

UNIVERSIDAD NACIONAL JORGE BASADRE GROHMANN

FACULTAD DE INGENIERÍA

Escuela Profesional de Ingeniería en Informática y Sistemas



Informe final

Proyecto: *Sistema de Gestión de Citas Médicas*

Curso: Programación avanzada

Docente: MSc. Ing. Hugo Manuel Barraza Vizcarra

Integrantes:

Limache Condori Dennis Jhosue (2025 – 119029)

Cristopher Diego Enrique Cauna Mamani (2025 – 119001)

Chipana Mamani Farid Roy Leo (2025 – 119018)

**Tacna – Perú
2025**

ÍNDICE GENERAL

1. Resumen	3
2. Antecedentes	3
3. Planteamiento del problema	3
a. Problema	
b. Justificación	
c. Alcance	
4. Objetivos	4
5. Marco teórico	4
6. Desarrollo de la solución	5
a. Tecnología de Desarrollo	
b. Metodología de implementación (VISIÓN, SRS, SAD)	
c. Especificación de requerimientos de software	
7. Conclusiones y recomendaciones	6
8. Bibliografía	6
9. Anexos	6

1. Resumen

El proyecto consiste en el desarrollo de una aplicación de consola en lenguaje C++ para la gestión de citas médicas. El sistema permite registrar pacientes, asignar prioridades de atención (urgencia, normal o baja), buscar pacientes por DNI y generar reportes en archivos de txt. Se ha aplicado el paradigma de Programación Orientada a Objetos (POO), implementando herencia, polimorfismo y encapsulamiento, así como el manejo de archivos para la persistencia de datos y algoritmos de ordenamiento para gestionar la cola de espera según la gravedad del paciente.

2. Antecedentes

Actualmente, muchos centros de salud pequeños o postas médicas hacen el registro de pacientes de manera manual utilizando muchas veces cuadernos o un Excel básico. Esto a veces ocasiona problemas como la pérdida de información, una dificultad para ordenar a los pacientes según su urgencia y lentitud al momento de buscar el historial de una cita. No existe una validación automática de datos ni una forma rápida de emitir reportes diarios.

3. Planteamiento del problema

a. Problema

La gestión manual de citas médicas genera una ineficiencia en la atención, llega a ocurrir errores en la priorización de pacientes graves y hay también el riesgo de pérdida de datos al no contar con un sistema de almacenamiento persistente.

b. Justificación

Por ello es necesario desarrollar un software que automatice estos procesos para garantizar que los pacientes con mayor urgencia sean atendidos primero y que la información quede guardada de forma segura. Además, este proyecto permite evidenciar las competencias adquiridas en la asignatura de Programación Avanzada, específicamente en el uso de punteros, clases polimórficas y manipulación de streams de datos.

c. Alcance

- El sistema está dirigido al personal de recepción de un centro médico. Incluye:
- Registro de citas presenciales con validación de datos.
- Listado general de citas.
- Búsqueda de pacientes mediante DNI.
- Ordenamiento automático de citas por nivel de prioridad.
- Generación de reportes en archivos .txt.
- Persistencia de datos (carga y guardado automático).

4. Objetivos

a. General

Desarrollar una aplicación en C++ bajo el paradigma orientado a objetos que solucione la gestión de citas médicas, asegurando la persistencia de datos y la correcta priorización de pacientes.

b. Específicos

- Implementar una arquitectura de clases utilizando Herencia (Clase padre Cita y derivada CitaPresencial y Polimorfismo (métodos virtuales).
- Desarrollar algoritmos de Búsqueda secuencial para localizar citas por DNI.
- Implementar Ordenación para organizar la lista de espera según la prioridad de atención.
- Utilizar flujos de archivos (fstream) para la lectura y escritura de datos en formato de texto (citas_data.txt y reporte_citas.txt).

5. Marco teórico

- **Programación Orientada a Objetos (POO):** Paradigma que organiza el diseño de software alrededor de objetos, mediante características y comportamientos.
- **Polimorfismo:** Capacidad de una clase derivada de redefinir métodos de la clase base (virtual) para comportarse de manera distinta. En nuestro proyecto, se usa en obtenerDetalles().
- **Persistencia de Datos:** Capacidad del software para mantener la información disponible a través del tiempo, lograda mediante archivos de texto plano.

- **Gestión de Memoria Dinámica:** Uso de punteros (new, delete) para manejar la lista de citas en tiempo de ejecución.

6. Desarrollo de solución

a. Tecnologías de desarrollo

- Lenguaje: C++
- Entorno de Desarrollo (IDE): Visual Studio Code.
- Control de Versiones: Git y GitHub para el trabajo colaborativo.
- Compilador: MinGW (G++).

b. Metodología de implementación

Se utilizó una metodología incremental. Primero se diseñaron las clases base (Cita), luego se implementó la lógica de gestión en memoria (Gestor), posteriormente se añadió la persistencia en archivos y finalmente se desarrolló la interfaz de consola.

c. Especificación de requerimientos de software

- Requerimientos Funcionales:

#	Descripción
RF-01	Registrar Cita: El sistema permite ingresar DNI, nombre, prioridad, especialidad y consultorio.
RF-02	Listar Citas: Muestra en pantalla todas las citas registradas con su formato detallado.
RF-03	Buscar por DNI: Permite localizar rápidamente si un paciente ya tiene cita.
RF-04	Ordenar por Prioridad: Reorganiza la lista para que las prioridades "Alta" (1) aparezcan primero.

- Requerimientos No Funcionales:

Necesidad	ID Req.	Descripción
-----------	---------	-------------

1.	Facilidad de uso e ingreso de información	RNF-01	El sistema debe contar con una interfaz de consola intuitiva y menús claros que guíen al personal de recepción, mostrando mensajes de confirmación tras cada registro exitoso de una cita.
2	Persistencia y Seguridad	RNF-02	El sistema debe asegurar la integridad de los datos mediante el almacenamiento automático en archivos de texto, permitiendo que la información de las citas persista correctamente tras el cierre de la aplicación.

d. Arquitectura de software

En la Figura 1 se muestra la organización del sistema mediante clases. La clase Cita fue definida como la base para representar una cita médica, donde se almacena la información general del paciente. A partir de esta clase se derivó la clase CitaPresencial, en la cual se agregó información específica como la especialidad y el número de consultorio. Las citas fueron administradas por la clase GestorCitas, que se encarga de registrar, buscar, ordenar y guardar los datos del sistema. Esta estructura permite que el sistema funcione de manera ordenada y facilite futuras mejoras.

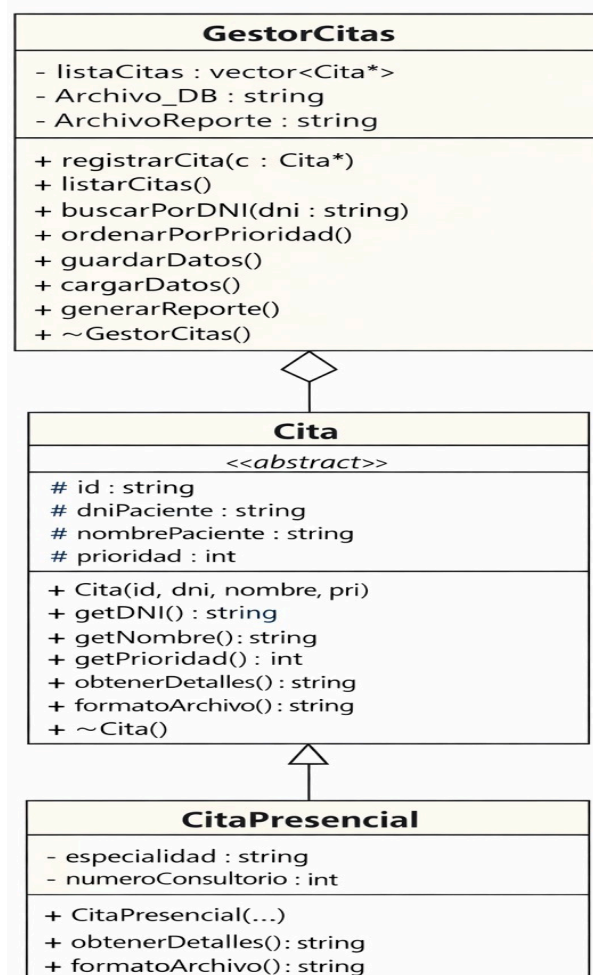


Figura 1) Diagrama UML

7. Conclusiones y recomendaciones

Conclusiones:

- Se logró implementar exitosamente un sistema que utiliza Polimorfismo para tratar diferentes tipos de citas de manera uniforme, facilitando la escalabilidad del código.
- El uso de archivos de texto permitió cumplir con el requisito de persistencia, asegurando que los registros médicos no se pierdan al cerrar el programa.
- Los algoritmos de ordenamiento implementados son fundamentales para el caso de uso médico, donde la prioridad de atención es crítica.

Recomendaciones:

- Para futuras versiones, se recomienda implementar una interfaz gráfica (GUI) utilizando librerías como Qt.
- Agregar un módulo de "Doctores" para asignar citas a médicos específicos y validar horarios disponibles.

8. Bibliografía

Joyanes Aguilar, L. (2008). Programación en C++: Algoritmos, Estructuras de Datos y Objetos (2.ª ed.). McGraw-Hill Interamericana.

https://books.google.com.pe/books/about/Programaci%C3%B3n_en_C++_Algoritmos_estructu.html?id=BeBrAAAACAAJ&hl=es-US&redir_esc=y

Input / output with files. Cplusplus.com

<https://cplusplus.com/doc/tutorial/files/?hl=es-US>

9. Anexos

Enlace al Repositorio Público:

https://github.com/fchipanam/Proyecto_Citas

Capturas de pantalla:

Imagen 1) Menú funcionando correctamente.

```
=== SISTEMA DE GESTION DE CITAS MEDICAS UNJBG ===
1. Registrar Nueva Cita
2. Ver Todas las Citas
3. Buscar Cita por DNI
4. Ordenar Citas por Prioridad (Urgencia)
5. Generar Reporte TXT
6. Guardar y Salir
*seleccione una opcion:
█
```

Imagen 2 y 3) Datos registrados.

```
data > citas_data.txt
1 P|C001|71234567|Juan Perez|1|Medicina General|101
2 P|C002|73456789|Maria Lopez|2|Pediatria|102
3 P|C003|75678912|Carlos Rojas|3|Odontologia|103
4 P|C004|77891234|Ana Torres|1|Traumatologia|104
5 P|C005|70123456|Luis Mendoza|2|Cardiologia|105
6 P|C006|72345678|Rosa Quispe|3|Ginecologia|106
7 P|C007|74567891|Pedro Salazar|1|Neurologia|107
8 P|C008|76789123|Elena Huaman|2|Oftalmologia|108
9 P|C009|78912345|Miguel Castro|3|Dermatologia|109
10 P|C010|79012346|Carmen Flores|1|Endocrinologia|110
```

Imagen 3) Generar Reporte usando datos registrados.

```
reportes > reporte_citas.txt
1  ---REPORTE DE CITAS---
2  =====
3  ID: C001 | Paciente: Juan Perez (DNI: 71234567)
4  | -> Tipo: Presencial | Consultorio: 101 | Especialidad: Medicina General | Prioridad: URGENCIA
5  | -----
6  ID: C002 | Paciente: Maria Lopez (DNI: 73456789)
7  | -> Tipo: Presencial | Consultorio: 102 | Especialidad: Pediatria | Prioridad: Normal
8  | -----
9  ID: C003 | Paciente: Carlos Rojas (DNI: 75678912)
10 | -> Tipo: Presencial | Consultorio: 103 | Especialidad: Odontologia | Prioridad: Baja
11 | -----
12 ID: C004 | Paciente: Ana Torres (DNI: 77891234)
13 | -> Tipo: Presencial | Consultorio: 104 | Especialidad: Traumatologia | Prioridad: URGENCIA
14 | -----
15 ID: C005 | Paciente: Luis Mendoza (DNI: 70123456)
16 | -> Tipo: Presencial | Consultorio: 105 | Especialidad: Cardiologia | Prioridad: Normal
17 | -----
18 ID: C006 | Paciente: Rosa Quispe (DNI: 72345678)
19 | -> Tipo: Presencial | Consultorio: 106 | Especialidad: Ginecologia | Prioridad: Baja
20 | -----
21 ID: C007 | Paciente: Pedro Salazar (DNI: 74567891)
22 | -> Tipo: Presencial | Consultorio: 107 | Especialidad: Neurologia | Prioridad: URGENCIA
23 | -----
24 ID: C008 | Paciente: Elena Huaman (DNI: 76789123)
25 | -> Tipo: Presencial | Consultorio: 108 | Especialidad: Oftalmologia | Prioridad: Normal
26 | -----
27 ID: C009 | Paciente: Miguel Castro (DNI: 78912345)
28 | -> Tipo: Presencial | Consultorio: 109 | Especialidad: Dermatologia | Prioridad: Baja
29 | -----
30 ID: C010 | Paciente: Carmen Flores (DNI: 79012346)
31 | -> Tipo: Presencial | Consultorio: 110 | Especialidad: Endocrinologia | Prioridad: URGENCIA
32 | -----
```