

CASO PRÁCTICO 10

- **TÍTULO: Programación orientada a objetos con Java**

- **SITUACIÓN**

Tenemos que resolver los siguientes problemas para la empresa de programación para la que trabajamos.

- **INSTRUCCIONES**

Desarrollar una aplicación en Java que permita gestionar una **lista de contactos**. La aplicación deberá ofrecer funcionalidades como agregar nuevos contactos, cargar contactos desde un archivo CSV, guardar los contactos en un archivo CSV y filtrar los contactos por categorías. Utilizando la biblioteca Swing, se debe crear una interfaz gráfica que permita a los usuarios interactuar con los datos de los contactos de manera sencilla.

Clase Contacto: Debes crear una clase llamada Contacto que tendrá los siguientes atributos:

- nombre (String): Nombre del contacto.
- telefono (String): Teléfono del contacto.
- email (String): Correo electrónico del contacto.
- categoria (String): Categoría del contacto (puede ser "Familia", "Amigos", "Trabajo", o "Todas").

Las funcionalidades de la aplicación deben ser:

- **Agregar un nuevo contacto:** Al hacer clic en el botón btnNuevo, el contacto debe ser agregado a la lista interna si todos los campos (nombre, teléfono, email y categoría) están correctamente completados.

Si alguno de los campos está vacío o la categoría no ha sido seleccionada, debe mostrarse un mensaje de advertencia.

- **Cargar contactos desde un archivo CSV:**

Al hacer clic en el botón btnCargar, se debe permitir al usuario seleccionar un archivo CSV que contenga contactos.

Los contactos deben ser leídos y añadidos a la lista interna, y se deben mostrar en el área de texto.

- **Guardar contactos en un archivo CSV:**

Al hacer clic en el botón btnGuardar, todos los contactos agregados deben guardarse en un archivo CSV llamado contactos.csv.

- **Filtrar contactos por categoría:** El ComboBox de categorías permitirá filtrar los contactos mostrados en el área de texto según la categoría seleccionada. Si se selecciona "Todas", deben mostrarse todos los contactos.

Cuando se seleccione una categoría específica (por ejemplo, "Familia"), solo se deben mostrar los contactos correspondientes a esa categoría.

Los campos de nombre, teléfono y email deben ser **obligatorios**. Si se intenta agregar un contacto sin llenar estos campos, se debe mostrar un mensaje de error.

La categoría no debe ser "Todas" al agregar un contacto; el usuario debe seleccionar una categoría específica para que el contacto sea agregado correctamente.

La aplicación debe permitir agregar hasta un **máximo de 30 contactos**.

El manejo de archivos debe ser robusto. Si se intenta cargar un archivo

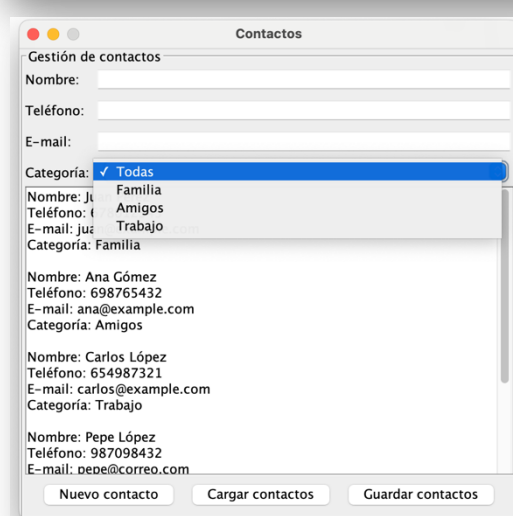
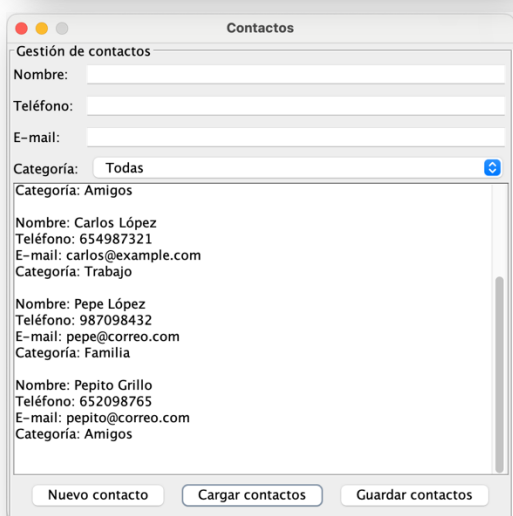
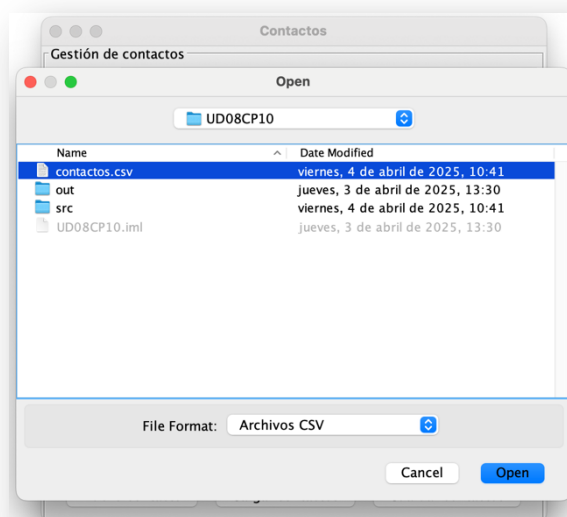
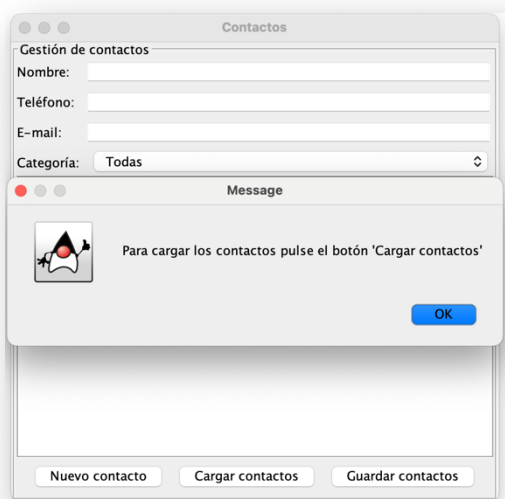
vacío o mal formado, debe mostrar un **mensaje de error** adecuado.

La interfaz debe ser clara y fácil de usar, con botones y mensajes adecuados para informar al usuario sobre el estado de las operaciones.

Al iniciar la aplicación se debe informar en un mensaje que para cargar los datos se debe pulsar sobre el botón Cargar contactos.

Cuando creamos un nuevo contacto NO se guarda directamente en el archivo, sino que se debe informar de que hay que pulsar sobre el botón Guardar contactos.

Ejemplo de ejecución:



Contactos

Gestión de contactos

Nombre:

Teléfono:

E-mail:

Categoría:

Nombre: Ana Gómez
Teléfono: 698765432
E-mail: ana@example.com
Categoría: Amigos

Nombre: Pepito Grillo
Teléfono: 652098765
E-mail: pepito@correo.com
Categoría: Amigos

Contactos

Gestión de contactos

Nombre:

Teléfono:

E-mail:

Categoría:

Categoría: Trabajo

Message

 Contacto agregado. Recuerde pulsar sobre 'Guardar Contactos' para guardarlo en el archivo

Nombre: Juan López Pérez
Teléfono: 987098432
E-mail: juanlopez@correo.com
Categoría: Amigos

Contactos

Gestión de contactos

Nombre:


Teléfono:

E-mail:

Categoría:

Categoría: Trabajo

Message

 Contactos guardado correctamente

Nombre: Juan López Pérez
Teléfono: 987098432
E-mail: juanlopez@correo.com
Categoría: Amigos