Unidad

4

DIPLOMATURA EN PROGRAMACION JAVA

Ejercicios

Capítulo 7

Genéricos y Colecciones

Capítulo 7

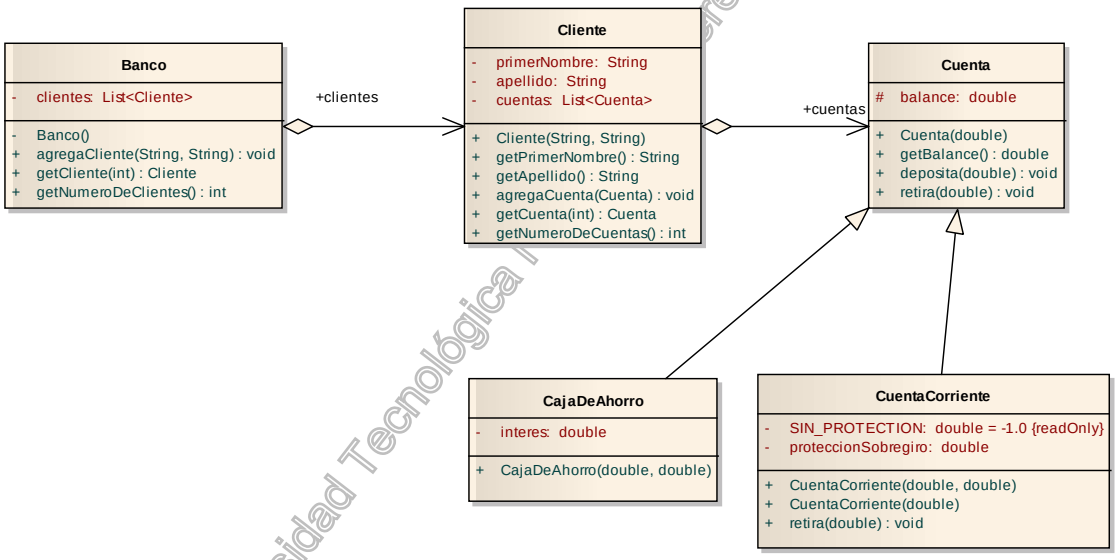
Ejercicio 1

En este ejercicio va a usar colecciones genéricas para representar asociaciones de clases en el modelo de dominio del proyecto Bancos.

En el diseño anterior se usaron vectores para implementar la multiplicidad en las relaciones entre el banco sus clientes, y entre los clientes y sus cuentas. Este diseño tiene varias limitaciones importantes, la principal es que el vector, una vez creado, tiene un tamaño fijo.

La API Collections se creó para solucionar ésta y otras limitaciones.

La siguiente figura muestra el modelo de dominio del proyecto Bancos con las asociaciones de clases: un banco presta servicio a muchos clientes y un cliente tiene muchas cuentas. La figura también muestra el diseño detallado de las clases Banco y Cliente que utilizan una lista genérica para mantener dichos vínculos.



Abrir el espacio de trabajo

1. Abrir el proyecto del directorio Capítulo 7\Ejercicio 1.

Modificación de la clase Banco

2. Eliminar las declaraciones en la clase que la convierten en un patrón de instancia única;

3. Modificar la declaración de la variable de instancia clientes para que sea estática y del tipo List<Clientes>. Borrar la variable de instancia numeroDeClientes.

4. Elimine las constantes que tenga la clase y que no utilice.

5. Crear el siguiente bloque estático para inicializar la variable de instancia clientes de manera que sea un nuevo objeto del tipo ArrayList.

```
static {
	clientes = new ArrayList<Cliente>(10);
}
```

6. Elimine los elementos declarados dentro del constructor por defecto. Ya no son necesarios porque se crea la colección en el bloque estático

7. Modificar el método agregaCliente para que utilice el método add y sea estático.

8. Modificar el método getCliente para que utilice el método get y sea estático.

9. Modificar el método getNumeroDeClientes para que utilice el método size y sea estático.

10. Verificar que la clase ReporteCliente no tenga ningún error de compilación. Tome nota de los cambios en la invocación de los métodos debido a que los mismos fueron declarados estáticos

11. Verificar que la clase PruebaOperacionesBancarias no tenga ningún error de compilación. Tome nota de los cambios en la invocación de los métodos debido a que los mismos fueron declarados estáticos

Modificación de la clase Cliente

12. Modificar la declaración de la variable de instancia cuentas para que sea del tipo List<Cuenta>. Borrar la variable de instancia numeroDeCuentas.

13. Modificar el constructor para inicializar la variable de instancia cuentas de manera que sea un nuevo objeto del tipo ArrayList.

14. Modificar el método agregaCuenta para que utilice el método add.

15. Modificar el método getCuenta para que utilice el método get.

16. Modificar el método getNumeroDeCuentas para que utilice el método size.

Verificación de la clase VerificaReporteDeClientes

17. Corroborar que la clase VerificaReporteDeClientes no tiene ningún error de compilación

Ejecución del programa

18. Ejecutar el programa desde la clase VerificaReporteDeClientes. La salida debe ser similar a la siguiente:

```
                REPORTE DE CLIENTES
                ===================

Cliente: Pérez, Juan
    Caja de Ahorro: el balance actual es $500,00
    Cuenta Corriente: el balance actual es $200,00

Cliente: García, Pedro
    Cuenta Corriente: el balance actual es $500,00
    Caja de Ahorro: el balance actual es $380,00

Cliente: Toma, Oscar
    Cuenta Corriente: el balance actual es $500,00
    Caja de Ahorro: el balance actual es $700,00

Cliente: Soley, Maria
    Caja de Ahorro: el balance actual es $700,00
    Cuenta Corriente: el balance actual es $150,00
```


Ejercicio 2

En este ejercicio se ordenará la lista de los clientes del banco por sus nombres, lo cual requiere que la clase Cliente implemente la interfaz Comparable
Comenzar creando un directorio llamado Ejercicio 2. Abrir el Eclipse sobre este directorio para que cree un nuevo espacio de trabajo. Importar proyecto del Ejercicio 1 realizando los siguientes pasos:

a) Pararse en el exploradorde paquetes
b) Presionar el botón derecho del mouse y seleccionar la opción Import del menú desplegable.
c) En el cuadro de diálogo, seleccionar “Existing Projects into Workspace” y presionar el botón Next.
d) Presionar el botón Browse y navegar hasta el directorio que contiene el Ejercicio 1 del Capítulo 7.
e) Seleccionar Ok y en la ventana Projects debería mostrarse el proyecto seleccionado.
f) Asegurarse que el check box “Copy projects into workspace” este seleccionado para que copie todo el proyecto en el nuevo espacio de trabajo creado
g) Presionar Finish

Realizar las siguientes tareas en el nuevo proyecto importado:

Clase Banco

1. Agregar el método ordenarClientes

Clase Cliente

2. Modificar la clase Clientes para implementar la interfaz Comparable. Se necesitará especificar el método compareTo. Hacer que este método compare dos clientes en orden lexicográfico tomando como precedencia el apellido antes del nombre

Clase VerificaReporteDeClientes

3. Modificar la clase VerificaReporteDeClientes para llamar al método ordenarClientes antes de generar el reporte al final del método estático inicializarClientes.
4. Compilar y ejecutar el programa VerificaReporteDeClientes y verificar que se obtenga la siguiente salida:

```
                REPORTE DE CLIENTES
                ===================

Cliente: García, Pedro
    Cuenta Corriente: el balance actual es $500,00
    Caja de Ahorro: el balance actual es $380,00

Cliente: Pérez, Juan
    Caja de Ahorro: el balance actual es $500,00
    Cuenta Corriente: el balance actual es $200,00

Cliente: Soley, Maria
    Caja de Ahorro: el balance actual es $700,00
    Cuenta Corriente: el balance actual es $150,00

Cliente: Toma, Oscar
    Cuenta Corriente: el balance actual es $500,00
    Caja de Ahorro: el balance actual es $700,00
```


Ejercicio 3

Tomando como punto de partida el ejercicio anterior (crear el espacio de trabajo e importar el proyecto del modo descrito en el Ejercicio 2), modificar la interfaz implementada en la clase Cliente para que admita genéricos (usar Comparable<T> en lugar de Comparable). Tener en cuenta que para implementar el parámetro genérico de la interfaz debe utilizarse el tipo Cliente, que es el tipo de objeto que se desea comparar.

También modificar el ArrayList para que sea genérico y utilice un tipo Cuenta.

Verificar que la salida queda inalterable y sólo mejoró la calidad del código porque se aseguró el tipo.