*“ AÑO DE LA UNIDAD, LA PAZ Y EL DESARROLLO “*

**UNIVERSIDAD NACIONAL AGRARIA DE LA SELVA**

**FACULTAD DE INGENIERÍA INFORMÁTICA Y SISTEMAS**

**Logotipo

Descripción generada automáticamente**

**Diagrama

Descripción generada automáticamente**

**INFORME**

**“Proyecto de programa administrativo de una farmacia”**

**CURSO**:

Programación básica

**DOCENTE**:

Carlos A. Ríos Rivera

**INTEGRANTES:**

Calderón Carlos, Luis miguel

Rubina Savedra, Franco Daimbler

Martin Romero, Aaron

17 de mayo de 2023

**INDICE**

[RESUMEN ii](#_Toc135288303)

[1.1. TEMAS PRESENTES: iii](#_Toc135288304)

[1.2. INTRODUCCION iv](#_Toc135288305)

[1.2.1. El programa se divide en tres secciones principales: v](#_Toc135288306)

[1.3. DEFINICIÓN vi](#_Toc135288307)

[2. Capítulo II Fundamentos esenciales de la programación orientada a objetos 1](#_Toc135288308)

[2.1. Polimorfismo 1](#_Toc135288309)

[2.2. Hereda 1](#_Toc135288310)

[2.3. método 1](#_Toc135288311)

[2.4. Atributo. 2](#_Toc135288312)

[2.5. clase 2](#_Toc135288313)

[2.6. objeto 2](#_Toc135288314)

[3. Capítulo III ESTRUCTURA BASICA DEL PROGRAMA 4](#_Toc135288315)

[3.1. Interfaz de usuario 4](#_Toc135288316)

[3.2. Modulo de gestión de clientes 4](#_Toc135288317)

[3.3. Gestión del almacenamiento secundario 4](#_Toc135288318)

[3.4. Módulo de gestión de productos y medicamentos 4](#_Toc135288319)

[3.5. Módulo de programación de citas: 5](#_Toc135288320)

[3.6. Generación de informes y reportes: 5](#_Toc135288321)

[3.7. Seguridad y acceso: 5](#_Toc135288322)

[4. Capítulo IV 2](#_Toc135288323)

[5. IMPLEMENTACIÓN DE GETTERS Y SETTERS 2](#_Toc135288324)

[6. CLASE CLIENTES Y CITAS 4](#_Toc135288325)

[7. CLASE EMPLEADOS 5](#_Toc135288326)

[7.1. Clase Persona: 5](#_Toc135288327)

[7.2. Clase Empleado: 5](#_Toc135288328)

[7.3. Estructura Templeado: 5](#_Toc135288329)

[7.4. Funciones y métodos adicionales: 5](#_Toc135288330)

[CONCLUCION 7](#_Toc135288331)

[8. Bibliografía 8](#_Toc135288332)

# RESUMEN

Este es un programa administrativo de una farmacia. La intención de este programa es que con el paradigma orientado a objetos, desarrolláramos un programa de control de citas, inventario y empleados, usando archivos separados para cada clase y posteriormente incluirlas en un main con un #include.

## TEMAS PRESENTES:

**Programación orientada a objetos**: El programa utiliza clases y objetos para representar entidades como empleados, clientes y productos.

**Manipulación de archivos**: Se utilizan archivos para almacenar y recuperar información, como la información de los empleados y clientes.

**Estructuras de control:** Se utilizan estructuras de control como bucles y condicionales para controlar el flujo del programa y realizar diferentes acciones según las opciones seleccionadas por el usuario.

**Entrada y salida de datos:** Se utilizan funciones como cout y cin para mostrar mensajes en pantalla y obtener datos del usuario.

**Uso de bibliotecas estándar:** Se incluyen bibliotecas como iostream, fstream, conio.h, stdio.h y string.h para utilizar funciones y características predefinidas en C++.

**Herencia y polimorfismo:** No se aprecia directamente en el fragmento de código proporcionado, pero los archivos "recursosmateriales.cpp", "Empleados.cpp" y "ClientesYCitas.cpp" podrían contener clases que heredan de otras y hacen uso de polimorfismo.

**Algoritmos de búsqueda y modificación:** Hay partes del código que realizan operaciones de búsqueda y modificación de registros, tanto en los archivos de empleados y clientes como en la lista de productos.

En resumen, el programa de gestión para el laboratorio de análisis clínicos permite realizar las siguientes acciones:

Gestionar los recursos materiales del laboratorio, como productos y medicamentos, permitiendo dar de alta productos, buscar productos por código, modificar precios y generar reportes específicos.

Controlar el personal del laboratorio, permitiendo agregar empleados, mostrar la lista de empleados, buscar empleados por código, modificar datos de empleados, eliminar empleados y ordenar la lista de empleados por edad.

Manejar la información de los clientes del laboratorio, permitiendo agregar clientes, buscar clientes por código o nombre, modificar datos de clientes, eliminar clientes, establecer citas para los clientes, mostrar la lista de citas programadas y generar reportes específicos de clientes y exámenes.

El programa utiliza estructuras de datos como listas enlazadas para organizar la información de productos, empleados, clientes y citas. Además, se emplea el manejo de archivos para guardar y cargar la información de los empleados desde un archivo externo.

En general, el programa facilita la gestión eficiente de los recursos, empleados, clientes y citas en un laboratorio de análisis clínicos, brindando una herramienta centralizada para el control y manejo de la información relacionada con estas áreas.

## INTRODUCCION

El programa en cuestión es un sistema de gestión para un laboratorio de análisis clínicos. Permite llevar a cabo diversas operaciones relacionadas con los recursos materiales, el control de personal y el manejo de clientes y citas.

### El programa se divide en tres secciones principales:

Recursos Materiales: Esta sección permite gestionar los productos y recursos necesarios para el funcionamiento del laboratorio. Algunas de las operaciones que se pueden realizar son:

Dar de alta productos: Permite agregar nuevos productos al inventario del laboratorio, ingresando los campos requeridos como el código, tipo, existencia y precio.

Buscar un producto: Permite buscar un producto en el inventario a partir de su código.

Modificar precio de un producto: Permite modificar el precio de un producto existente en el inventario.

Generar reportes de productos: Permite generar reportes específicos de productos, como analgésicos, antibióticos y vitamínicos.

Control de Personal: Esta sección permite gestionar los empleados del laboratorio. Algunas de las operaciones que se pueden realizar son:

Agregar un empleado: Permite agregar un nuevo empleado al sistema, ingresando sus datos personales como nombre, código, tipo de empleado, etc.

Mostrar empleados: Muestra la lista de empleados registrados en el sistema.

Buscar un empleado: Permite buscar un empleado en el sistema a partir de su código.

Modificar datos de empleado: Permite modificar los datos de un empleado existente en el sistema.

Eliminar un empleado: Permite eliminar un empleado del sistema.

Agregar un producto: Permite agregar un nuevo producto al inventario, pero con permisos especiales solo para los empleados con el tipo "laboratorista".

Ordenar empleados por edad: Permite ordenar la lista de empleados por edad y mostrarlos en orden ascendente.

Manejo de Clientes y Citas: Esta sección permite gestionar los clientes del laboratorio y las citas programadas. Algunas de las operaciones que se pueden realizar son:

Agregar un cliente: Permite agregar un nuevo cliente al sistema, ingresando sus datos personales como nombre, código, etc.

Buscar un cliente por código o nombre: Permite buscar un cliente en el sistema a partir de su código o nombre.

Modificar datos del cliente: Permite modificar los datos de un cliente existente en el sistema.

Eliminar un cliente: Permite eliminar un cliente del sistema.

Establecer una cita: Permite programar una cita para un cliente, ingresando los detalles de la cita como fecha, hora, etc.

Mostrar lista de citas: Muestra la lista de citas programadas.

Generar reportes de clientes y exámenes: Permite generar reportes específicos de clientes y sus exámenes, como exámenes de sangre, nivel de azúcar y orina.

El programa utiliza estructuras de datos como listas enlazadas para gestionar los productos, empleados, clientes y citas. Además, se utiliza el manejo de archivos para guardar y cargar la información de empleados desde un archivo externo.

## DEFINICIÓN

El programa de gestión para el laboratorio de análisis clínicos es una aplicación informática que permite administrar y controlar de manera eficiente los recursos materiales, el personal y la información de los clientes en un laboratorio de análisis clínicos. A través de diversas funcionalidades, el programa permite dar de alta y gestionar productos, medicamentos y empleados, así como también realizar tareas como buscar, modificar y eliminar información relacionada con los clientes. Además, ofrece la posibilidad de establecer citas para los clientes y generar reportes específicos sobre los productos, empleados, clientes y exámenes realizados en el laboratorio. En resumen, el programa facilita la organización y gestión integral de un laboratorio de análisis clínicos, optimizando los procesos y mejorando la eficiencia en la atención y servicio brindado a los clientes.

**CAPÍTULO II**

# Capítulo II Fundamentos esenciales de la programación orientada a objetos

## Polimorfismo

El polimorfismo es la capacidad de un objeto de tomar diferentes formas o comportarse de diferentes maneras en función del contexto en el que se utilice. Permite que objetos distintos respondan de manera diferente a la misma llamada a un método, según su propia implementación. El polimorfismo facilita la flexibilidad y la extensibilidad del código, ya que se pueden utilizar diferentes objetos de una jerarquía de clases de manera intercambiable, sin conocer los detalles internos de cada uno.

## Hereda

La herencia es un mecanismo en la programación orientada a objetos que permite crear nuevas clases (llamadas subclases o clases hijas) a partir de una clase existente (llamada clase padre o superclase). Las subclases heredan los atributos y métodos de la clase padre y pueden agregar nuevos atributos y métodos, o modificar los existentes. La herencia facilita la reutilización de código, la organización jerárquica de las clases y la creación de relaciones de especialización.

## método

Un método es una función o procedimiento asociado a un objeto oa una clase. Define el comportamiento y las acciones que un objeto puede realizar. Los métodos permiten manipular y modificar los atributos de un objeto, así como realizar operaciones específicas. Proporcionan la funcionalidad de un objeto y pueden recibir parámetros y devolver resultados. Al recibir una interrupción, la computadora pone en espera cualquier programa que se esté ejecutando, guarda su estado y ejecuta el código del sistema operativo asociado con la interrupción.

## Atributo.

Un atributo es una variable o dato que está asociado a un objeto específico. Representa una característica o estado del objeto y almacena información relevante sobre el mismo. Los atributos pueden ser de diferentes tipos, como números, texto, booleanos, entre otros. Permiten definir las características distintivas y la información que puede ser almacenada y accedida por los métodos del objeto.

## clase

Una clase es una plantilla o molde para crear objetos. Define las propiedades y comportamientos comunes que tendrán los objetos que se creen a partir de ella. Las clases encapsulan datos y funciones relacionadas en un solo lugar, lo que facilita la reutilización de código y la creación de múltiples objetos con características similares.

## objeto

Un objeto es una instancia concreta de una clase en la programación orientada a objetos. Representa un elemento o entidad que tiene propiedades (atributos) y comportamientos (métodos). Los objetos son utilizados para modelar y manipular datos de forma modular y estructurada.

**CAPÍTULO III**

# Capítulo III ESTRUCTURA BASICA DEL PROGRAMA

## Interfaz de usuario

El programa contaría con una interfaz gráfica intuitiva y fácil de usar para que los usuarios puedan interactuar con él. Esto incluiría menús, botones y campos de entrada para acceder a diferentes funciones y visualizar información relevante.

## Modulo de gestión de clientes

Este módulo permitiría dar de alta nuevos clientes en la base de datos del laboratorio, almacenando información como nombre, dirección, número de teléfono, historial médico, entre otros. También se podrían buscar, modificar y eliminar registros de clientes existentes.

## Gestión del almacenamiento secundario

Aquí se registrarían los datos de los empleados del laboratorio, incluyendo nombres, cargos, horarios, información de contacto, etc. Este módulo permitiría la asignación de tareas, control de horarios y gestión de permisos.

## Módulo de gestión de productos y medicamentos

En este módulo se registrarían los productos y medicamentos utilizados en el laboratorio, incluyendo su descripción, proveedor, fecha de vencimiento, entre otros datos relevantes. Se podrían realizar búsquedas, modificar registros y controlar el inventario.

## Módulo de programación de citas:

Este módulo permitiría a los usuarios programar citas para los clientes, asignando fechas y horarios específicos para realizar los análisis clínicos. También podría ofrecer recordatorios automáticos de citas.

## Generación de informes y reportes:

El programa contaría con funcionalidades para generar informes y reportes personalizados. Esto podría incluir informes de análisis realizados, resultados de exámenes, facturación, entre otros.

## Seguridad y acceso:

Se implementarían mecanismos de seguridad para proteger la información confidencial y restringir el acceso a usuarios autorizados. Esto podría incluir autenticación de usuarios, niveles de permisos y registro de actividades.

**CAPÍTULO IV**

# Capítulo IV

# IMPLEMENTACIÓN DE GETTERS Y SETTERS

En la clase “empleados”, los getters y setters se implementan como métodos miembros de las clases Persona y Empleado. Estos métodos permiten acceder y modificar los atributos privados de las clases. Aquí está una explicación de cómo se implementan los getters y setters en el programa:

Dentro de la clase Persona, se definen los atributos privados nom, edad y cod.

Los métodos setNom, setEdad y setCod se utilizan para establecer los valores de los atributos nom, edad y cod respectivamente. Cada método recibe los valores correspondientes como argumentos y actualiza los atributos.

Los métodos getNom, getEdad y getCod se utilizan para obtener los valores de los atributos nom, edad y cod respectivamente. Cada método devuelve el valor del atributo correspondiente.

Dentro de la clase Empleado, se definen los atributos privados suel, dir y tipoEm.

Los métodos setSuel, setDir y setTipoEm se utilizan para establecer los valores de los atributos suel, dir y tipoEm respectivamente. Cada método recibe los valores correspondientes como argumentos y actualiza los atributos.

Los métodos getSuel, getDir y getTipoEm se utilizan para obtener los valores de los atributos suel, dir y tipoEm respectivamente. Cada método devuelve el valor del atributo correspondiente.

En el programa, los getters y setters se utilizan en varias partes del código, por ejemplo, en la función AgregarEmp, se llaman a los métodos setNom, setEdad, setCod, setSuel, setDir y setTipoEm para establecer los valores de los atributos del objeto empl de la clase Empleado.

Del mismo modo, en la función MostrarEmp, se llaman a los métodos getNom, getEdad, getCod, getSuel, getDir y getTipoEm para obtener los valores de los atributos del objeto empl y mostrarlos en la salida.

Los getters y setters proporcionan una forma controlada de acceder y modificar los atributos de una clase, lo que ayuda a mantener la encapsulación y facilita la manipulación de los datos en el programa.Subsistema de Archivos

Proporciona una vista uniforme de todas las formas de almacenamiento, implementando el concepto de archivo como una colección de bytes. El Sistema Operativo deberá proveer métodos para:

• Abrir, cerrar y crear archivos

• Leer y escribir archivos

# CLASE CLIENTES Y CITAS

void AgregarCli(): Permite agregar un nuevo cliente al archivo "clientes.dat" solicitando al usuario que ingrese los detalles del cliente.

void MostrarCli(): Muestra los detalles de todos los clientes almacenados en el archivo "clientes.dat".

void BuscarCli(int): Busca un cliente por su código y muestra los detalles correspondientes.

void BuscarCli(char\*): Busca un cliente por su nombre y muestra los detalles correspondientes.

void ModificarCli(): Permite modificar los detalles de un cliente existente.

void EliminarCli(): Permite eliminar un cliente existente.

void EstablecerCit(): Permite establecer una nueva cita solicitando al usuario que ingrese los detalles de la cita y lo guarda en el archivo "citas.dat".

void ListaCit(): Muestra los detalles de todas las citas almacenadas en el archivo "citas.dat".

void ReporteS(): Genera un informe de citas filtrado por el tipo de examen "Sangre".

void ReporteNA(): Genera un informe de citas filtrado por el tipo de examen "Nivel Azucar".

# CLASE EMPLEADOS

## Clase Persona:

Esta clase representa a una persona genérica y tiene tres atributos: nom (nombre), edad y cod (código).

Los métodos setNom, setEdad y setCod se utilizan para establecer los valores de los atributos.

Los métodos getNom, getEdad y getCod se utilizan para obtener los valores de los atributos.

## Clase Empleado:

Esta clase hereda de la clase Persona y agrega tres atributos adicionales: suel (sueldo), dir (dirección) y tipoEm (tipo de empleado).

Los métodos setSuel, setDir y setTipoEm se utilizan para establecer los valores de los atributos adicionales.

Los métodos getSuel, getDir y getTipoEm se utilizan para obtener los valores de los atributos adicionales.

## Estructura Templeado:

Esta estructura se utiliza para almacenar los datos de un empleado en un formato específico para su escritura y lectura en el archivo "empleado.dat".

## Funciones y métodos adicionales:

AgregarEmp: Permite agregar un nuevo empleado al archivo "empleado.dat" solicitando al usuario que ingrese los detalles del empleado.

MostrarEmp: Muestra todos los empleados almacenados en el archivo "empleado.dat".

BuscarEmp: Permite buscar un empleado en el archivo "empleado.dat" utilizando el código del empleado como criterio de búsqueda.

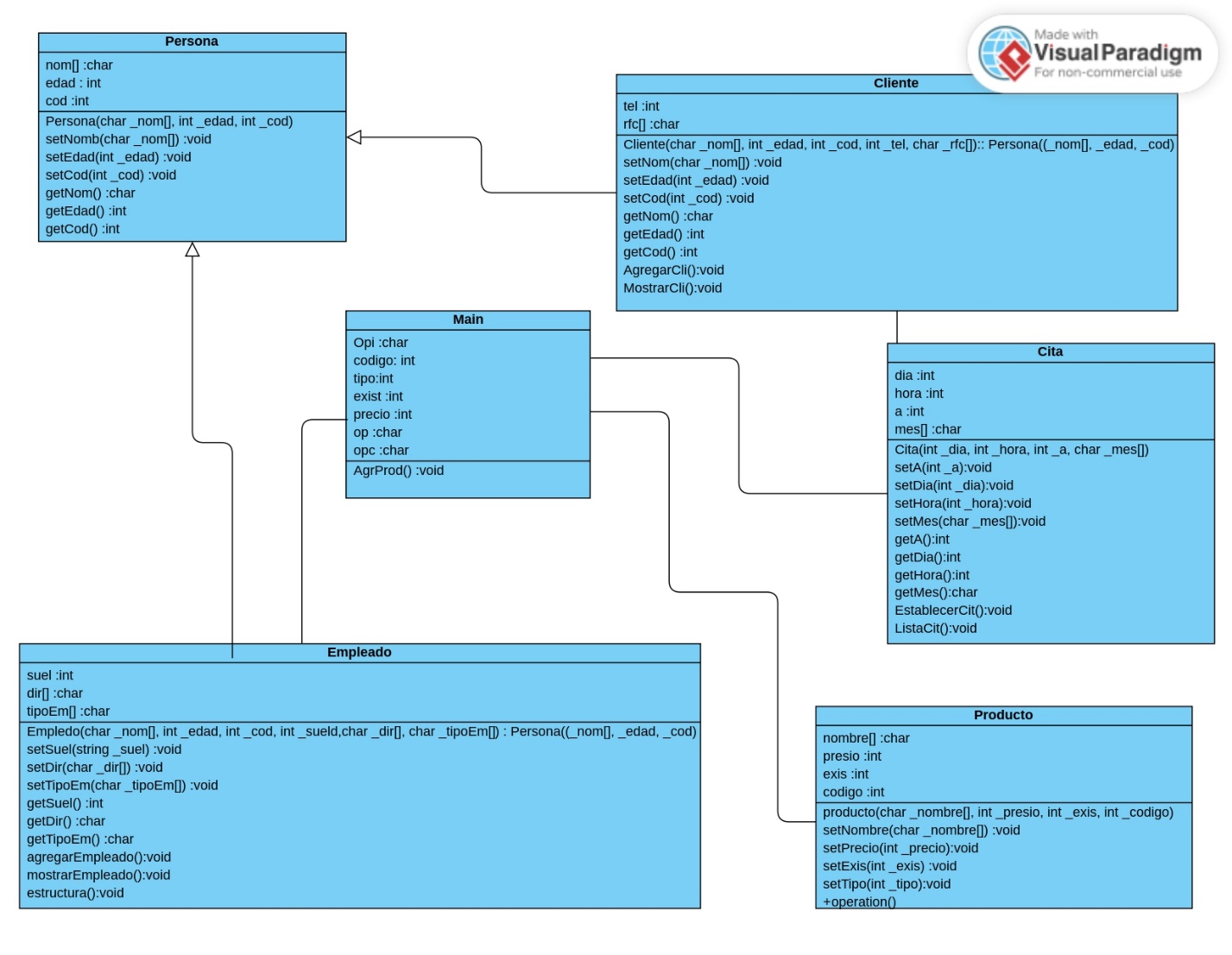
ModificarEmp: Permite modificar los datos de un empleado existente en el archivo "empleado.dat" utilizando el código del empleado como criterio de búsqueda.

EliminarEmp: Permite eliminar un empleado del archivo "empleado.dat" utilizando el código del empleado como criterio de búsqueda.

Estructura: Lee los datos de los empleados del archivo "empleado.dat" en una matriz de estructuras Templeado y los ordena por edad utilizando el algoritmo de ordenamiento de burbuja.

burbuja: Implementa el algoritmo de ordenamiento de burbuja para ordenar la matriz de empleados por edad.

El programa utiliza operaciones de entrada y salida de archivos para escribir y leer datos desde el archivo "empleado.dat". Los datos se almacenan en una estructura Templeado antes de escribirse en el archivo y se leen desde el archivo en una matriz de estructuras Templeado para mostrar y ordenar los datos.



# CONCLUCION

En conclusión, hemos desarrollado un sistema de gestión para un laboratorio de análisis clínicos que abarca diversas áreas, como recursos materiales, control de personal y manejo de citas. Este sistema permite llevar un registro organizado de los productos disponibles, así como realizar búsquedas y modificaciones según sea necesario. Además, cuenta con funcionalidades para gestionar el personal del laboratorio, desde agregar nuevos empleados hasta realizar modificaciones en sus datos o eliminar registros.

En cuanto al manejo de clientes, el sistema ofrece la posibilidad de agregar nuevos clientes, buscarlos por código o nombre, y establecer citas para realizar diferentes tipos de exámenes. Además, se pueden generar reportes específicos de los resultados de los exámenes, como análisis de sangre, nivel de azúcar u orina.

Nuestro sistema proporciona una interfaz amigable y fácil de usar, con opciones claras y precisas. También hemos implementado medidas de seguridad para garantizar el acceso adecuado a las funcionalidades según el rol del empleado. Por ejemplo, solo los laboratoristas tienen permiso para dar de alta nuevos productos.

En resumen, el sistema de gestión desarrollado para el laboratorio de análisis clínicos brinda eficiencia y orden en el manejo de recursos materiales, personal y citas de los clientes. Facilita el seguimiento y control de las actividades diarias del laboratorio, mejorando la calidad del servicio brindado.

# Bibliografía

[Programación orientada a objetos | Blog UE (universidadeuropea.com)](https://universidadeuropea.com/blog/programacion-orientada-objetos/#:~:text=La%20programaci%C3%B3n%20orientada%20a%20objetos%20es%20un%20modelo%20de%20programaci%C3%B3n,l%C3%B3gica%20necesaria%20para%20esa%20manipulaci%C3%B3n.)

[Libro: Introducción a la programación C++ para ingenieros - 9788413404219 - Gracia del Río, Jorge - Júlvez Bueno, Jorge - Miguel Casado, Gregorio de - · Marcial Pons Librero](https://www.marcialpons.es/libros/introduccion-a-la-programacion-c-para-ingenieros/9788413404219/)

[¿Qué es la Programación Orientada a Objetos (POO)? | EDteam](https://ed.team/blog/que-es-la-programacion-orientada-a-objetos-poo)

[CURSO-PROGRA-C-/CURSO-PROGRA-C-/PROYECTO FINAL/20 trabajo at main · Daimblerubina/CURSO-PROGRA-C- · GitHub](https://github.com/Daimblerubina/CURSO-PROGRA-C-/tree/main/CURSO-PROGRA-C-/PROYECTO%20FINAL/20%20trabajo)