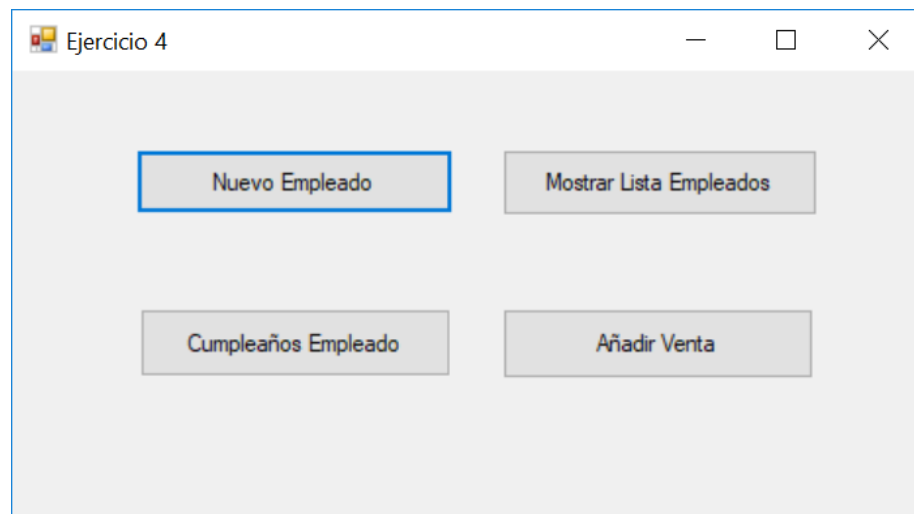


1. Crear la clase (Orientado a Objetos) necesaria para almacenar los datos de **una** persona. Los datos que se quieren almacenar serán su nombre, edad, teléfono, sexo y si está casado. Tendremos un botón desde el que leeremos los datos y otro botón desde el que los mostraremos.
2. Apoyándonos en el ejercicio anterior, realizar un programa que trabaje con una lista de personas. Realizar un botón para leer los datos de una persona y otro para imprimir los datos de todas las personas.
3. Escribir un programa que construya una lista de fechas (la fecha será una clase con tres campos: día, mes, año) con los siguientes botones:
  - ✓ Botón *leer*, que lee una fecha añadiéndola a la lista.
  - ✓ Botón *ordenar* que clasifica la lista anterior tomando como campo más importante el año, luego el mes y finalmente el día.
  - ✓ Botón *listar*, que imprime la lista de fechas.Podríais validar la fecha con los módulos realizados en prácticas anteriores para tal fin.

4. Realizar un ejercicio con el siguiente formulario:

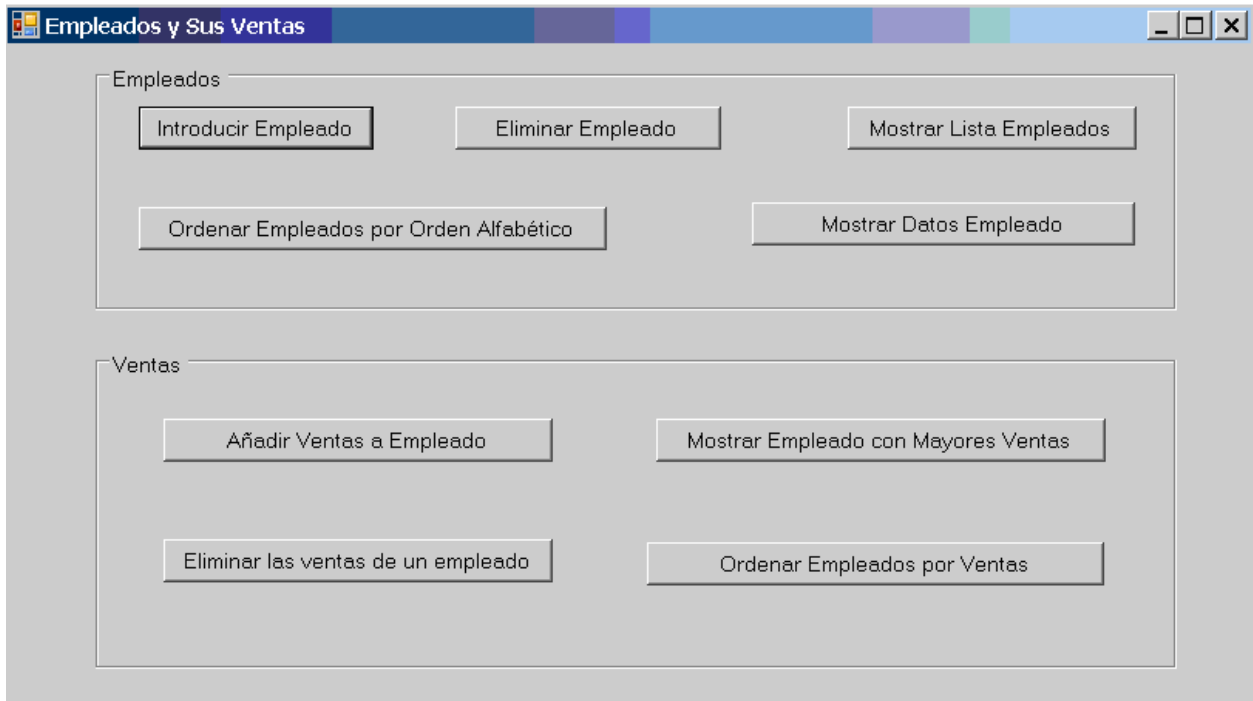


Este ejercicio trabajaremos con una lista de empleados. **Tanto el empleado como la lista serán clases de Visual C# .NET.**

- En el primer botón se leerá un nuevo empleado.

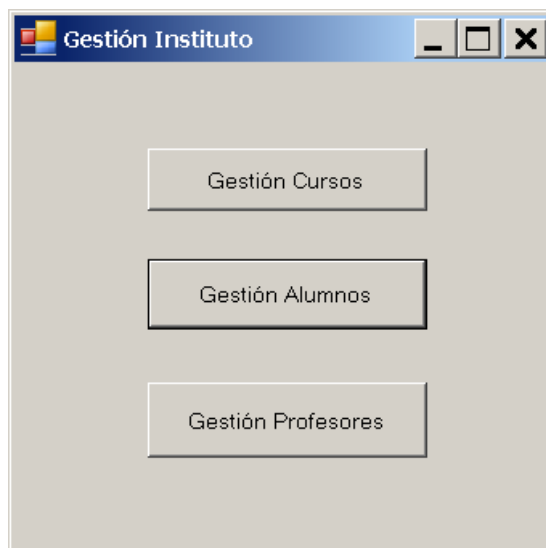
- El segundo botón mostrará los datos de todos los empleados que tenemos en nuestra lista.
- El botón de Cumpleaños de Empleado pedirá el **nombre** del empleado y le sumarán un año llamando al método cumpleAnyos del objeto.
- Añadir Venta a Empleado pedirá el **nombre** del empleado y permitirá introducirle ventas.

5. Utilizando las clases empleado y lista de empleados del ejercicio anterior implementar:



6. Queremos realizar un programa para gestionar un centro escolar.

Al **iniciar el programa** nos aparecerá una pantalla como la siguiente:



Este formulario inicial nos permitirá acceder a las distintas pantallas o formularios para gestionar los cursos, los alumnos y los profesores.

Tendremos que crear las siguientes clases en nuestro proyecto:

### CLASES:

A semejanza del ejercicio anterior tendremos las clases **Curso**, **ListadeCursos**, **Alumno**, **ListadeAlumnos**, **Profesor**, **ListadeProfesores**.

**Los cursos** tendrán los siguientes datos:

nombre y código.

**Los alumnos:**

nombre, dni, teléfono, lista de notas (simplemente una lista de valores double) y **código del curso** al cuál pertenecen.

**Los profesores:**

nombre, dni, teléfono, lista de nombres de las asignaturas que imparten y código del curso del cual son **tutores** (si lo son).

(Continúa en la página siguiente...)

En este ejercicio tendremos que **llamar a otros formularios** desde el formulario inicial.

Además, el objeto lista de alumnos, el de la lista de profesores y el objeto de la lista de cursos **se crearán en el formulario inicial** y por tanto **tendremos que pasarlos al formulario correspondiente** donde queramos manejarlos (cursos, alumnos, profesores).

Estos objetos tendrán el new en el formulario inicial y después en los distintos formularios tendremos instancias del mismo.

Es importante que **la clase ListaCursos sea pública** para que no haya problemas. Poner public delante de la definición de clase:

```
public class ListaCursos
```

Por ejemplo, **en el formulario de cursos** definiremos un objeto de tipo ListadeCursos pero no haremos new ya que recogeremos la lista de cursos que se creó en el formulario inicial. Además, cambiaremos el constructor para que tenga un parámetro a través del cual le pasaremos la lista de cursos creada en el formulario inicial:

```
// Declaramos la lista de cursos para pasarla desde el formulario inicial
private ListaCursos listaCursos;

public fCursos(ListaCursos listaCursos)
{
    InitializeComponent();

    // Ponemos en la lista de cursos del formulario la lista
    // de cursos que se pasa desde el formulario inicial
    this.listaCursos = listaCursos;
}
```

Vamos a ver la **forma de llamar a un formulario desde otro**.

Primero añadimos el nuevo formulario a nuestro proyecto y luego lo llamamos de la siguiente forma (en este caso desde el click del botón de cursos).

Además, **le pasamos al formulario la lista de cursos en el constructor** para que pueda manejarla:

```
// Creamos la lista de cursos.
ListaCursos listaCursos = new ListaCursos();
// Crear aquí también la lista de profesores y de alumnos.

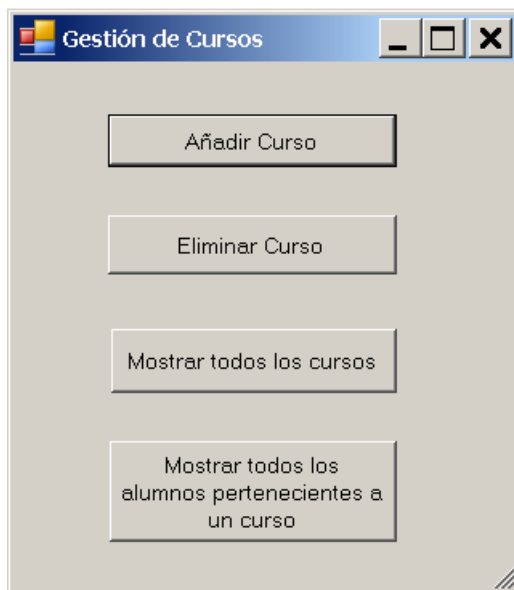
private void bCursos_Click(object sender, EventArgs e)
{
    // Creamos el formulario pasándole la lista de cursos creada en este...
    fCursos fCur = new fCursos(listaCursos);

    // Aquí mostramos el formulario, que ya tendrá la lista de cursos.
    fCur.ShowDialog();
}
```

De esta forma en cada botón de la pantalla inicial llamaremos a los formularios de cursos, alumnos y profesores.

## CURSOS:

Cuando se pulse el botón de cursos aparecerá una pantalla similar a la siguiente:

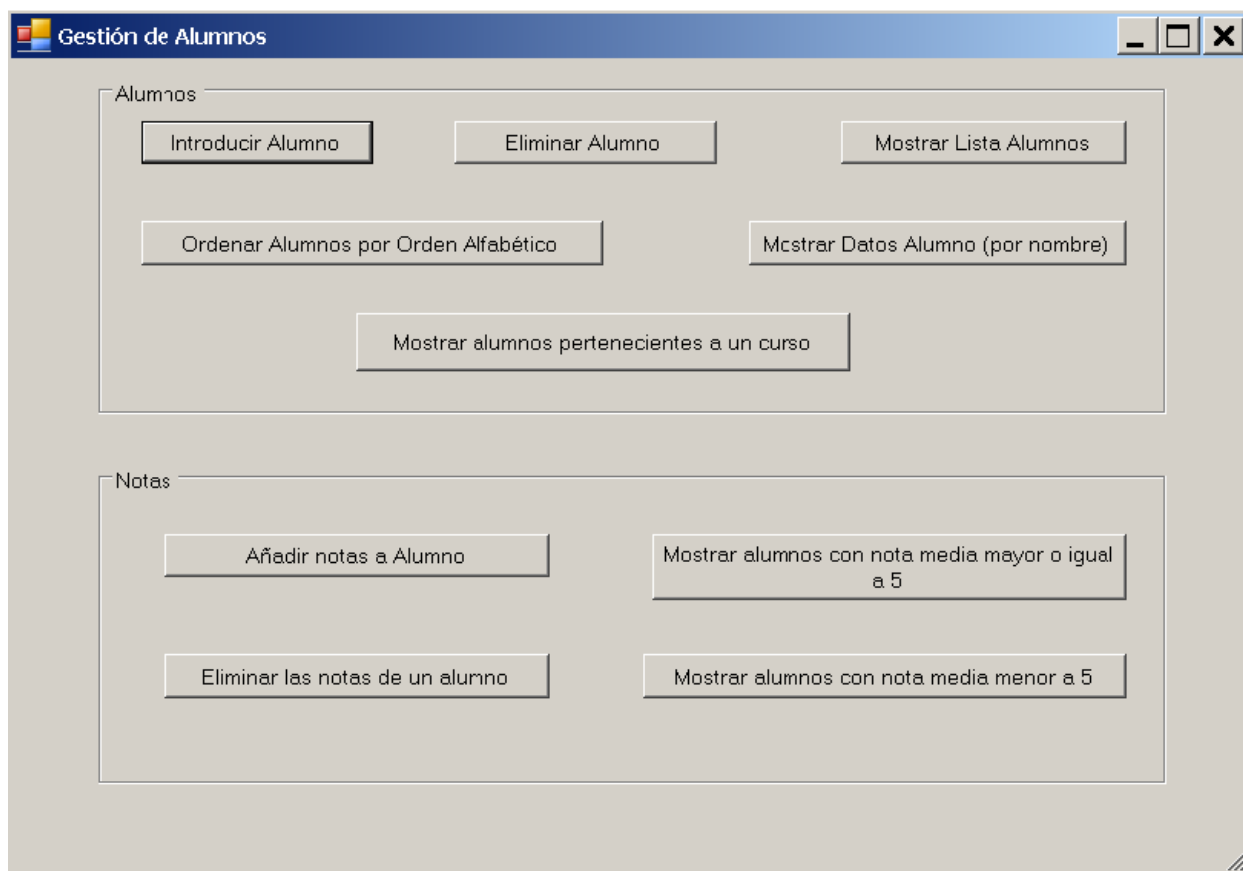


- Añadir curso nos permitirá introducir un nuevo curso (código y nombre).
- Eliminar curso, borrará un curso pidiendo su código.
- Mostrar todos los alumnos. Se introducirá un código de curso y se mostrará los datos de todos los alumnos pertenecientes a ese curso.

(Continúa en la página siguiente...)

## ALUMNOS:

Al entrar en la gestión de alumnos aparecerá una pantalla similar a la siguiente:

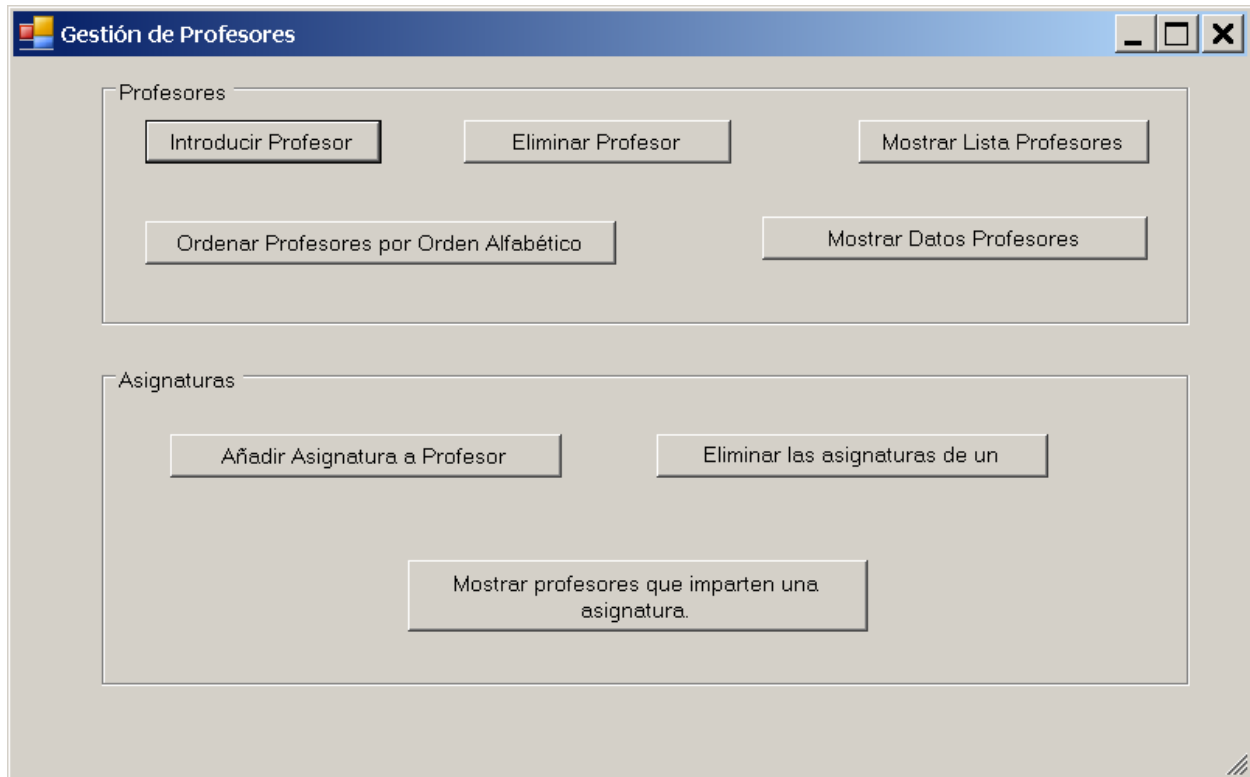


- Este formulario nos permite introducir un nuevo alumno, o eliminarlo (introduciendo el nombre del alumno).
- Mostrar todos los alumnos, ordenarlos por orden alfabético.
- Mostrar los datos del alumno cuyo nombre se introduzca, e introducir un curso y mostrar todos los alumnos de ese curso.
- Nos permitirá introducir un nombre y añadir o eliminar las notas de ese alumno.
- Y mostrar listas de alumnos con nota media mayor o menor que 5.

Se puede validar (no es obligatorio) por ejemplo que al introducir los datos del alumno (nombre, dni, tlf ...) el curso al que se le matricula debe existir en la lista de cursos.

## PROFESORES:

Al pulsar el botón de gestión de profesores aparecerá un formulario similar al siguiente:



The screenshot shows a Windows-style application window titled "Gestión de Profesores". It contains two main sections: "Profesores" and "Asignaturas".

**Profesores section:**

- Buttons: "Introducir Profesor", "Eliminar Profesor", "Mostrar Lista Profesores", "Ordenar Profesores por Orden Alfabético", and "Mostrar Datos Profesores".

**Asignaturas section:**

- Buttons: "Añadir Asignatura a Profesor", "Eliminar las asignaturas de un", and "Mostrar profesores que imparten una asignatura."

**Os paso el proyecto iniciado. Simplemente os pongo el formulario inicial, el comienzo de la clase ListaCursos y el formulario de cursos que se llama desde el inicial.**