Proyecto Taller de Programación I

«Sistema de Agendamiento Médico Electrónico (S. A. M.)»

Informe Final

Estudiantes : Álvaro Molina Jara
Juliano Muñoz Sepúlveda
Florencia Staforelli Reyes

Sección : 01

Carrera : Ingeniería Civil Informática

Docente : Braulio Quiero Hernández

Concepción, 28 de diciembre de 2021

Índice

Introducción

El objetivo de este informe es presentar información detallada acerca del proyecto final del curso de **Taller de Programación I (IN1045C)** impartido dentro de la carrera de **Ingeniería Civil Informática** en la **Universidad Católica de la Santísima Concepción (UCSC)**, el cual consiste en el desarrollo de un programa en lenguaje de programación C que pueda contribuir a resolver un problema que afecta a un sector específico del ámbito social o laboral.

Dicho programa, se denomina **Sistema de Agendamiento Médico Electrónico (S. A. M.)**, el cual consiste en un software que permitirá administrar con mayor agilidad y celeridad las horas médicas, mejorando los procesos de gestión administrativa al interior de las instituciones de salud. Los temas relevantes que se detallarán del programa a continuación son:

- **Definición del problema:**
  - Identificación del problema
  - Identificación de funcionalidades.
  - Diseño de solución.

- **Etapa de elaboración:**
  - Justificación de algoritmos y estructuras.
  - Diagrama de nodos de estructuras relevantes.
  - Diagrama de flujo de datos de algoritmos relevantes.

- **Proceso de construcción:**
  - Funcionalidades de la aplicación.
  - Pormenores en el diseño y construcción de la aplicación.

Identificación del problema

Descripción del problema

En las instituciones de salud en general, se puede evidenciar la falta de un sistema automatizado que pueda gestionar las horas agendadas de cada médico, lo cual puede llevar a diversos inconvenientes debido a la caligrafía de un profesional de la salud, la cantidad de libros que se llenan (ya sean físicos o en Excel), o demoras al revisar los registros antes de poder encontrar lo que se está buscando, junto a los posibles errores humanos que pueda conllevar lo anterior.

Para solucionar esta problemática, como grupo creamos un programa **Sistema de Agendamiento Médico Electrónico (S. A. M.)**, el cual estará desarrollado mediante lenguaje de programación C y podrá ser ejecutado desde una terminal (por ejemplo: CMD (Símbolo del Sistema) o PowerShell en Windows, Bash en GNU/Linux y Mac, entre otros).

Este programa tendrá como objetivo facilitar la gestión en el agendamiento de horas médicas que tenga cada especialista de la salud dentro de un consultorio, hospital o clínica, permitiendo que éstos puedan prestar un servicio más eficaz hacia sus pacientes gracias a la inmediatez que tendrá este software a la hora de registrar o buscar la información solicitada. Lo anterior, facilitará la labor diaria del personal médico y de secretarios/as, beneficiando también a los pacientes que podrán invertir menos tiempo en trámites administrativos (tales como agendar una hora, confirmar una hora, entre otros) y dedicar la mayor parte de él para el objetivo principal por el cual se dirigió al recinto de salud (recibir la atención médica requerida).

Interesados

- **Médicos**
  - **Descripción:** Profesionales de la salud que prestan servicios en un hospital, clínica o consultorio.
  - **Interés:** Poder agendar, editar o consultar las horas médicas que tiene a su haber, evitando cualquier error humano al registrar o asignar las citas (por ejemplo: choques de horario), permitiendo al especialista poder organizar sus tiempos de mejor manera.

- **Secretarios/as**
  - **Descripción:** Trabajadores/as que desempeñan labores de carácter administrativo al interior de una institución de salud (redactar, guardar y ordenar documentos, gestionar agenda de médicos, entre otros).
  - **Interés:** Agendar, editar o consultar los registros de horas médicas de los profesionales de la salud que prestan servicios en un hospital, clínica o consultorio, facilitando labores administrativas (por ejemplo: buscar o confirmar citas agendadas previamente) y permitiendo a la institución prestar un mejor servicio a sus pacientes.

- **Pacientes**
  - **Descripción:** Persona que requiere poder recibir atención médica por parte de un especialista en una institución de salud.
  - **Interés:** Solicitar agendar una cita médica o consultar una hora programada con anterioridad, de forma ágil y confiable, permitiendo destinar menos tiempo dentro de un recinto de salud para trámites administrativos, y tener la confianza de que podrá atenderse con un especialista en la fecha y hora agendadas previamente.

Identificación de funcionalidades

¿Qué hace el programa?

Este programa para el usuario contará al principio con un menú principal que le permitirá tener acceso a las diversas herramientas que se proporcionan. Para desplazarse por las diversas utilidades del software, el usuario deberá digitar el número correspondiente de la opción que se muestra por pantalla dentro de la terminal.

Entre las principales funcionalidades con las que cuenta el programa, se destaca una interfaz para agendar horas médicas (consignando RUN, nombre y apellidos del paciente, la especialidad médica (donde se mostrará los médicos con agenda libre), y la fecha y hora para la que solicita la atención), buscar registros de citas agendadas previamente (por RUN, por nombre y por fecha), eliminar horas ya programadas, entre otras opciones.

Entradas, salidas y descripción del procedimiento a realizar

- **Contexto:** Formulario de agendamiento de horas médicas
- **Ejemplar:** `char hourForm.apellidoP[] <- Munoz`
- **Entrada:** `char hourForm.apellidoP[]`
- **Salida:** `char hourForm.nombre[]`
- **Descripción:** Dentro del formulario de agendamiento de horas médicas, se le solicitan al usuario una serie de datos, dentro de los cuales se encuentran el primer nombre (o nombre de pila) más los dos apellidos, los que posteriormente se concatenan en una única cadena de caracteres denominada `char hourForm.nombre[]` que permite mostrarla por pantalla y registrarla adecuadamente (sin saltos de línea) en el almacén de datos.

- **Contexto:** Formulario de búsqueda de horas médicas por RUN
- **Ejemplar:** `char searchTerm[] <- 12345678-9`
- **Entrada:** `char searchTerm[]`
- **Salida:** `FILE * flujoSearchHour` comparado con `char hourForm.run[]`
- **Descripción:** Dentro del formulario de búsqueda de horas médicas por RUN, se le solicita al usuario ingresar dicho dato respetando el formato que ahí se indica, el cual se almacena en la cadena `char searchTerm[]` y que posteriormente se compara con la cadena `char hourForm.run[]` devuelta por `FILE * flujoSearchHour`, entregando el(los) resultado(s) encontrado(s) dentro del almacén de datos del programa o desplegando solo un mensaje en caso contrario.

- **Contexto:** Formulario de eliminación de horas médicas
- **Ejemplar:** `char deletionEntry[] <- Poblete`
- **Entrada:** `char deletionEntry[]`
- **Salida:** `FILE * fileHoursLog` y `FILE * auxHoursLog`
- **Descripción:** Dentro del formulario de eliminación de horas médicas, se le solicita al usuario ingresar una entrada del registro correspondiente a la hora médica agendada que se desea eliminar del sistema, la que se almacena en la cadena `char searchTerm[]` y posteriormente se compara con `FILE * fileHoursLog` donde se extrae la información del almacén de datos y se copia en el archivo auxiliar `FILE * auxHoursLog`, menos la entrada que se definió para eliminar, luego la información regresa al almacén de datos y se elimina el archivo auxiliar.

Diseño de solución

Posibles estructuras a utilizar

- **Enumeradores:**
  - Este primer enumerador tiene contemplado el poder agrupar las especialidades médicas (en orden alfabético) y estará compuesta por cardiólogo (inicializado en 1), dermatólogo, gastroenterólogo, odontólogo, oftalmólogo, otorrinolaringólogo, psicólogo, psiquiatra, entre otras
  - También se contempla otro enumerador que dará la posibilidad al usuario de elegir una opción según el número que le corresponde. Estará compuesta de “Buscar horas médicas” (inicializada en 1), “Administrar horas médicas”, “Revisar registros de horas médicas” y “Salir del programa”. Se espera poder incluirla como parte de una biblioteca con el objetivo de que en el código principal no exista una abundante cantidad de `printf` y `scanf`.
  - Se planea la creación de un enumerador que permitirá ordenar a los profesionales de la salud dentro de una institución asignándoles un número que posteriormente servirá para una estructura que tendrá como función el desplegar los datos del especialista al momento de buscar una cita médica agendada previamente o cuando se deseé agendar una nueva hora dentro del formulario correspondiente.
  - En la misma idea de mantener más limpio el código fuente principal, se tiene pensado crear un enumerador que pueda ser utilizado a través de una función para desplegar los mensajes en caso de error y para entregar un aviso de fin de ejecución del programa.

- **Estructuras:**
    - Se contempla crear estructura que pueda ser utilizada como formulario para agendar nuevas horas, la cual estaría compuesta por el nombre y los dos apellidos del paciente, su RUN y digito verificador, la especialidad para la que se agendará una hora (aquí, de acuerdo a la opción elegida (por ejemplo: cardiólogo, dermatólogo, entre otros), se desplegará un listado de médicos con agenda disponible), y la fecha (separada en día, mes (en número) y año) y hora (separada en horas y minutos en formato 24 horas) con que quedará reservada la cita.
    - También se tiene contemplado crear una estructura se utilizará como formulario para buscar horas médicas agendadas previamente y estará compuesta por el nombre y los dos apellidos del paciente, su RUN y digito verificador, la fecha (separada en día, mes (en número) y año) y hora (separada en horas y minutos en formato 24 horas). Dependiendo de la opción que, elegida dentro del programa, se solicitarán solamente el nombre y los dos apellidos, el RUN y el DV, o la fecha y/o hora.
    - Se planea la creación de una estructura que agrupará los datos de un profesional de la salud, los que posteriormente serán desplegados por pantalla en el programa por medio de un enumerador en el contexto de los formularios de búsqueda y agendamiento de horas médicas.

Justificación de algoritmos y estructuras

- **Justificación de algoritmos:**
  - **`int samHead():`** Esta función permite desplegar los encabezados y cabeceras que tienen cada uno de los menús y opciones que proporciona el programa. Al igual que las funciones de más abajo, cumple el objetivo de mantener más limpio el código principal y permitir la adecuada modularización de todos los archivos fuente que componen el software.
  - **`int samOption():`** Esta función permite mostrar, propiamente tal, las opciones y los formularios de cada apartado del programa. Por lo general, esta función suele usarse debajo de `int samHead()`.
  - **`int samMessage():`** Esta función permite desplegar los mensajes por defecto en caso de error y para indicar el fin de la ejecución del programa cuando se sale de éste mediante la opción que se indique.
  - **`int medicalInfo():`** Esta función forma parte del formulario de agendamiento de nuevas citas médicas, permitiendo desplegar los horarios disponibles para agendar según la especialidad y el profesional de la salud respectivo indicado previamente. Dentro de esta función, también se realiza la comunicación con el almacén de datos para registrar las horas médicas ingresadas correctamente al sistema.
  - **`char searchTerm[]:`** Dentro del formulario de búsqueda de horas médicas, esta cadena de caracteres tiene como objetivo recibir un término que posteriormente se compara con los datos contenidos en la estructura `hourForm` que están dentro del almacén de datos del programa, devolviendo la(s) hora(s) médica(s) agendada(s) previamente en el sistema de acuerdo a lo ingresado por el usuario o un mensaje informativo en caso de no haber encontrado coincidencias.

- **`char deletionEntry[]`:** Dentro del formulario para eliminar horas médicas, esta cadena de caracteres tiene como objetivo recibir una entrada que posteriormente se compara con la información contenida en el almacén de datos del software, para luego realizar la operación de eliminación de la cita médica indicada.
- **`int cont`:** Esta variable se utilizará en algunos ciclos del programa para prevenir algunos problemas comunes dentro del lenguaje de programación C como, por ejemplo, que se muestre dos veces una cadena de caracteres ingresada dentro de un almacén de datos, entre otros inconvenientes.

- **Justificación de estructuras y enumeradores:**
  - **`hourForm_t <- hourForm`:** Esta estructura se encargará de ir agrupando cada uno de los datos que se ingresen a través del formulario de agendamiento de horas médicas, los cuales posteriormente serán registrados dentro del almacén de datos del programa. Consta de 11 cadenas de caracteres que recogen datos como RUN, nombre y apellidos del paciente, fecha, hora, especialidad y datos del médico, entre otros.
  - **`principalOptions_enum <- principalOptions`:** Este enumerador se encarga de agrupar las opciones del menú principal del programa. Entrega las opciones de buscar, administrar y revisar registros de horas médicas agendadas previamente, además de la opción para salir del programa.
  - **`searchOptions_enum <- searchOptions`:** Este enumerador se encarga de agrupar las opciones del menú de búsqueda de horas médicas del programa. Entrega las opciones de buscar por RUN, nombre, apellido paterno, apellido materno y fecha, además de las opciones para volver al menú principal o salir del programa.

- **`adminOptions_enum <- adminOptions`:** Este enumerador se encarga de agrupar las opciones del menú de administración de horas médicas del programa. Entrega las opciones de agendar y eliminar una hora médica del sistema, además de las opciones para volver al menú principal o salir del programa.
- **`especialidad_enum <- especialidad`:** Este enumerador se encarga de agrupar las diversas especialidades médicas que pueden recibir agendamiento de horas dentro de la institución de salud. Se compone de cardiólogo, dermatólogo, gastroenterólogo, geriatra, ginecólogo, kinesiólogo, medico general, neurólogo, nutriólogo, odontólogo, entre varias otras.
- **`medicalHour_enum <- medicalHour`:** Este enumerador se encarga de agrupar las horas médicas que están disponibles para ser agendadas en el sistema. La información se despliega durante el formulario de agendamiento de horas médicas.
- **`readyAddHour_enum <- readyAddHour`:** Este enumerador se encarga de preguntar al usuario si desea agendar una nueva hora médica tras concretar un agendamiento anterior dentro del formulario respectivo. Entrega las opciones de agendar nueva hora médica y eliminar hora médica, además de volver al menú principal y salir del programa.
- **`messageType_enum <- messageType`:** Este enumerador se encarga de agrupar los mensajes de error en caso de que el usuario digite una opción errónea, y de fin de ejecución del programa en el caso que se seleccione la opción correspondiente en alguno de los menús.
- **`yesNo_enum <- yesNo`:** Este enumerador se encarga de agrupar las opciones de sí o no para algunas instancias del programa donde solo se requieren de alguna de estas dos respuestas.

Diagrama de nodos de estructuras relevantes

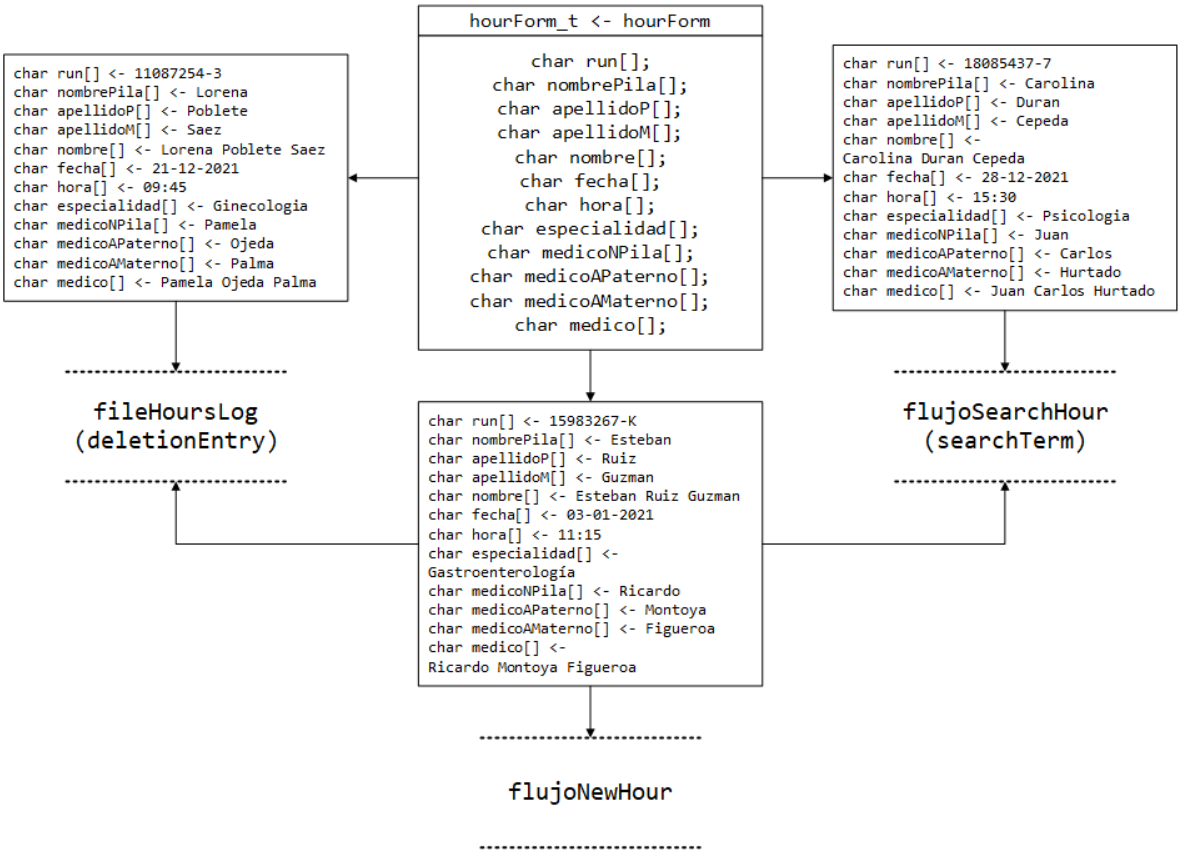

Figura 1: Diagrama de nodos representando estructura relevante del código correspondiente al formulario de agendamiento de horas médicas donde se incluyen casos de prueba con su respectiva aplicación en uso de almacén de datos bajo diferentes flujos.

Diagrama de flujo de datos de algoritmos relevantes

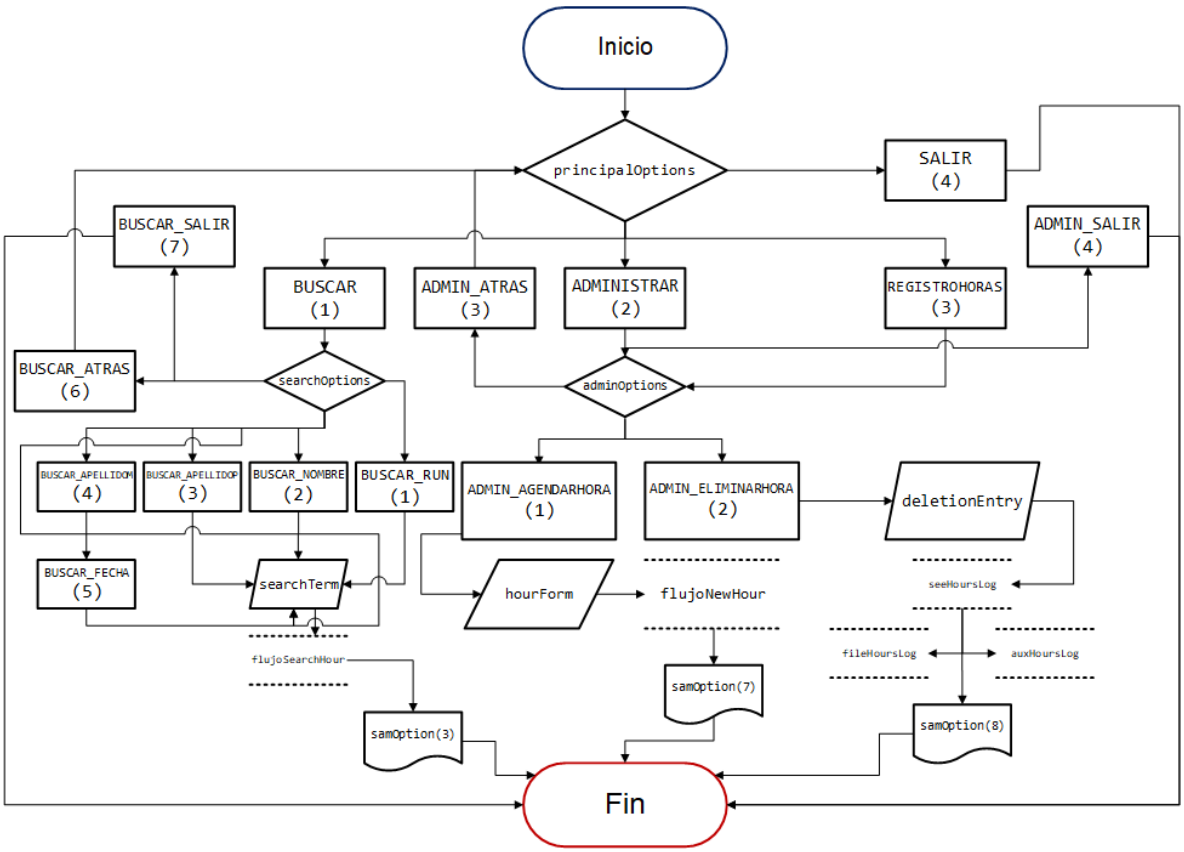

Figura 2: Diagrama de flujo de datos con los algoritmos relevantes que controlan los menús que permiten acceder a las principales funcionalidades del programa y los formularios que van registrando o localizando las horas médicas agendadas en el almacén de datos.

Funcionalidades de la aplicación

El programa cuenta con diversas funcionalidades que permiten realizar una eficaz gestión de la administración de las horas médicas dentro de una institución de salud.

- **Agendamiento de horas médicas**

  La principal funcionalidad, y la razón de ser de S. A. M., es permitir ingresar de forma rápida y segura las horas médicas que requieren ser agendadas dentro de un recinto hospitalario, clínico o un consultorio. Lo anterior, se logra gracias al almacén de datos que dispone el programa para ir registrando todos los agendamientos que se van recibiendo sin que éstos se pierdan una vez cerrado el programa o la sesión del respectivo sistema operativo donde está funcionando.

- **Búsqueda de horas médicas**

  El programa cuenta con un buscador que permite localizar horas médicas que han sido registradas en el sistema, para ello se da la opción de buscar por RUN, primer nombre (o nombre de pila), apellido paterno, apellido materno o fecha. Posterior a lo anterior, se solicita al usuario que ingrese el dato respectivo y en caso de existir coincidencias con lo contenido en el almacén de datos, mostrará en pantalla la información, en caso contrario, se mostrará un mensaje de que no se encontraron coincidencias.

- **Eliminación de horas médicas**

  También el software dispone de una opción que permite eliminar una hora médica que ha sido registrada en el sistema, para lo anterior primero se muestra la tabla completa con las horas registradas en el sistema y luego se solicita al usuario ingresar la entrada correspondiente al registro, tras lo cual se elimina del almacén de datos del programa.

Pormenores en el diseño y construcción de la aplicación

Al principio durante la etapa de definición del problema, hubo algunas dudas con respecto a la idea central del programa y cómo éste podría ayudar a las personas potencialmente interesadas en él. Luego de una reunión coordinada previamente con el docente, se pudieron clarificar de mejor manera estos aspectos, lo que posibilitó seguir adelante con el propósito original del proyecto.

Durante la etapa de elaboración, aún sin tener claro cómo se iban a programar las funciones centrales del programa (búsqueda, agendamiento y eliminación de horas médicas dentro del sistema), se partió con el desarrollo del código fuente en lenguaje de programación C, el cual desde un principio se fue modularizando y dividiendo en varios archivos de bibliotecas (.h) con sus respectivas implementaciones (.c), junto con la creación de enumeradores para los menús y estructuras para los potenciales formularios, con el objetivo de lograr un mayor avance antes de pasar a trabajar en la construcción de las funcionalidades centrales del software.

En la etapa de construcción, se comenzó a trabajar en la parte más compleja del desarrollo del programa que consistía en la puesta en marcha de sus principales funcionalidades relacionadas con la administración de horas médicas junto con la implementación del almacén de datos que posibilitaría ir registrando las citas ingresadas al sistema sin que éstas se pierdan cuando se cierra el software. Se partió por construir el sistema que permitiría el agendamiento de horas médicas, lo cual no significó mayores problemas, salvo cuestiones de orden en el registro de los datos que se fueron resolviendo haciendo varias pruebas con la función `strtok` (que elimina los saltos de línea de las cadenas). Lo más laborioso fue la implementación de las funcionalidades de búsqueda y eliminación, para las cuales se debió recurrir a recursos externos (vídeos de YouTube (LearnWtutorials, 2015) (González, 2016) (González, 2016), foros de programación (Afia & Sh, 2019), entre otros). También debieron resolverse algunos *bugs* dentro de la ejecución del programa a través de la terminal.

Conclusiones

A pesar de todas las dificultades que surgieron durante la última etapa de desarrollo del proyecto, fue posible poner a disposición del usuario final un programa de calidad completamente funcional que cumpliera con los principales lineamientos que se habían planteado desde un principio (inmediatez y confiabilidad), además de cumplir casi todas las expectativas que como grupo de trabajo nos habíamos propuesto para sacar adelante este software.

Algunas funcionalidades como las de modificar o inhabilitar una hora médica ya agendada previamente no pudieron llevarse a cabo debido a limitaciones propias del lenguaje de programación C y del almacén de datos ya implementado (en formato .txt). Se había pensado en algún momento cambiar el tipo de almacén de datos por un archivo en formato .csv (datos separados por comas y saltos de línea) que tal vez hubiera posibilitado lo anterior, pero ello habría implicado retroceder en los avances ya realizados, por lo que ya no resultaba viable.

Como grupo de trabajo, la experiencia durante el desarrollo del proyecto fue positiva, ya que cada una de las responsabilidades se fueron repartiendo en las respectivas etapas. Se organizaron reuniones periódicas mediante Discord para conversar los avances e ir trabajando en conjunto en la elaboración de los documentos del proyecto junto a la construcción del código fuente del programa. Por otro lado, el registro de las tareas con sus responsables se llevó a cabo a través de Trello (enlace indicado al final de esta página).

Finalmente, se agradece el apoyo del docente del curso, quien nos orientó en algunas de las fases de elaboración y construcción del programa.

Nombre del espacio de trabajo	IN1045C Sistema de Agendamiento Médico
Enlace del espacio de trabajo	Ingresar al espacio de trabajo en Trello

Referencias

Afia, & Sh, F. (22 de Marzo de 2019). *Delete a specific line from a text file using C*. Recuperado el 26 de Diciembre de 2021, de Tutorials Panel: http://www.tutorialspanel.com/delete-a-specific-line-from-a-text-file-using-c/index.htm

González, L. (25 de Junio de 2016). *Agenda telefónica (1/4) | Programación en C*. Recuperado el 23 de Diciembre de 2021, de YouTube: https://www.youtube.com/watch?v=6dHW5cnjVzw&ab_channel=LuisAlbertoGonzalezAlvarez

González, L. (27 de Junio de 2016). *Agenda telefónica (2/4) | Programación en C*. Recuperado el 23 de Diciembre de 2021, de YouTube: https://www.youtube.com/watch?v=RFgCRX68Nbk

LearnWtutorials. (8 de Enero de 2015). *Tutorial 22 del lenguaje C - Leer y buscar en un archivo*. Recuperado el 23 de Diciembre de 2021, de YouTube: https://www.youtube.com/watch?v=l2mtY858YPI&ab_channel=LearnWtutorials