Minutas del proyecto

Limitaciones del programa:

Debido a las limitaciones de tiempo, se ha priorizado la resolución de los enunciados del proyecto, perjudicando así un poco tanto la estética como diseño de la interfaz gráfica.

La interfaz gráfica no se puede agrandar, ya que no mantiene su relación de aspecto en el modo pantalla completa, siempre medirá lo mismo en tamaño, desde su ejecución.

Se pueden sobrescribir pacientes en la lista con el mismo número de documento, pero cuando se eliminen, solo eliminarán a uno solo, debido a la naturaleza del mapeo, y de insertar personas con su número de DNI como clave. Se asume que no existan pacientes con mismo número de documento. Al ingresar otro paciente en el sistema con mismo DNI, y al eliminarlos a los 2, uno de ellos sobrescribirá al anterior en el mapeo de pacientes históricos. (Ver: alcances de las soluciones)

los campos de texto de inserción de numero de documento se han editado de manera rudimentaria para que solo acepte la inserción de caracteres numéricos (0, 1..., 9) o bien "BACKSPACE" para borrar su contenido (Mas información en el apartado de alcances de las soluciones). Cuando se insertan caracteres de otro tipo, el campo de texto aparecerá como "deshabilitado", sin embargo, se pueden seguir ingresando caracteres numéricos en este, el cual habilitará automáticamente el campo en cuestión.

Debido a las limitaciones con la clase "Integer" de java (almacenamiento de 32bits), al ingresar un número mayor a 2147483648, producirá una excepción desde java. Por lo que en los campos numéricos a rellenar con información de numero de documento, se requiere que el cliente no ingrese un número mayor al "cap" de la clase Integer (2147483648). Se asume que el número de

documento de un paciente es usualmente de menos de 8 cifras, lo cual no debería interferir con este límite.

En el campo de nombre y apellido se pueden ingresar caracteres alfanuméricos, se asume que el cliente va a ingresar nombres y apellidos con caracteres únicamente alfabéticos. Sin embargo, esto no afecta al funcionamiento del programa.

Alcances de las soluciones:

Se han implementado la pila y cola con enlaces, la lista con enlace doble y sin centinelas, el mapeo con hash abierto y una cola con prioridad utilizando un heap, sin embargo, a la hora de la resolución global del proyecto, la lista es la única estructura que no se ha llegado a utilizar en ninguna solución.

Utilizamos la implementación de heap para la cola con prioridad, teniendo en cuenta de que esta es la implementación cuyo tiempo de ejecución para cada una de las operaciones del TDA, es en general, más rápida, teniendo un orden de tiempo de ejecución logarítmico para los métodos insert() y removeMin(), y orden constante para el método min(), en contraposición al orden de la implementación con lista ordenada, el cual tiene un min() y removeMin() de tiempo constante, pero un insert() de orden lineal. En una lista sin ordenar, el insert() es de orden constante, pero queda descartado ya que min() y removeMin() son de orden lineal.

Solo 2 estructuras han sido utilizadas desde siempre como atributo a la lógica del programa, el mapeo y la cola con prioridad, el resto de las estructuras creadas se crearán y se inicializarán de manera auxiliar dentro de cada uno de los métodos de resolución de los incisos del proyecto.

Para la resolución de los problemas, y para conectar soluciones con la interfaz gráfica, desarrollamos la clase "Logica", la cual mantiene todos los métodos empleados a la resolución de cada uno de los enunciados del proyecto. Sin

embargo, hay algunos aspectos de la solución que se han resuelto dentro de la interfaz gráfica, más que nada relacionados con la obtención de datos de los campos de texto, muestra de datos en pantalla, funcionamiento de los botones (oyentes) y muestra de mensajes de notificación en pantalla (faltan completar datos, búsqueda de pacientes, etc.).

Para facilitar la resolución en la lógica del programa, abstrajimos a los pacientes en una clase "Persona", la cual almacena datos importantes de los pacientes, como el DNI y su nombre completo. Dicho objeto de clase persona, es el valor a insertar tanto en la cola con prioridad de "Pacientes" o el mapeo de "historialPacientes".

En la lógica se inicializa una cola con prioridad cuya prioridad es el grupo de riesgo del paciente, y el valor a almacenar es un objeto de tipo Persona, y un mapeo, cuyas claves son el número de documento de una persona, y el valor a almacenar es un objeto de tipo Persona.

Para calcular la prioridad del paciente, según grupo de riesgo, se modificó el comparador de la cola con prioridad, para que priorice las claves mayores por sobre las menores, permitiendo así que no se deba manipular el valor de grupo de riesgo que nos dé en cualquier momento la inserción de pacientes.

Para mostrar el paciente más riesgoso, lo mostramos en un campo de texto deshabilitado. Internamente, capta al paciente más riesgoso utilizando simplemente el método “min()” propuesto por la cola con prioridad.

Para mostrar todos los pacientes en pantalla, de mayor a menor prioridad, tuvimos que remover a todos los pacientes de la cola con prioridad, y almacenarlos en una cola con prioridad auxiliar, y una cola de caracteres en caso del nombre y apellido de los pacientes. Esta última luego se irá concatenando para mostrar en pantalla todos los pacientes. Por último, todos los pacientes de la cola con prioridad auxiliar, volverán a ingresar a la cola con prioridad de la lógica. Esta solución tal vez sea un poco lenta a la hora de su tiempo de ejecución, ya que la cola con prioridad presenta limitaciones para

poder obtener todos sus datos en orden, ya que hay que removerlos y luego volver a insertarlos, recorriendo así toda la estructura al menos 2 veces.

De manera similar, para mostrar los pacientes de menor a mayor prioridad, en lugar de ingresar los nombres de los pacientes en una cola, los ingresamos en una pila, la cual, ingresa y luego brinda los pacientes en un orden inverso al de "llegada" a esta estructura de datos, mostrando así una lista inversa a la que nos puede mostrar la cola. El procedimiento de obtención de datos de la cola con prioridad, va a seguir siendo el mismo de remoción e inserción en otra cola con prioridad auxiliar, y viceversa.

Para eliminar paciente, la solución es aún más engorrosa, al tener que remover todos los pacientes de la cola con prioridad, chequear según su número de documento si coinciden con el que se ingresó en la interfaz, y si este coincide, no será ingresado a la cola con prioridad auxiliar, sino que será ingresado al mapeo de historialPacientes, donde será guardado en el historial de pacientes ya eliminados de la base de datos. Nuevamente se repite el proceso de reinserción de pacientes de la cola con prioridad auxiliar a la cola de la clase Logica.

Para consultar en el registro de pacientes ya atendidos, se obtendrá un número de DNI y se comparará con alguna clave del mapeo que almacena a los pacientes en el historial, se realizará simplemente con el metodo "get(K)", que obtendrá a la persona, y luego le pedimos el nombre de ésta, en un String, para mostrarlo en pantalla.

Problemas en el desarrollo del proyecto:

Debido a la nula capacidad de interactuar en persona entre compañeros de proyecto, no solo por cuarentena, sino que, por puro desconocimiento mutuo fuera del ambiente de la cursada, se tuvo que coordinar de manera muy fija para utilizar plataformas de llamada online, para resolver el proyecto. Hubo complicaciones grandes relacionadas con el tema de comunicación y puesta a

punto del proyecto. Fue difícil asignar tareas entre cada uno para este, y respetar las ideas de trabajo.