



INSTITUTO POLITÉCNICO NACIONAL

Escuela Superior de Cómputo



Unidad de Aprendizaje:

Programación Orientada a Objetos

Grupo:

2CV8

Profesor:

Daniel Cruz García

Práctica 4

Alumnos:

Luciano Espina Melisa

Dávila García Rivas Emiliano

Ramos Mesas Edgar Alaín

Fecha de entrega:

08/04/2018

Introducción

En la programación orientada a objetos se ven diversos temas, los cuales se deben completar para poder introducirse de lleno a la unidad; por lo tanto para la implementación en las aplicaciones que se pidieron en esta práctica las cuales fueron: implementación de clases, sobrecarga de métodos, relaciones entre clases, instanciación, manejo de arrays, uso de jar.

Marco teórico

Arraylist

La clase ArrayList en Java, es una clase que permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el número de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays. Para todos aquellos que hayáis estudiado en alguna asignatura las estructuras de datos de las Pilas, Colas, Listas, Arboles (AVL, B, B+, B*) etc. hay decir que los ArrayList "tiran por tierra" toda la teoría que hay detrás de esas estructuras de datos ya que los ArrayList nos permiten añadir, eliminar y modificar elementos (que pueden ser objetos o elementos atómicos) de forma transparente para el programador. [1]

```
// Declaración de un ArrayList de "String". Puede ser de cualquier otro Elemento u Objeto (float, Boolean, Object, ...)
ArrayList<String> nombreArrayList = new ArrayList<String>();
// Añade el elemento al ArrayList
nombreArrayList.add("Elemento");
// Añade el elemento al ArrayList en la posición 'n'
nombreArrayList.add(n, "Elemento 2");
// Devuelve el número de elementos del ArrayList
nombreArrayList.size();
// Devuelve el elemento que está en la posición '2' del ArrayList
nombreArrayList.get(2);
// Comprueba si existe el elemento ('Elemento') que se le pasa como parámetro
nombreArrayList.contains("Elemento");
// Devuelve la posición de la primera ocurrencia ('Elemento') en el ArrayList
nombreArrayList.indexOf("Elemento");
// Devuelve la posición de la última ocurrencia ('Elemento') en el ArrayList
nombreArrayList.lastIndexOf("Elemento");
// Borra el elemento de la posición '5' del ArrayList
nombreArrayList.remove(5);
// Borra la primera ocurrencia del 'Elemento' que se le pasa como parámetro.
nombreArrayList.remove("Elemento");
// Borra todos los elementos de ArrayList
nombreArrayList.clear();
// Devuelve True si el ArrayList está vacío. Sino Devuelve False
nombreArrayList.isEmpty();
// Copiar un ArrayList
ArrayList arrayListCopia = (ArrayList) nombreArrayList.clone();
```

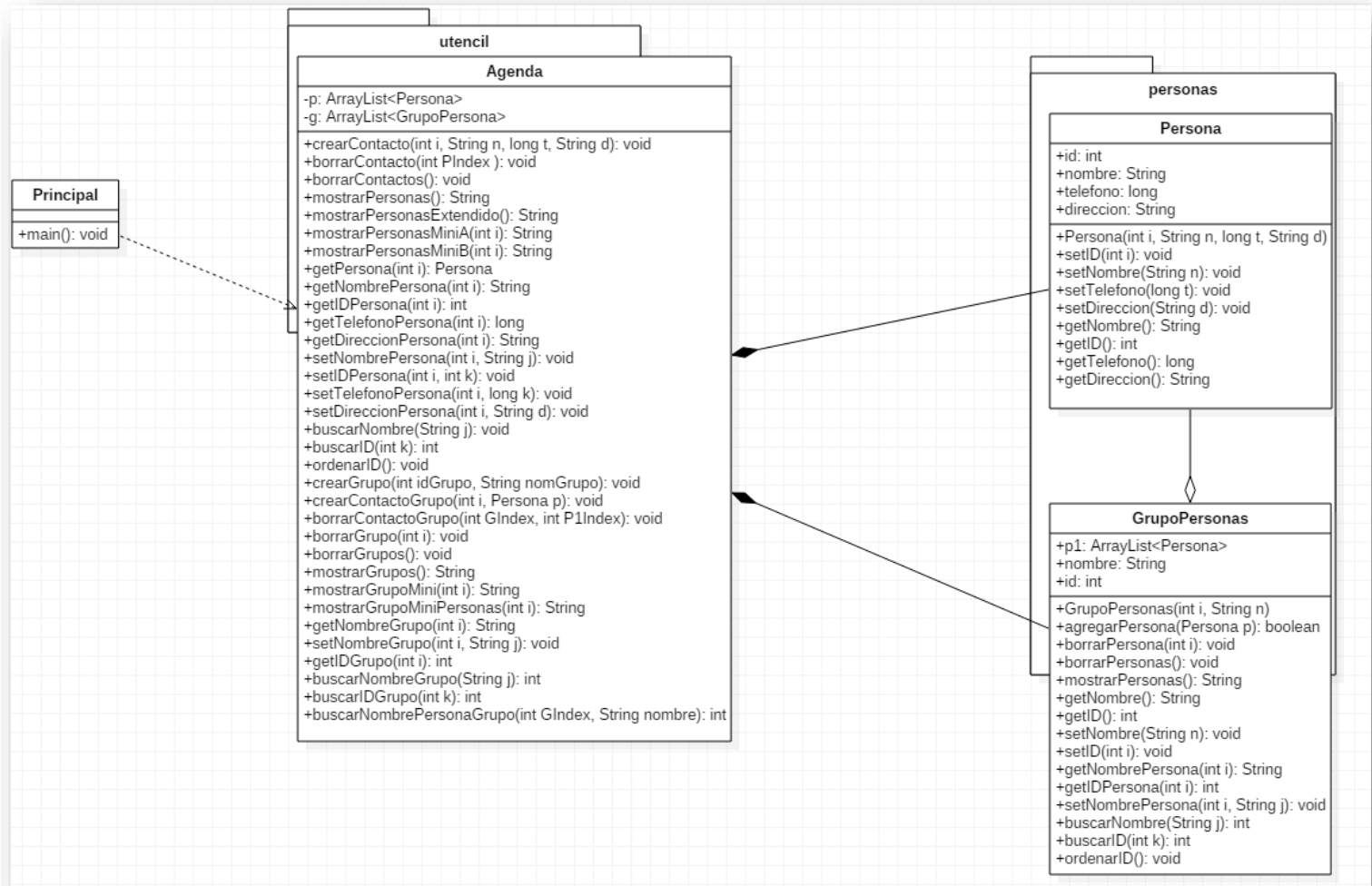
Método get y set

Los métodos getters y setters, o métodos de "acceso", proporcionan acceso a las propiedades de un objeto. Un método get devuelve el valor de una propiedad de un objeto. Un método get tiene un tipo de retorno que se relaciona con el tipo de variable miembro asociada. Los métodos get generalmente no toma ningún parámetro. Un método set tiene un tipo de retorno "void" y toma un parámetro del tipo adecuado para asignar a la variable miembro asociada.

Los métodos de acceso son utilizados por los objetos externos y por lo tanto se declaran como métodos "públicos" (externamente visible). La convención aceptada es la de nombrar métodos get y set para la variable de miembro asociada (por ejemplo, "getName" y "setName", asociado a la variable "nombre"), con el prefijo "get" o "set". No todas las variables miembros pueden tener métodos de acceso asociados. Estos métodos se escriben solamente para los valores que se deben acceder externamente. [2]

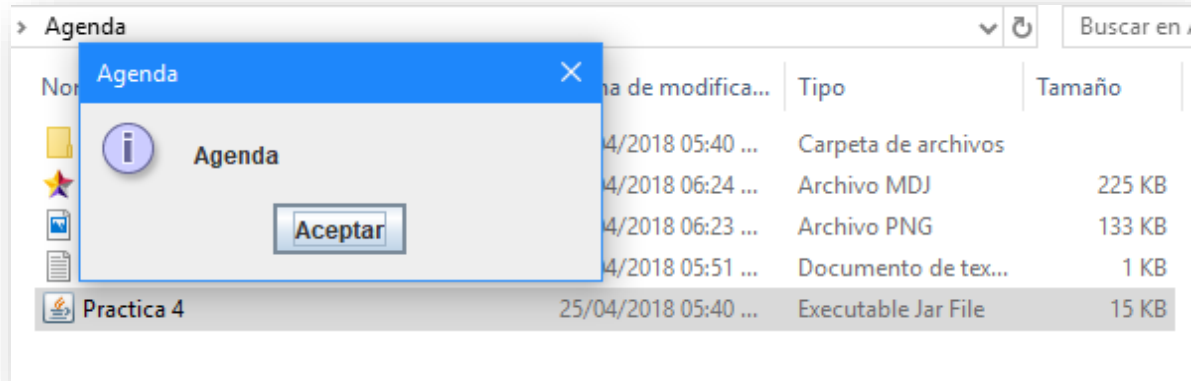
Análisis del problema

Análisis del problema I



Pruebas y resultados I

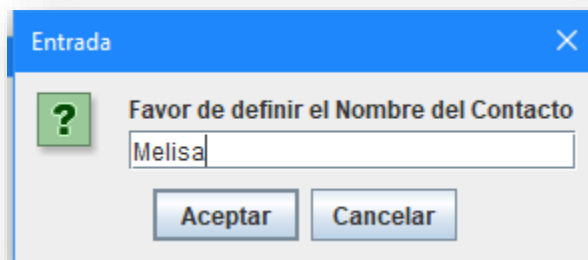
1.- Sólo dando doble click al .jar aparece la ventana de la aplicación



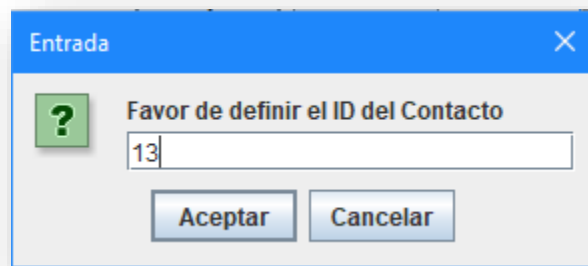
2.- Al dar click en aceptar aparece la segunda ventana



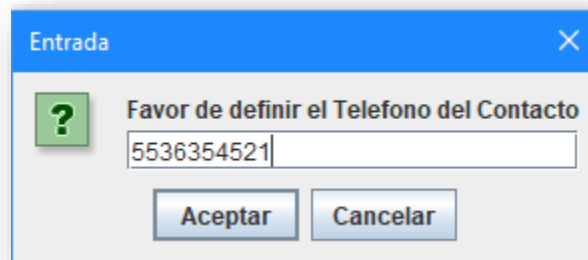
3.- Al dar click en "Crear contacto" aparece la segunda ventana ingresando el nombre del contacto nuevo y luego aceptar



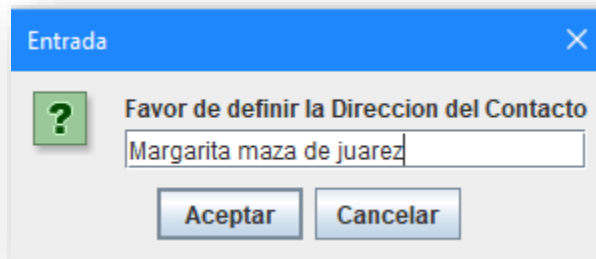
4.- Aparece otra ventana la cual te pide el ID del contacto, ingresamos un ID



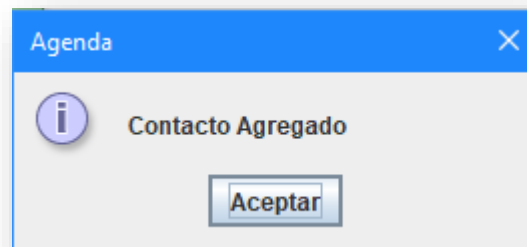
5.- Aparece nuevamente una ventana pero ahora pidiendo el numero del usuario



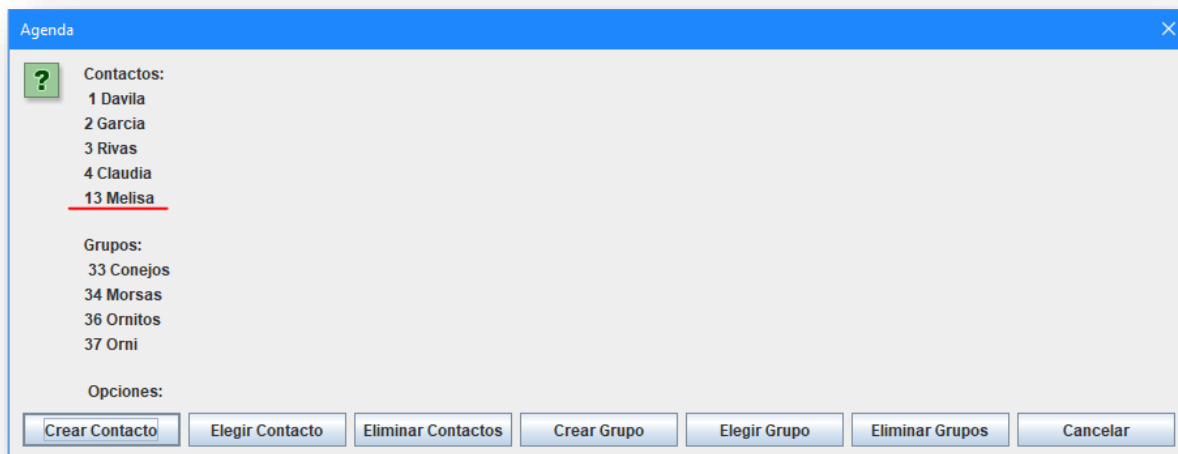
6.- Después aparece otra ventana pidiendo la dirección del usuario



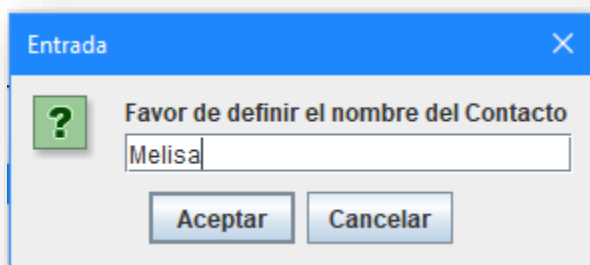
7.- Cuando se agrega aparece esta ventana



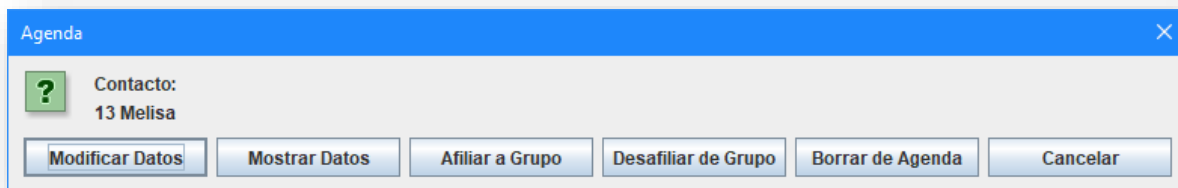
8.- Aparece nuevamente el menú con el nuevo usuario en este caso “Melisa”



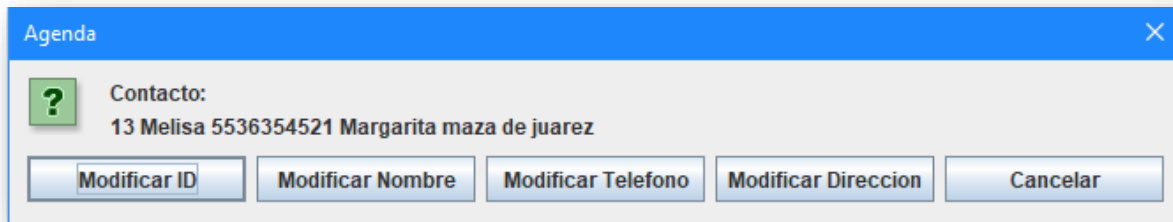
8.- Ahora se selecciona “Elegir contacto” e ingresamos el nombre del contacto que se acaba de crear



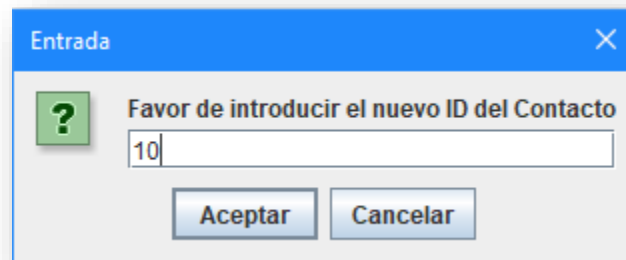
9.- Aparece una nueva ventana, donde seleccionaremos “Modificar Datos”



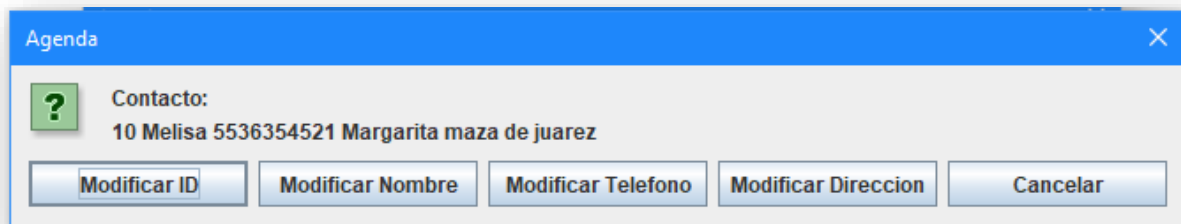
10.- Aparece una nueva ventana donde seleccionaremos “Modificar ID”



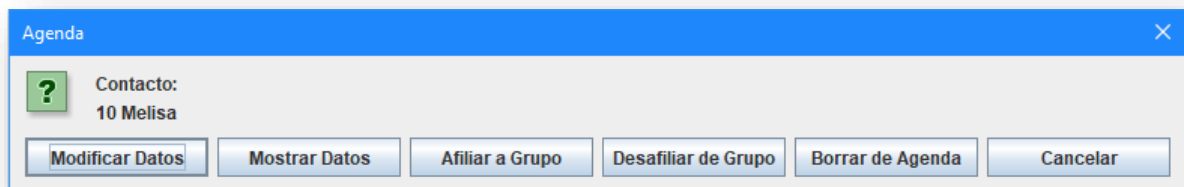
11.- Se ingresa otro número para su ID

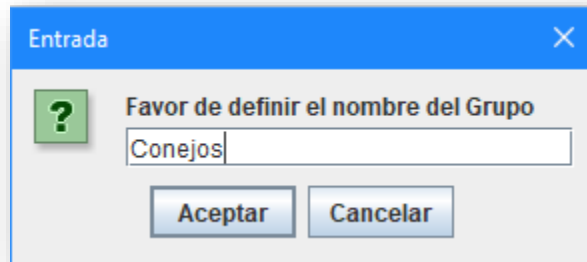


12.- Ahora aparece el contacto editado y se selecciona cancelar, se puede hacer lo mismo con los otros datos

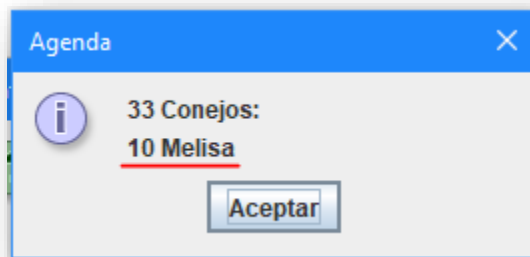
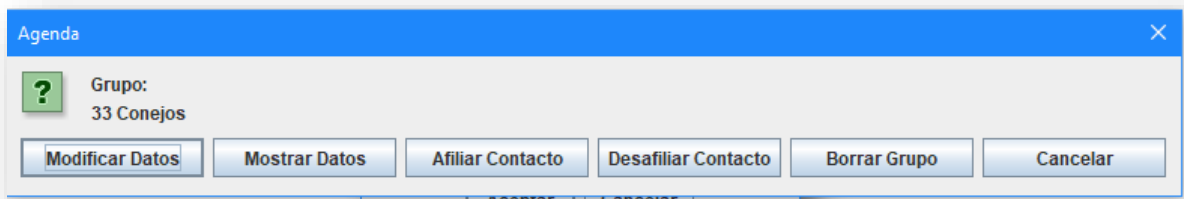


13.- Se puede afiliar a un grupo en este caso lo haremos al de conejos

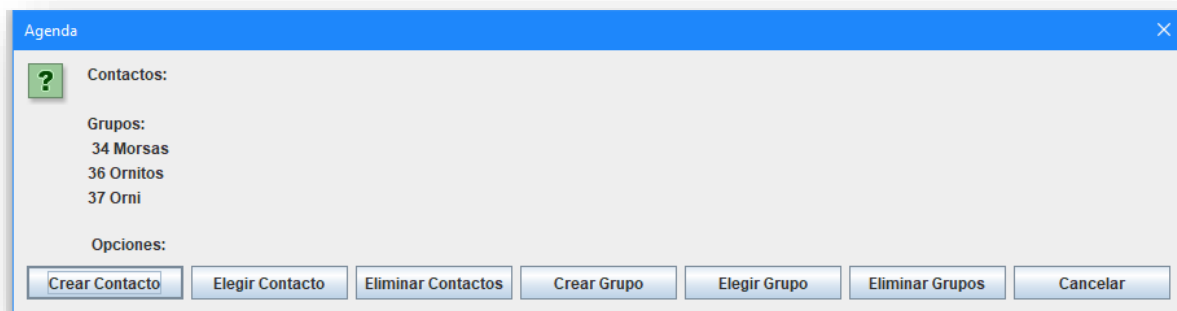
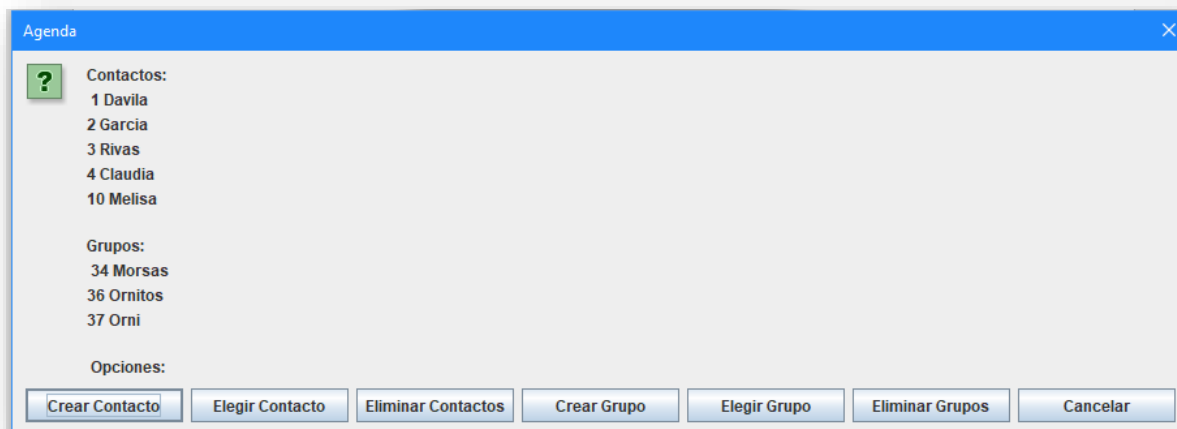




14.- Una vez afiliada se puede observar en ese grupo

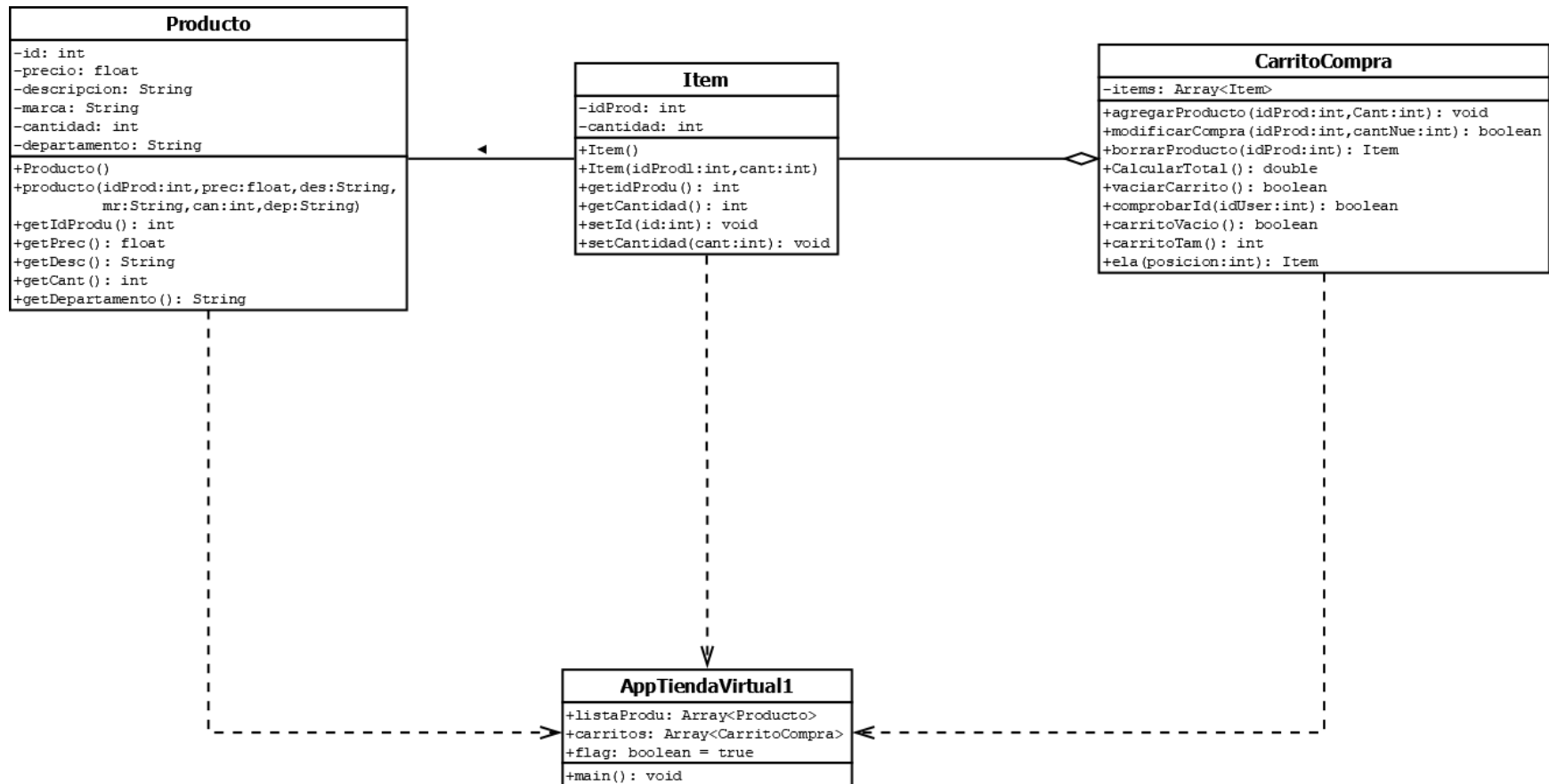


15.- Se puede borrar grupos y aquí eliminamos al de los conejos y ya no aparece o se puede eliminar a todos los contactos

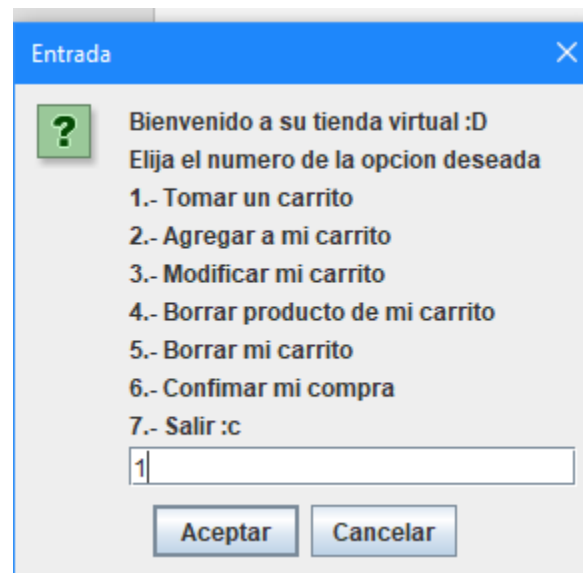




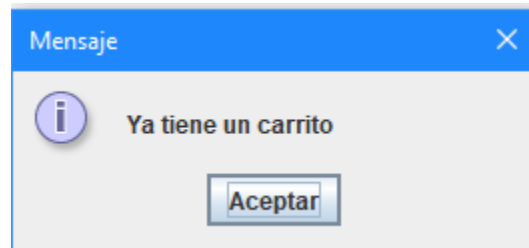
Análisis del problema II



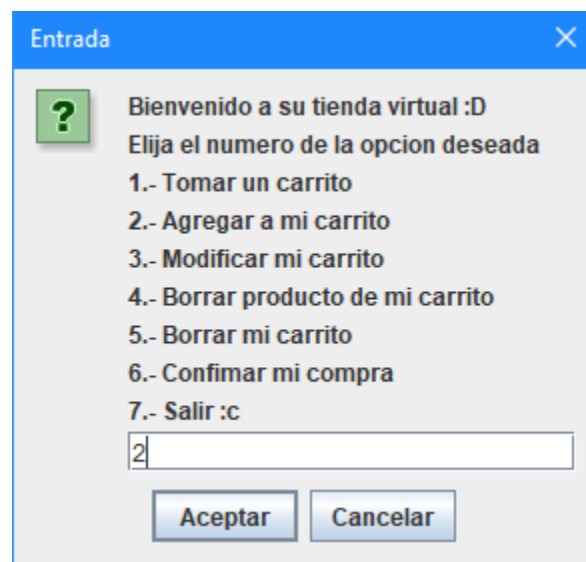
1.- Se observa la ventana principal, seleccionando la opción uno



2.- Aparece la ventana y se da aceptar




3.- Los ID están numerados del 1 al 10



4.- Seleccionando la opción 2 aparece


Entrada

 Ingresa el ID del producto que desea agregar al carrito

1

Aceptar Cancelar


Entrada

 Ingresa la cantidad de productos que quieres

12

Aceptar Cancelar


Mensaje

 Se agrego el producto :D

Aceptar

5.- Ahora seleccionando la opción 3

Entrada


 Bienvenido a su tienda virtual :D
Elija el numero de la opcion deseada

1.- Tomar un carrito
2.- Agregar a mi carrito
3.- Modificar mi carrito
4.- Borrar producto de mi carrito
5.- Borrar mi carrito
6.- Confirmar mi compra
7.- Salir :c

3

Aceptar Cancelar


Entrada

 Ingresa el ID del producto que desea modificar del carrito

1

Aceptar Cancelar

Entrada


 Ingresa la nueva cantidad

10

Aceptar Cancelar

6.- Confirmando la compra

Entrada


 Bienvenido a su tienda virtual :D
Elija el numero de la opcion deseada

- 1.- Tomar un carrito
- 2.- Agregar a mi carrito
- 3.- Modificar mi carrito
- 4.- Borrar producto de mi carrito
- 5.- Borrar mi carrito
- 6.- Confimar mi compra
- 7.- Salir :c

6

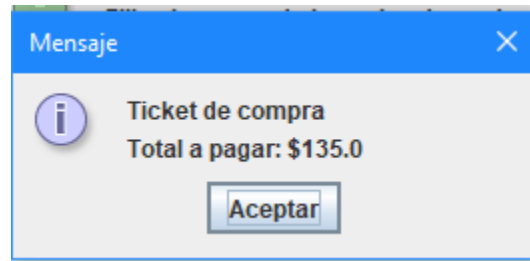
Aceptar Cancelar

Mensaje

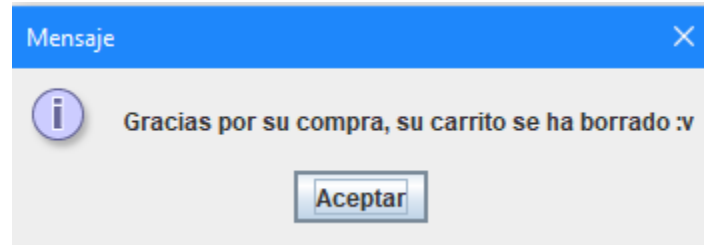
 Ticket de compra

ID	Cantidad de productos	Precio por unidad	Precio total por poductos
1	10	13.5	135.0

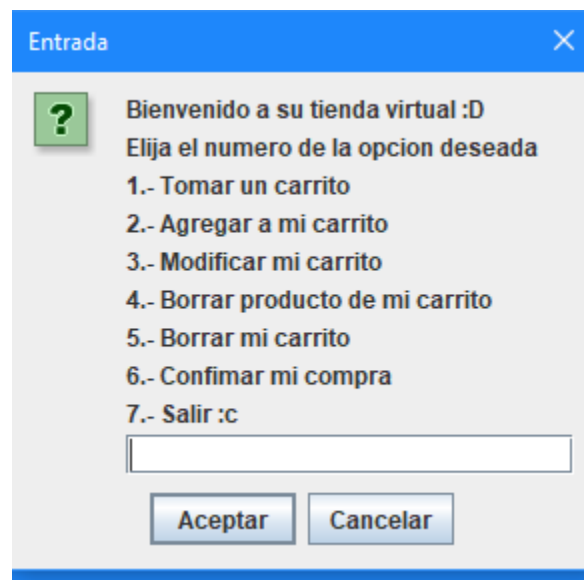
Aceptar



7.-Se borra el carrito y aparece la ventana



8.- aparece el menú principal



9.- Se puede hacer una nueva compra o simplemente salir de la aplicación

Conclusiones

Luciano Espina Melisa

Con esta práctica se pudo implementar los distintos tipos de métodos que ya se habían visto en clase, como son los “set” y “get”. Se aplicó para tener acceso a los atributos privados de las clases que se tenían, lo cuales se declararon de esa forma para que el usuario no tuviera acceso a esos atributos.

Para hacer el programa más eficiente y con menos líneas de código se utilizó los “Arraylist” los cuales nos permitieron optimizar el programa.

Se identificaron cada una de las relaciones y está más claro cómo utilizarlos y cómo declarar o hacer esas relaciones, por ejemplo para una composición debe estar instanciado en una de las clases un objeto del tipo de otra clase, para la asociación es solo pasar parámetros a un método de otra clase.

Bibliografía

[1] ArrayList en Java, con ejemplos – Jarroba | Jarroba [Online] Available: <https://jarroba.com/arraylist-en-java-ejemplos/>

[2] S/A Techlandia.com [Online] Available: https://techlandia.com/metodo-get-set-java-sobre_165891/