Entrega SIA FINAL

Tema: Gestión de Agendas y Reuniones

Integrantes: Bastian Carrasco, Matias Fuentes y Jan Houter

1.1 Realizar un análisis de los datos a utilizar y principales funcionalidades a implementar que dan sentido a la realización del proyecto

Para el proyecto nuestro grupo seleccionó el tema de gestión de agenda y reuniones. En el cual se propone diseñar una aplicación para el tema escogido. Busca ser una mejora en el proceso de planificar y organizar el día a día, ya que las gestiones de agendas actuales son de un difícil manejo para los usuarios y no han sido modificadas a lo largo del tiempo. El objetivo de la aplicación es que el usuario pueda diseñar, organizar y planificar sus reuniones o actividades en una agenda de manera agradable y eficiente.

Para lograr lo anterior mencionado, se proponen las siguientes clases:

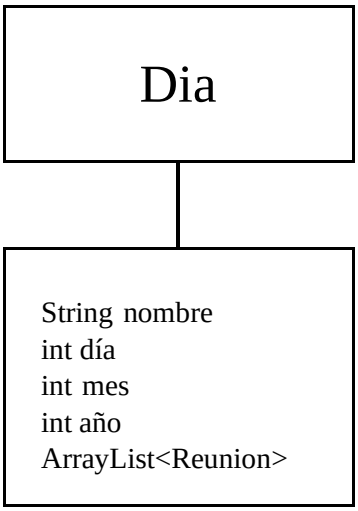
- AgendayReuniones, clase principal de la aplicación donde se encuentra el menú.
- Día, clase que contendrá las fechas de las actividades.
- Reuniones, el usuario podrá ver las actividades programadas.
- NewJFrame, clase principal de la aplicación que contiene la interfaz gráfica.
- SoloLetras
- SoloNumeros

Funcionalidades que dan sentido al proyecto:

- mostrar, llama al método data_toda.
- data_toda, muestra los datos de la actividad/es de la persona.
- seguridad, hace validaciones correspondientes.
- Llenr_con_datos, llena el hashmap con csv.
- Errores, revisa si hay caracteres ajenos a los datos.
- mostrarDatos(), muestra los datos de la primera anidación.
- mostrarDatos(String), muestra los datos de la segunda anidación.
- btn_cambia1ActionPerformed, cambia los datos de la tabla 1.
- btnelimina2ActionPerformed, elimina datos de la tabla 2.
- btn_muestraActionPerformed, muestra los datos de la tabla 1.
- btn_cambia2ActionPerformed, cambia los datos de la tabla 2.
- btneliminaActionPerformed, elimina datos de la tabla 1.
- btnguardaActionPerformed, guarda nuevo dato.
- CSVActionPerformed, alerta que se creó un CSV.
- datosVariosActionPerformed, función propia.
- original1ActionPerformed, crea un manual.

(Tabla 1, se refiere a nombre)

1.2 Diseño conceptual de clases del Dominio y su código en Java



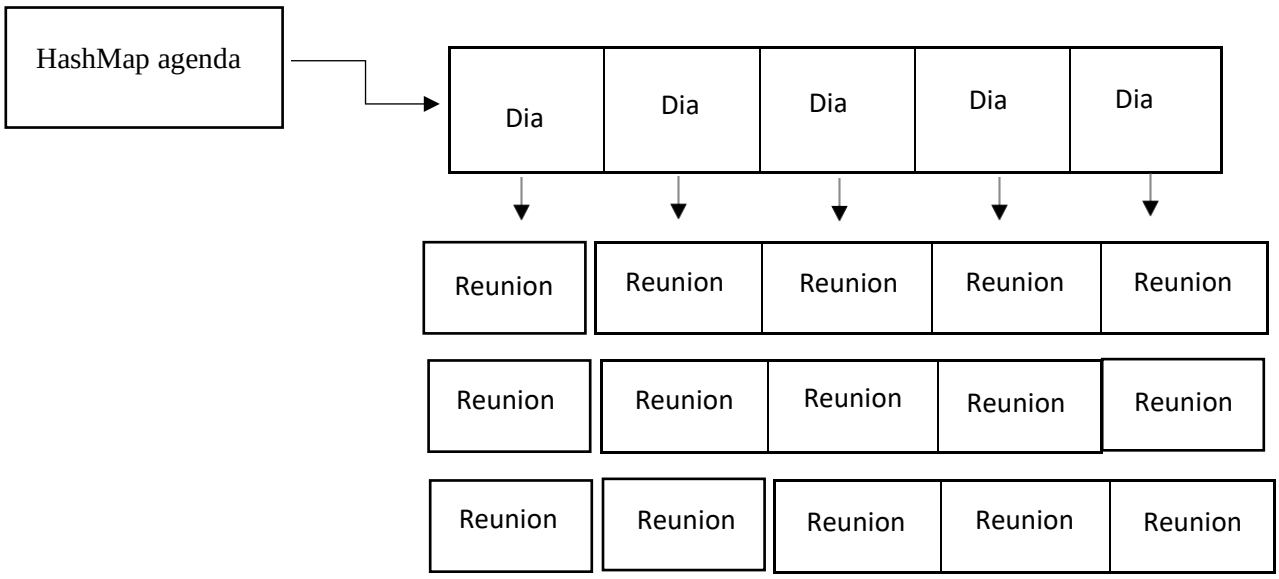
1.3 Todos los atributos de todas las clases deben ser privados y poseer sus respectivos métodos de lectura y escritura (getter() / setter())

Se consideraron las clases Día y Reuniones como se pueden ver en los archivos java correspondientes se tienen los atributos en privado junto con su correspondiente getter y setter.
En el archivo Dia.java podemos ver entre las líneas 16 y 20 como los atributos están en privado y entre las líneas 31 y 79 los métodos getter y setter correspondiente.
En el archivo Reuniones.java podemos ver entre las líneas 14 y 18 como los atributos están en privado y entre las líneas 29 y 67 los métodos getter y setter correspondiente.

1.4 Se deben incluir datos iniciales dentro del código

Los datos iniciales se encuentran dentro de la clase AgendayReuniones, en el main. Existe un método llamado llener_con_datos y se puede visualizar entre las líneas 198 y 244.

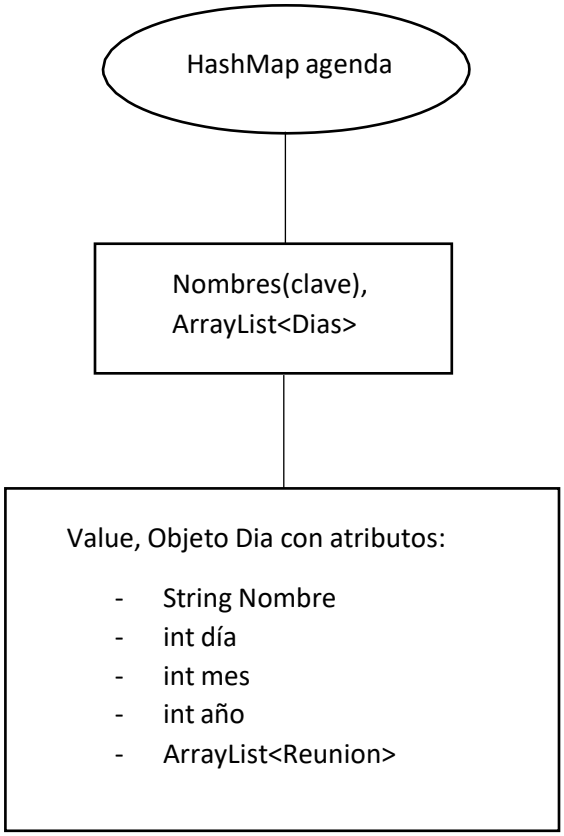
1.5 Diseño conceptual y codificación de 2 colecciones de objetos, con la 2ª colección anidada como muestra la figura. Las colecciones pueden ser implementadas mediante arreglos o clases del Java Collections Framework (JCF)



1.6 Diseño conceptual y codificación de 2 clases que utilicen sobrecarga de métodos (no de constructores)

En la clase Días entre las líneas 104 y 114 se encuentra el método seguridad, el cual valida que el día y mes de la actividad ingresada se encuentren dentro de los parámetros determinados. Y a su vez se encuentra seguridad, como sobrecarga de método el cual valida que el año de la actividad se encuentre valido.
Y en la clase Reunión entre las líneas 85 y 96 se encuentra el método seguridad, el cual valida que la hora, minuto inicial como final se encuentren dentro de los parámetros determinados. Y a su vez se encuentra seguridad, como sobrecarga de método el cual valida que la hora final no sea menor que la hora inicial.

1.7 Diseño conceptual y codificación de al menos 1 clase mapa del Java Collections Framework



Como se aprecia en la figura, nuestro HashMap agenda, trabaja con un objeto value Dia con atributos propios y es identificado a partir de la clave nombre.

1.8 Se debe hacer un menú para el Sistema donde ofrezca las funcionalidades de: 1) Inserción Manual / agregar elemento y 2) Mostrar por pantalla listado de elementos. Esto para la 2ª colección de objetos (colección anidada) del SIA1.5

Al momento de ejecutar se puede apreciar el menú con las funcionalidades solicitadas.
En la clase AgendayReuniones, en el método crearMenu, se imprime por pantalla el menú. Y entre las líneas 186 y 195 se puede apreciar como el menú fue programado para su funcionamiento.

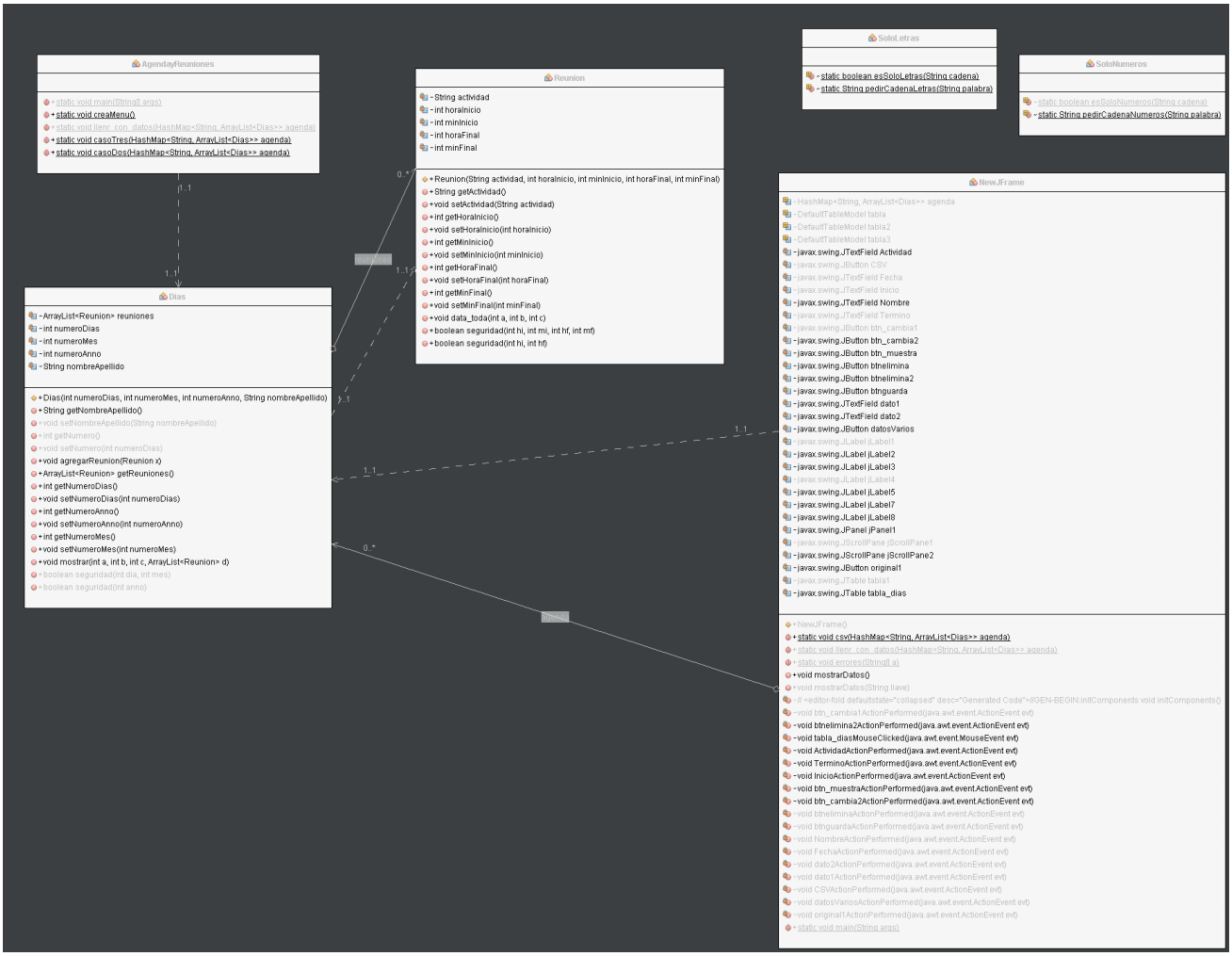
1.9 Todas las funcionalidades deben ser implementadas mediante consola (Sin ventanas)

Se implementaron las funciones solicitadas por consola.

1.10 Utilización de GitHub (Realización de al menos 3 Commit)

Como se puede ver en el historial de repositorio se hicieron los commit correspondientes.

2.1 Diseño de diagrama de clases UML



2.2 Persistencia de datos utilizando archivo de texto, CSV, Excel, o conexión con DBMS local (ej. MySQL). Utiliza sistema batch (carga datos al iniciar la aplicación y graba al salir)

Dentro del proyecto se encuentra un archivo csv, llamado “datos.csv”. El archivo se encuentra en la pestaña “Files” dentro de Apache NetBeans.

2.3 La implementación de todas las interfaces gráficas (ventanas) para interactuar con el usuario, considerando componentes SWING

En la clase “NewJFrame”, se encuentra la interfaz gráfica, en donde el usuario puede interactuar. Antes de interactuar se recomienda utilizar el botón llamado “Manual”, importante leer el manual antes de usar la ventana.

2.4 Se debe hacer un menú para el Sistema donde ofrezca las funcionalidades de: 1) Edición o modificación del elemento y 2) Eliminación del elemento. Esto para la 2ª colección de objetos (colección anidada) del SIA1.5

El menú esta implementado de manera visual en el archivo “NewJFrame.java” y en el se encuentran las funcionalidades requeridas.

Antes de explicar las funciones, es importante aclarar que la tabla 1, es donde se encuentran los nombres de las personas que están en la agenda y la tabla 2 es la posee toda la información respecto a la actividad guardada.

En la función private void btn_cambia1ActionPerformed(java.awt.event.ActionEvent evt), modifica los datos de la tabla 1.
En la función private void btn_cambia2ActionPerformed(java.awt.event.ActionEvent evt), modifica los datos de la tabla 2.
En la función private void btneliminaActionPerformed(java.awt.event.ActionEvent evt), elmina los datos de la tabla 1.
En la función private void btnelimina2ActionPerformed(java.awt.event.ActionEvent evt), elimina los datos de la tabla 2.

2.5 Se deben incluir al menos 1 funcionalidad propia que sean de utilidad para el negocio (distintas de la inserción, edición, eliminación y reportes)

Dentro del archivo, “NewJFrame.java”, se encuentra la función propia.
La función tiene como nombre: datosVariosActionPerformed.
Esta función lo que hace es calcular en número de personas que hay en la agenda y además calcula cuantas actividades existen por mes.
La función se puede utilizar mediante la interfaz gráfica. Apretando el botón llamado “Datos de la Agenda”.

2.6 El código fuente debe estar bien modularizado de acuerdo a lo descrito en el informe además de seguir las buenas prácticas de documentación interna y legibilidad

Para la correcta implementación del encapsulamiento y los principios de OO, eliminamos los getter y setters de mapas, nos aseguramos de que todos los atributos de las clases estuvieran en privacidad private y organizamos el código de tal forma que cada clase tenga sus propios métodos de manera que resguarde la integridad de los datos dentro del programa.

2.7 Diseño y codificación de 2 (dos) clases que utilicen sobreescritura de métodos

La primera clase es “SoloLetras” en donde se utiliza sobreescritura de métodos. Primero la función verifica si los String son formados solo por letras.
La sobreescritura la podemos ver en la función NoEsLetraExcepcion, donde una es con parámetros y la otra sin parámetro. Esta función avisa si el string contiene algún carácter que no sea letras.
Y la última función se ejecuta si el string son solo letras si no, ejecuta la excepción.

La primera clase es “SoloNumeros” en donde se utiliza sobreescritura de métodos. Primero la función verifica si los String son formados solo por dígito.
La sobreescritura la podemos ver en la función NoEsNumeroExcepcion, donde una es con parámetros y la otra sin parámetro. Esta función avisa si el string contiene algún carácter que no sea dígito.
Y la última función se ejecuta si el string son solo letras si no, ejecuta la excepción.

2.8 Implementar el manejo de excepciones capturando errores potenciales específicos mediante Try-catch

Los try-catch utilizados en el proyecto se pueden describir como los dos siguientes, cabe destacar que son excepciones para lectura y escritura de archivos. Y se pueden encontrar en el archivo “NewJPanel.java”.

- Estos try-catch lo que hacen es recibir Srting dentro de un array de String y testean primeramente si están vacíos. De estarlo, se avisa de la excepción. A continuación, el catch ve si cumple con la condición de que el string contiene solo caracteres y retorna si el string contiene algún carácter que no corresponda. De lo contrario no hace nada.

- Estos Try, aunque semejantes a los de strings. Estos reciben un string que se debe verificar si solo contienen números. Hacen lo mismo pro las condiciones fueron cambiadas en la clase.

2.9 Crear 2 clases que extiendan de una Excepción y que se utilicen en el programa

Las dos clases que utilizan excepciones fueron explicadas en el punto 2.7 del mismo informe.

2.10 Continuidad en la utilización de GitHub (Realización de al menos 3 Commit adicionales a los ya hechos en el avance)

Como se puede ver en el historial de repositorio se hicieron los commit correspondientes.

3.4 Se debe generar un reporte en archivo txt que considere mostrar datos de la colección de objetos

Para generar un reporte, es necesario utilizar el botón “CSV”, que se encuentra en la interfaz gráfica dentro del archivo “NewJFrame.java”.
El reporte llamado “agenda.csv” se encuentra en la pestaña “Files” dentro de Apache Neatbeans.