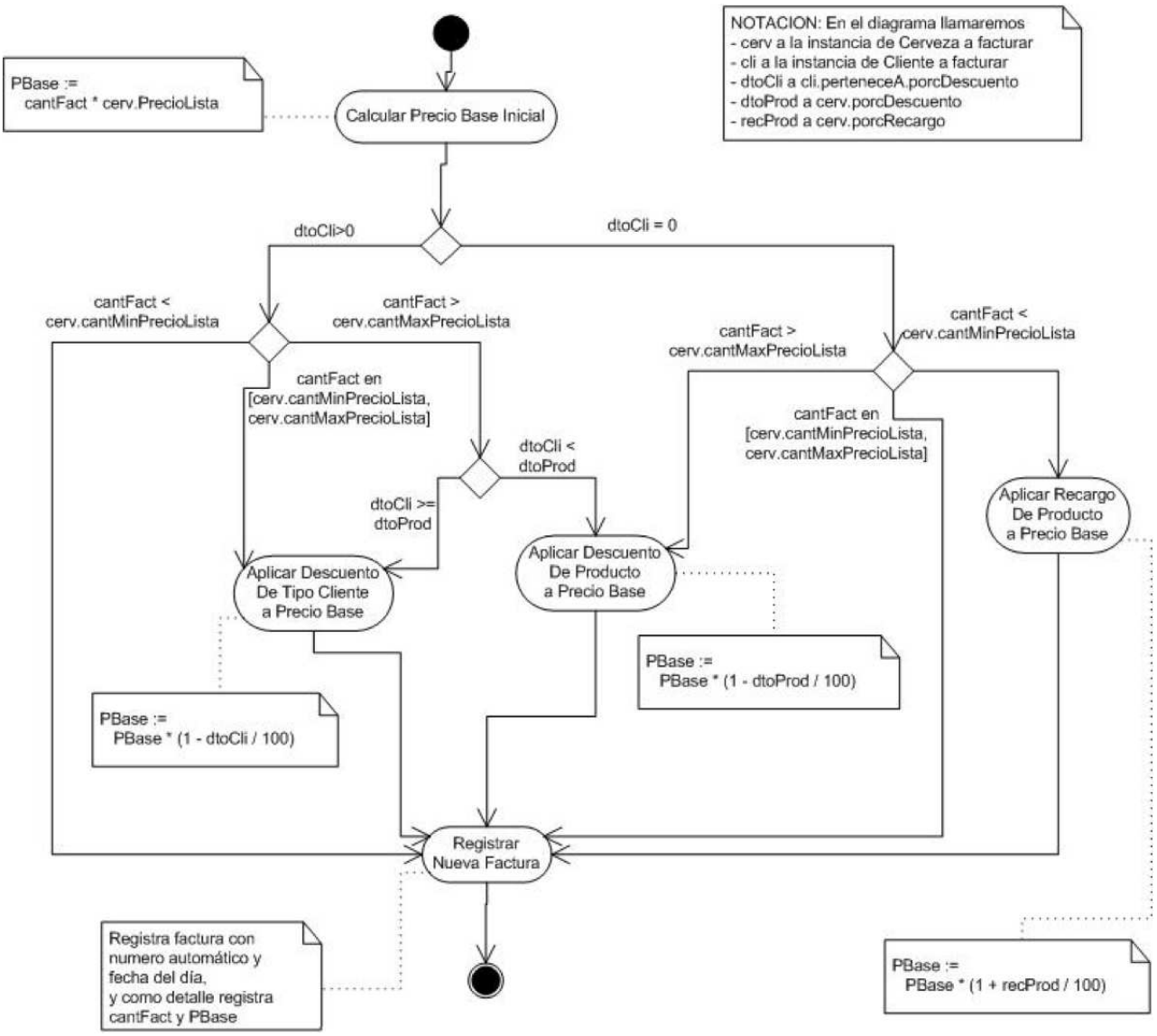



```
context Tipo Cliente
  inv: self.porcDescuento >= 0 and self.porcDescuento <= 100

context Factura
  inv: Factura->allInstances()->forAll(x, y | x <> y implies x.numero <> y.numero)

context DetalleItem
  inv: self.cantidad > 0

context Cerveza
  inv: self.cantMinPrecioLista> 0
  inv: self.cantMaxPrecioLista > self.cantMinPrecioLista
  inv: self.porcRecargo > 0 and self.porcRecargo <= 100
  inv: self.porcDescuento > 0 and self.porcDescuento <=100
```

Diagramas de actividad:

DA: Registrar factura rápida.
Precondición:

- Código de cliente (`codCli`) válido.
- Código de cerveza (`codCerv`) válido.
- Cantidad a facturar (`cantFact`) válida.



Se pide diseñar los casos de prueba para la funcionalidad mencionada utilizando el método de partición de categorías, indicando los factores, categorías, elecciones y las eventuales restricciones de estas últimas.