**SISTEMA DE GESTIÓN DE EXÁMENES DE SANGRE**

Los exámenes de sangre son un procedimiento con el que se mide el estado de diferentes sistemas del cuerpo humano de forma indirecta. Esto debido a que el sistema circulatorio actúa como un medio de transporte para los diferentes compuestos químicos que son requeridos de con alguna funcionalidad en específica en el cuerpo humano. Permitiendo entonces que, por medio de la medición de estos compuestos químicos, y de la relación con su funcionalidad, sea posible diagnosticar el estado del cuerpo humano.

Se pide entonces implementar un sistema para la gestión de 5 diferentes tipos de exámenes de sangre, donde estos son: recuento sanguíneo completo, panel tiroideo completo, panel metabólico integral o perfil bioquímico completo, perfil o panel lipídico y ácido úrico.

**Funcionalidades del sistema:**

* Debe almacenar los exámenes según el paciente a quien se le realiza, permitiendo establecer un historial de todos los exámenes que se ha realizado el paciente.
* La generación del historial del paciente y el almacenamiento de la información de este debe suceder dentro del sistema.
* La consulta de algún examen en específico se debe realizar por medio del ID específico de un examen, si se desea la realizar la consulta de todos los exámenes asociados a un paciente se deber realizar por medio del nombre de identidad de este.

**Simplificaciones:**

* Para simplificar el acercamiento a la generación de los diferentes exámenes de sangre, la diferencia que radicará en cada uno de estos es el tipo de compuesto químico que se mide.
* La información en general que determina un examen de sangre en general es: ID del examen, fecha de realización, método en que se realiza, nombre del médico que pide el examen y compuesto químico medido. La información asociada a la persona que se realiza el examen puede omitirse, debido a que la información de este se encuentra en el sistema, y dentro de esta información están los diferentes exámenes realizados.

**Solución**

Para la resolución del problema se separó este en tres secciones. Las primeras dos secciones son las relacionadas con el desarrollo de las clases que se encargarán de capturar la información relacionada con los exámenes. Debido a que existen ciertos campos que se comparten en todos los exámenes, se decidió entonces generar una clase de donde heredan todos estos. Esta clase se llama examenGeneral, y su estructura se muestra en la columna izquierda de la próxima hoja.

En la columna de la derecha se encuentra la estructura de los exámenes en específico. Lo único que diferencia cada clase generada son los atributos que capturan la cantidad del compuesto que se mide en el determinado examen. Por ejemplo, en el examen de perfil Bioquímico completo se mide (Simplificando) la cantidad de glucosa, bilirrubina y creatinina; por lo tanto, la clase generada tendrá tres atributos, más los heredados del examen general. Además, se implementa un método especial \_\_str\_\_ el cual se encarga de especificarle a Python qué forma tendrá el objeto instanciado cuando se convierte en un string. Este método debe retornar por obligación un string.

Algo interesante de mencionar durante la implementación del método \_\_str\_\_ es el uso de lo que se llama f-strings; estos son una forma de darle formato a un string durante la declaración de este. Para hacer uso de estos se debe anteponer la f antes de las comillas, luego por medio del uso de las llaves se ingresa la variable (o expresión a evaluar) que se desea que vaya en esa posición.





En el código anterior se encuentran los atributos que, junto a los métodos de la clase, permiten establecer la funcionalidad del sistema. A continuación, se mostrará cuál es la intención detrás de cada atributo.

|  |  |
| --- | --- |
| self.\_\_baseDeDatos | Diccionario que se encarga de almacenar los datos ingresados dentro del sistema. Cada llave del diccionario es un usuario (Su nombre), mientras que el contenido hacia el que este apunta es una lista de objetos, donde cada uno de estos es uno de los exámenes disponibles dentro del sistema |
| self.\_\_objetivo | Variable en la que se almacena el nombre del usuario (la llave del diccionario) sobre el que se desee realizar alguna tarea, por ejemplo, ingresar un nuevo examen. |
| self.\_\_estadoMenu | Variable que ayuda a la gestión de los menús derivados del más externo (primer menú) que ofrece el sistema. |
| self.\_\_estado | Variable que ayuda a la gestión del menú más externo (primer menú) que ofrece el sistema. |
| self.\_\_examen | Variable que captura algunos de los tipos de exámenes disponibles. |
| self.\_\_msj1 | Menú más general del sistema. Permite elegir entre: Ingresar examen, ver el/los exámenes de un paciente o salir del sistema. |
| self.\_\_msj2 | Menú derivado de la opción de ingresar examen; permite escoger qué tipo de examen se va a ingresar. |
| self.\_\_msj3 | Menú derivado de la opción de ver el/los exámenes de un paciente. |

Para explicar la funcionalidad del sistema se mostrará una línea del tiempo donde se resalta y explica qué está sucediendo. Esto se hará con las diferentes opciones del menú general:

**Ingresa examen:**

****

Cuando se instancia el sistema, se llama el método iniciar para que este comience su funcionamiento. En este método se genera un while infinito, que es el que permitirá el funcionamiento continuo de la aplicación. Lo primero que hace este es entonces ingresar en el sector del código resaltado con amarillo, ya que las variables se encuentran inicializadas. En este bloque de código se vuelve a su valor original el atributo estadoMenu y luego se muestra en pantalla el msj1 para luego almacenar la opción elegida por el usuario en la variable estado.

Como lo que se mostrará inicialmente es el ingreso de un examen, la variable estado toma el valor de 1; luego sale de la estructura del condicional y como se encuentra en el While infinito, se vuelve a ejecutar las sentencias que hay dentro de este.



Estando dentro de las sentencias señaladas en el código anterior, se llama el método ingresarExamen de la clase sistema. En esta se hará gestión del menú derivado del general, ya que se debe escoger dentro de los diferentes exámenes que se pueden escoger. Lo primero que se ejecuta en este es el guardar el tipo de examen que se ingresará, esta opción se almacena en el atributo estadoMenu. Luego, se verifica si se escogió la opción de salir del sistema, si no, se pide al usuario que ingrese el nombre del paciente al que se le ingresará el examen.



En el código anterior de la izquierda se muestra lo que se evalúa luego de ingresar la opción que se almacena en el estadoMenu. Si se desea salir del sistema basta con solo hacer que el atributo estado vuelva a su valor inicial 0. Luego, no entrará en ninguna de las otras sentencias que están en el método ingresarExamen; ya que inicialmente había entrado en el condicional donde el estadoMenu se encuentra en 0, por lo que los otros elif no son evaluados. Luego, este regresa a la sección del método inicial que está resaltado en el código de la derecha, debido a que el atributo examen sigue estando en su valor de None, por lo que se encuentra en este no es evaluado. De esta forma, vuelve a ejecutarse las sentencias que están en el While infinito, y como estado volvió a su valor de 0, se regresa al inicio de la funcionalidad del sistema.

Ahora bien, si el valor que se ingreso en estadoMenu es otro, o sea ingresar algún tipo de examen, las sentencias que se ejecutan son las señaladas en azul del código de la izquierda. En estas se realizan diferentes cosas: Inicialmente se le pide al usuario que ingrese el nombre del paciente al que se le ingresará el examen, este lo almacena en el atributo objetivo, el cual será usado como llave en el atributo baseDeDatos. Luego se verifica si esta llave (paciente) se encuentra en la base de datos, si no, se genera un nuevo campo en el diccionario donde la llave es el nombre del paciente y su contenido es una lista vacía. Luego sucede lo mismo que en el párrafo anterior, no se ejecutan los elif’s y tampoco se ejecuta la parte señalada del código de la derecha, ya que el atributo examen sigue teniendo su valor en None. Sin embargo, debido a que el atributo estado sigue estando en 1, cuando se evalúa de nuevo el While infinito este vuelve a entrar en el método de ingresarExamen.



Estando en el método ingresarExamen se evalúan los diferentes elif’s, en este caso se mostrará sólo que pasa con uno de los tipos de exámenes, ya que el ingreso de los diferentes tipos de exámenes solo varía en el tipo de compuesto que se asigna.

Cuando se evalúa alguno de los elif’s se llama alguno de los métodos examenTipo\_, en el código de la derecha se señala qué se comparte por todos los métodos: inicialmente se instancia el objeto que contendrá la información asociada a un examen (Este objeto queda almacenado en el atributo examen de la clase sistema), luego se llama el método examenGeneral, el cual se encarga de asignar los datos generales que identifican a un examen. Luego, se asignan los otros valores que identifican a un examen en específico. En el código de la derecha se muestra cómo se hizo con el examen de recuento sanguíneo: primero se instancia el objeto perteneciente a la clase recuentoSanguineoCompleto, luego se llama el método examenGeneral para asignar los atributos correspondientes y por último se asignan los atributos específicos de este tipo de examen.

Luego de tener el definido por completo el objeto que captura la información de un examen, lo siguiente es almacenarlo en la base de datos. Para esto se realiza lo siguiente:



Cuando se termina de ejecutar las sentencias dentro del método examenTipo\_ no se evalúan los otros elif’s y sigue entonces ejecutándose las sentencias que se señalan en el código anterior. En este caso el atributo examen sí tiene un objeto asociado, por lo que es diferente de None y se ejecuta lo que está dentro de ese condicional. Inicialmente se vuelve la variable estadoMenu en 0 para su futuro uso, luego se almacena el objeto con la información del examen en la base de datos, esto se hace usando el atributo objetivo, el cual es la llave asociada al paciente y el método append, el cual anexa al final de la lista a la que apunta la llave en el diccionario baseDeDatos. Luego se vuelve de nuevo a None el atributo examen (Esto porque este condicional se evalúa durante muchas etapas de funcionamiento del sistema como se observa anteriormente) y se le dice a usuario que el examen ya fue ingresado. Con todos los atributos de vuelta a su estado inicial y el atributo estado estando en 1, se regresa al comienzo de la funcionalidad de este apartado.

**Ver examen:**

|  |  |
| --- | --- |
|  |  |

Cuando el atributo estado toma el valor de 2 se llama al método verExamen, en este lo primero que se realiza es lo señalado en el código de la derecha. Se muestra el msj3 para decidir si se observará un examen en específico o todos los exámenes asociados a un paciente o se saldrá del sistema. La opción ingresada por el usuario es almacenada en el atributo estadoMenu. Luego, los otros condicionales elif’s no se evalúan y se sigue ejecutando el While infinito. Si el usuario ingresó la opción de salir de este menú, el atributo estado regresa al valor de 0 y cuando se vuelve a ejecutar las sentencias dentro del While infinito, se ingresa al condicional inicial donde se muestra al usuario el msj1.

Si la opción ingresada en el estadoMenu es otra, se ejecutarán algunas de las dos opciones que se resaltan en azul en el código de la derecha. A continuación, se mostrará qué hacen estas dos opciones:

|  |  |
| --- | --- |
|  |  |

Cuando se ingresa la opción 1 el usuario decidió ver un examen en específico, para esto debe ingresar el id del examen. El sistema entonces llama al método buscarExamen\_Id, cuya funcionalidad se muestra en el código de la derecha. Este método implementa dos for’s anidados, los cuales recorren toda la base de datos en busca del id suminastrado por el usuario. Si el id suministrado coincide con el de algún examen, este examen se asocia al atributo examen de la clase y el método retorna True, si no False.

Luego, en función de lo retornado en el anterior apartado se imprime en pantalla el examen (El cual ejecuta el método especial \_\_str\_\_ del objeto asociado) y luego se reinicializan las variables involucradas, o se muestra en pantalla que el Id suministrado no coincide con ningún examen. Al finalizar ambos caminos el atributo estadoMenu regresa al valor de 0, por lo que se regresa al estado del sistema donde se muestra el msj3 en pantalla.

Con respecto a la otra opción la lógica que sucede es muy simple: Se ingresa el nombre del paciente, luego se verifica si este está en la base de datos. Si está, se recorre la lista a la que apunta este en el atributo baseDeDatos, y cada examen se imprime, luego se reinicializan las variables para regresar a la sección de código donde se muestra el msj3. Si el paciente no está registrado, se muestra en pantalla que no se encuentra registrado y se vuelve el atributo estadoMenu en 0, para regresar a la sección del msj3.

**Salida del Sistema**

****

Por último, si se decide salir del sistema se le asigna el valor de 3 al atributo estado, en las sentencias dentro de este condicional se llama a la palabra clave break, la cual hace que rompa la ejecución del While infinito.

**Aclaraciones finales:** algunas secciones de los códigos presentados no están completamente validados, por lo que no se muestra qué pasa si el usuario ingresa un valor que no se espera, esto puede ser corregido fácilmente. Además, varias de las lógicas presentadas podrían ser implementadas de varias formas.