## Área PROGRAMACIÓN ORIENTADA A OBJETOS

**Excepciones Octubre 2017 Laboratorio 4/6**

**OBJETIVOS**

** Perfeccionar el diseño y código de un proyecto considerando casos especiales y errores.

** Construir clases de excepción encapsulando mensajes.

** Manejar excepciones considerando los diferentes tipos.

** Registrar la información de errores que debe conocer el equipo de desarrollo de una aplicación en producción.

** Vivenciar la prácticas ***Designing*** [*Simplicity*](http://www.extremeprogramming.org/rules/simple.html)*.*

**Coding** Code must be written to agreed [standards](http://www.extremeprogramming.org/rules/standards.html)

## ENTREGA

 Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.

 Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada,

en los espacios preparados para tal fin.

# EQUIPOS EN BLUEJ

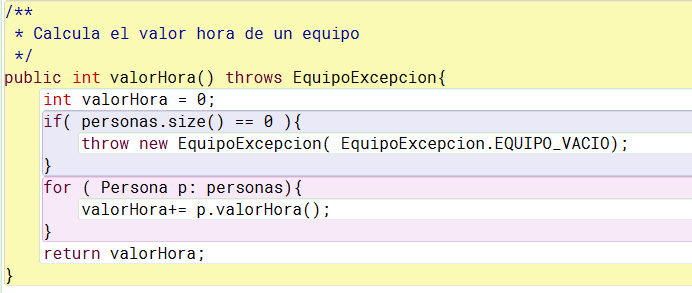
**PRACTICANDO MDD y BDD con EXCEPCIONES [En lab04.doc, equipos.asta y BlueJ equipos**

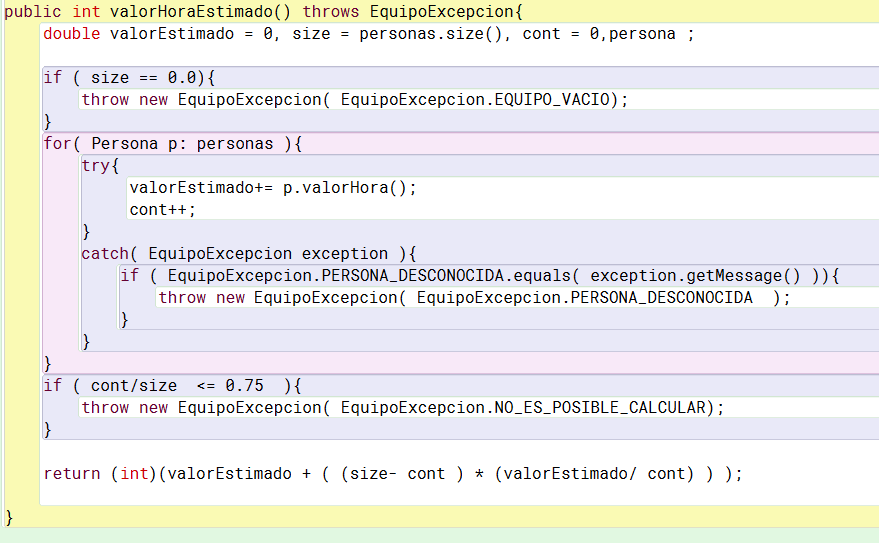
En este punto vamos a aprender a diseñar, codificar y probar usando excepciones. Para esto se van a trabajar dos métodos de la clase Equipo y la excepción EquipoExcepcion

1. **En su directorio descarguen los archivos contenidos en equipo.zip, revisen el contenido y estudien el diseño estructural de la aplicación.**
2. **¿De qué clase no pueden consultar las fuentes? ¿En qué formato está? ¿Por qué? Estudien su documentación en el diseño o en el código (/doc).**

No se pueden consultar las fuentes de la clase Persona, se encuentra en formato html(documentación), no se deja ver puesto que el código fuente de la clase Persona (archivo.java) no se encuentra en la carpeta proporcionada de equipos.

1. **Dadas las pruebas, diseñen y codifiquen el método valorHora.**



1. Dada la especificación, diseñen, codifiquen y prueben el método valorHoraEstimado.
2. 

### PARA LAS PRUEBAS SE TIENE EN CUENTA

Las siguientes personas que tienen valor hora Las siguientes personas son conocidas pero no conocido: tienen valor hora:

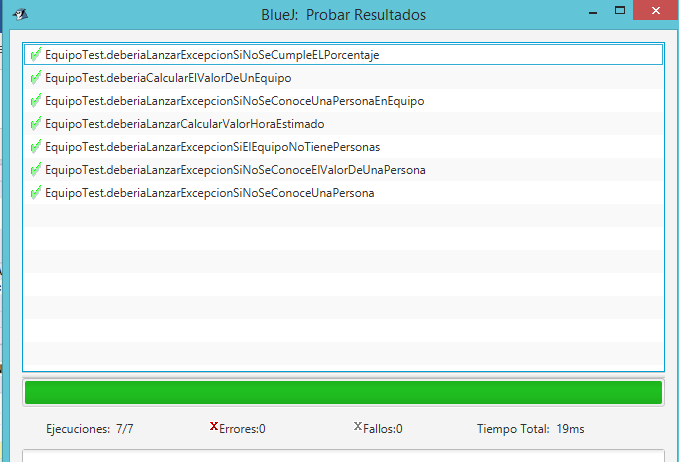
("Pedro",10000); ("Garcia");

("Santiago",20000); ("Ospina");

("Marcos",30000); ("Guarin")

("Juan",40000);

("Judas",50000);



**SINAP. Sistema Nacional de Áreas Protegidas**

**EN CONSOLA**

**Conociendo el proyecto SINAP [En lab04.doc]**

**No olviden respetar los directorios bin docs src**

1. En su directorio descarguen los archivos contenidos en SINAP.zip, revisen el contenido.

**¿Cuántos archivos se tienen?**

Se encuentran 5 archivos en el proyecto SINAP

**¿Cómo están organizados?**

No se encuentran organizados (Todos se encuentran en la misma carpeta), pero en los

**¿Cómo deberían estar organizados?**

Deberíaencontrarse separado por las carpetas **bin docs src** además deben estar separados por los paquetes aplicación y presentación.

1. **Estudien el diseño del programa: diagramas de paquetes y de clases.**

**¿cuántos paquetes tenemos?**

Tenemos 2 paquetes, aplicación, presentación

.

**¿cuántas clases tiene el sistema?**

4 clases **(Área, Registro, Sinap, SinapGUI)**

**¿cómo están organizadas?**

Por Paquetes (aplicación, presentación).

**¿cuál es la clase ejecutiva?**

SinapGUI

1. Prepare los directorios necesarios para ejecutar el proyecto. ¿qué estructura debe tener?

Src aplicación .java

Presentacion .java

Test .java

Bin aplicación .class

Presentacion .class

Test .class

Doc aplicación

Presentacion

Test

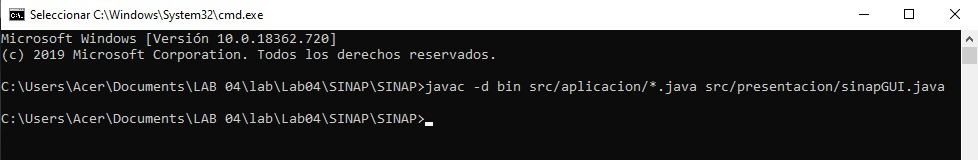
Debe estar separado por paquetes y por las carpetas src, bin ,docs

**¿qué instrucciones debe dar para ejecutarlo?**

Primero se debe compilar y luego ejecutar

Para Compilar

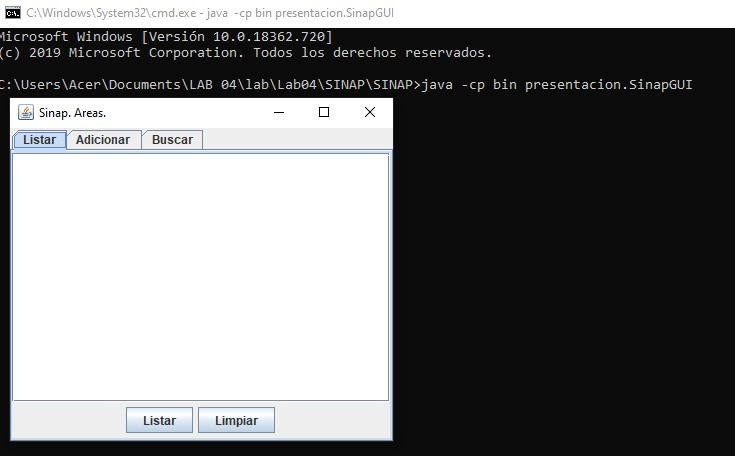




Para Ejecutar



1. Ejecute el proyecto Realicen el diagrama de casos de uso correspondiente.



¿qué funcionalidades ofrece?

* Listar
* Adicionar
* Buscar

¿cuáles funcionan?

* Listar
* Adicionar

**Diagramas de caso de uso Astah.**

1. **¿De dónde salen las áreas iniciales?** Las áreas iniciales salen de la clase Sinap, del método adicioneCinco y quien llama este método es la clase SinapGUI

**Revisen el código y la documentación del proyecto.**

**¿Qué clase pide que se adicionen?**

SinapGUI

**¿Qué clase los adiciona?**

Sinap

**Adicionar y listar. Todo OK.** [En lab04.doc, SINAP.asta y \*.java]

(NO OLVIDEN BDD - MDD)

El objetivo es realizar ingeniería reversa a las funciones de adicionar y listar.

1. Adicionen una nueva área

Tuparro

Tuparro National Park

Vichada

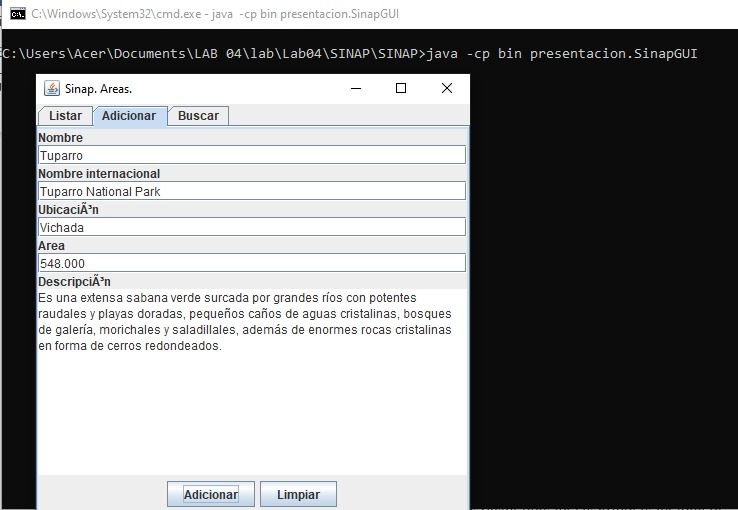
548.000

Es una extensa sabana verde surcada por grandes ríos con potentes raudales y playas doradas, pequeños caños de aguas cristalinas, bosques de galería, morichales y saladillales, además de enormes rocas cristalinas en forma de cerros redondeados.

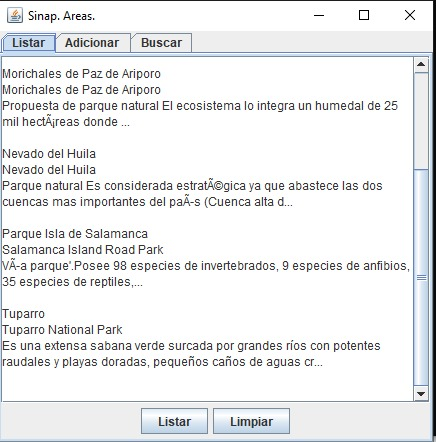
Primero se debió ejecutar el programa, para luego meter por un

¿**Qué ocurre? ¿Cómo lo comprueban? Capturen la pantalla. ¿Es adecuado este comportamiento?**

Aparece un programa para añadir, listar o buscar información



Cuando abrimos el método listar, este no permite mostrar la ubicación ni el área de lo ingreasado.



**¿Es adecuado este comportamiento?**

No es correcto ya que no muestra toda la información establecida.

1. **Revisen el código asociado a adicionar en la capa de presentación y la capa de aplicación.**

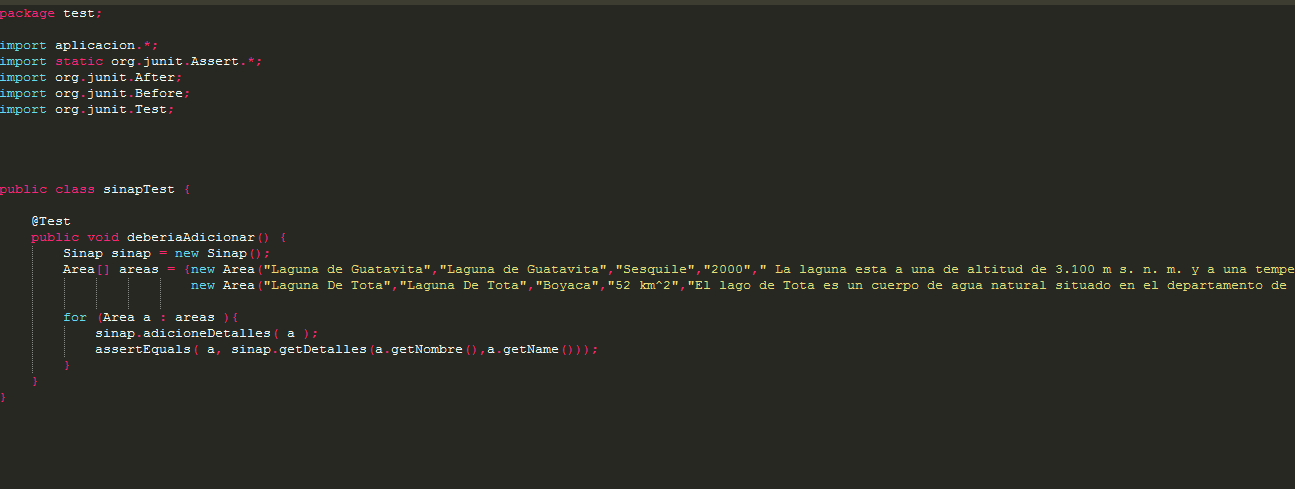
**¿Qué método es responsable en la capa de presentación?**

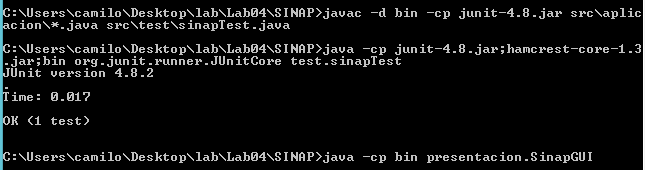
En la capa presentación el método encargado es accionAdicionar() perteneciente a la clase SinapGUI

**¿Qué método en la capa de aplicación?**

En el método adicione() de la clase Sinap

1. **Realicen ingeniería reversa para la capa de aplicación para adicionar. Capturen los resultados de las pruebas de unidad.**





1. **Revisen el código asociado a listar en la capa de presentación y la capa de aplicación.**

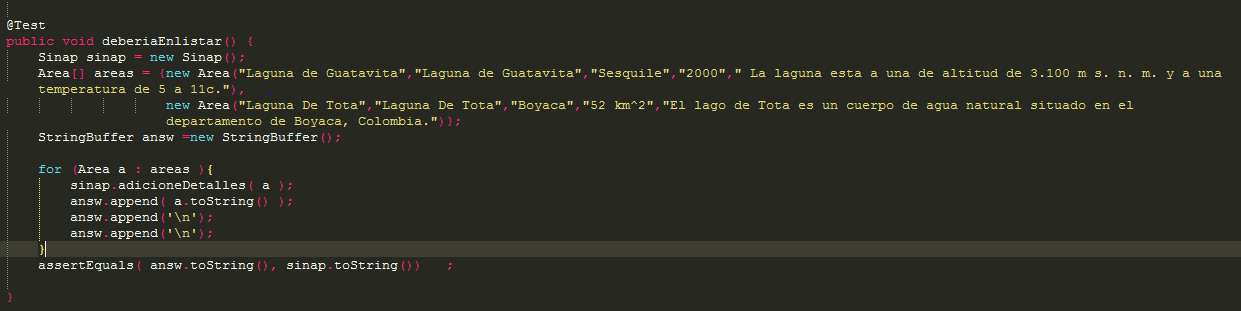
**¿Qué método es responsable en la capa de presentación?**

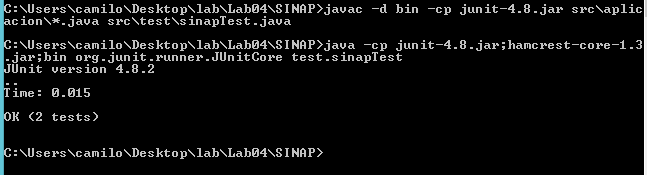
El método responsable de enlistar el input es el método accionListar() de la clase SinapGUI

**¿Qué método en la capa de aplicación?**

El método que se encarga de enlistar las áreas es toString() en la clase Sinap

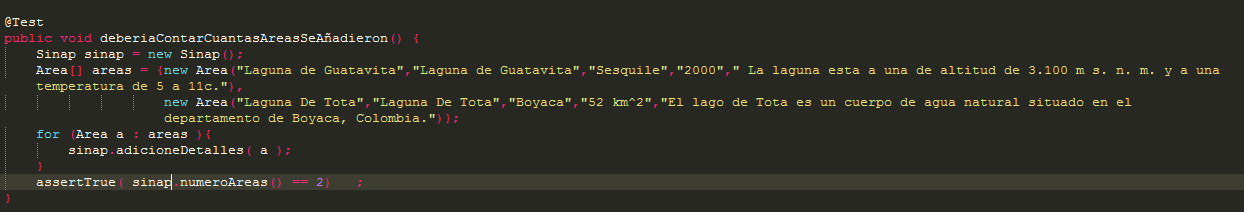
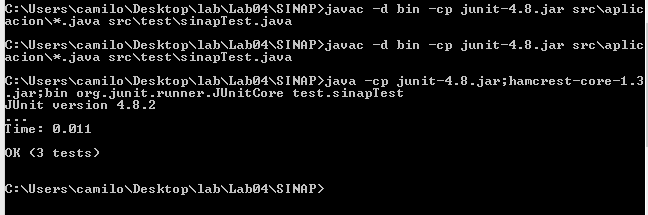
1. **Realicen ingeniería reversa para la capa de aplicación para listar. Capturen los resultados de las pruebas de unidad.**





1. **Propongan y ejecuten una prueba de aceptación.**

La prueba fue **deberiaContarCuantasAreasSeAñadieron()**



**Adicionar un área. ¿Y si no da un nombre?** [En lab04.doc, SINAP.asta y \*.java]

(NO OLVIDEN BDD – MDD)

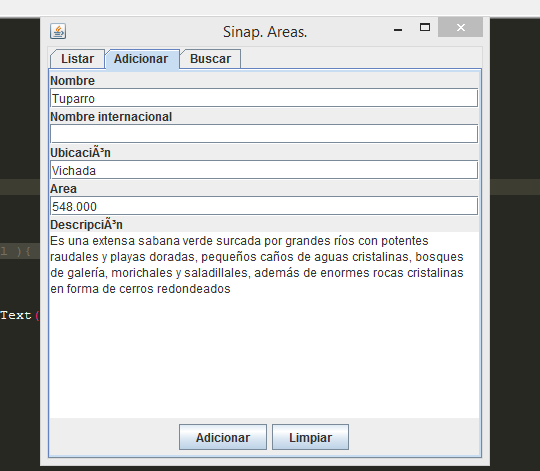
El objetivo es perfeccionar la funcionalidad de adicionar un área.

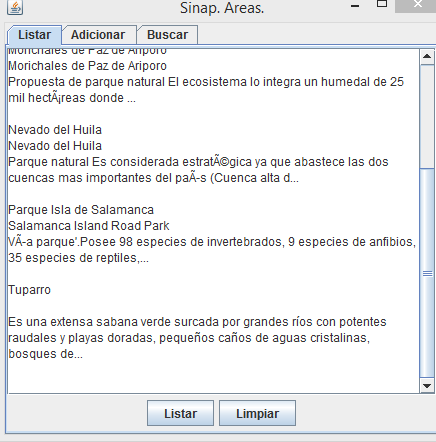
1. **Adicionen el área Tuparro sin nombre internacional.**

**¿Qué ocurre?**

Permite ingresar el área aunque no posea un nombre internacional.

**¿Cómo lo comprueban? Capturen la pantalla.**



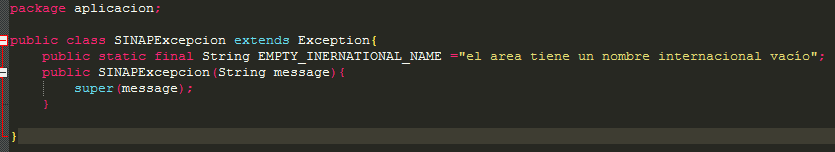


**¿Es adecuado este comportamiento?**

No es adecuado ya Permite ingresar el área, aunque no posea un nombre internacional.

1. **Vamos a evitar la creación de áreas con un nombre internacional vacío manejando una excepción** **SINAPExcepcion**. **Si el área no tiene nombre internacional, no la creamos y se**

**lo comunicamos al usuario**[**1**](#_bookmark0)**. Para esto lo primero que debemos hacer es crear la nueva clase** **SINAPExcepcion** **considerando este primer mensaje**.



1. **Analicen el diseño realizado.**

**¿Qué método debería lanzar la excepción?**

* **El método adicione de la clase Sinap:** ya que es el método que llama a la clase Area

la cual es la encargada de guardar los datos que fueron ingresados por el componente periférico **(Teclado).**

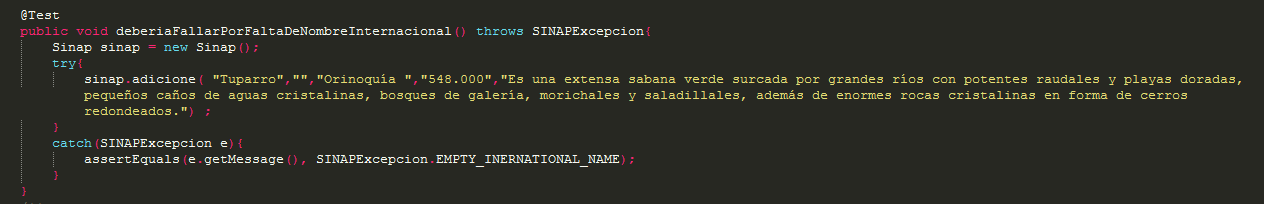
**¿Qué métodos deberían propagarla?**

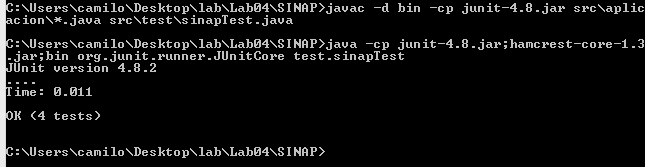
* **El método accionAdicionar de la clase SinapGUI:** ya que fue el método el cual fue llamado para que se añadiera, por lo que debe decirle a actionPerformed si se pudo completar la operación o no.

**¿Qué método debería atenderla? Explique claramente.**

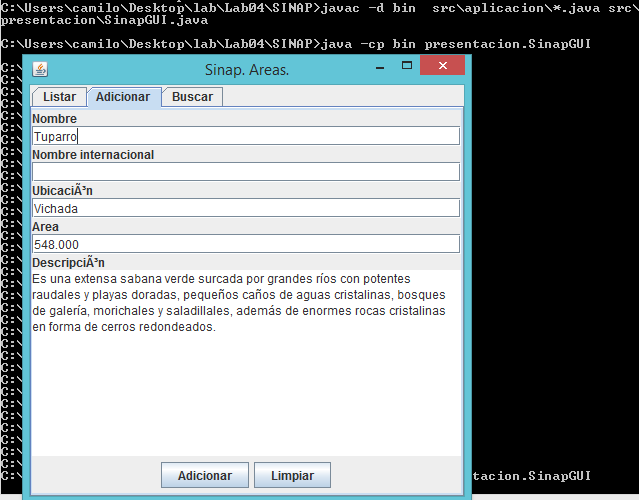
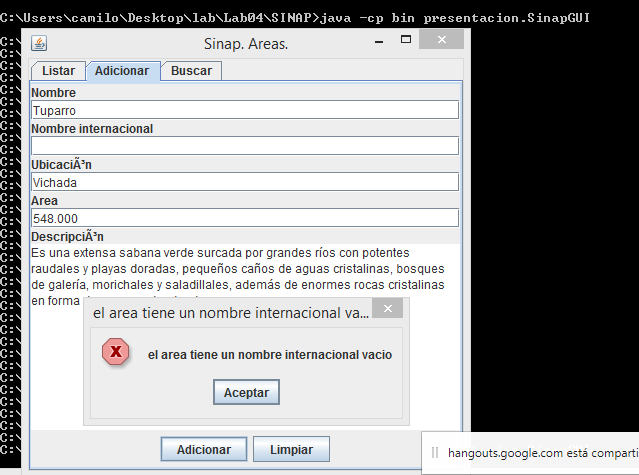
* **Metodo actionPerformed de la clase SinapGUI:** Ya que este será el que decida si el método debe ejecutarse o no.

1. **Construya la solución propuesta. Capturen los resultados de las pruebas**





1. **Ejecuten nuevamente la aplicación con el caso de prueba propuesto en 1., ¿Qué sucede ahora? Capture la pantalla.**



**Adicionar un área. ¿Y si ya se encuentra?** [EnSINAP.asta, lab04.java y \*.java]

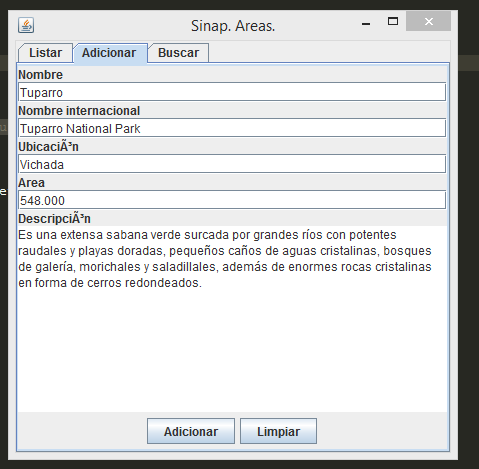
(NO OLVIDEN BDD - MDD)

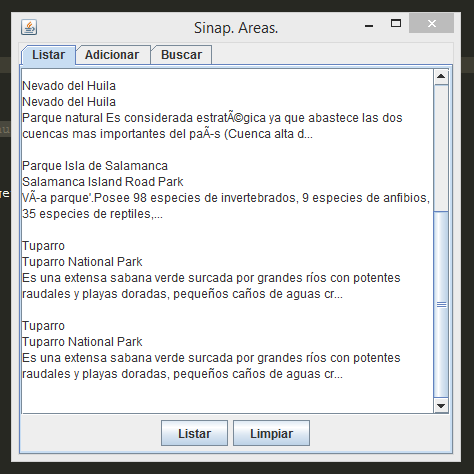
El objetivo es perfeccionar la funcionalidad de adicionar un área.

1. **Adicionen dos veces la nueva área Tuparro ¿Qué ocurre? ¿Cómo lo comprueban? Capturen la pantalla. ¿Es adecuado este comportamiento?**

Permite Añadir información de una misma área 2 o más veces, este comportamiento no es adecuado ya

que permite redundancia.





1. **Analicen el diseño realizado. ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique claramente.**

**¿Qué método debería lanzar la excepción?**

* **El método adicione de la clase Sinap:** ya que es el método que llama a la clase Area

la cual es la encargada de guardar los datos que fueron ingresados por el componente periférico **(Teclado).**

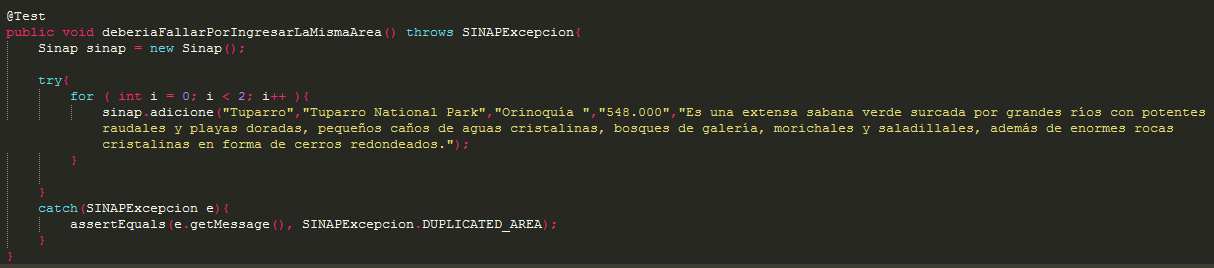
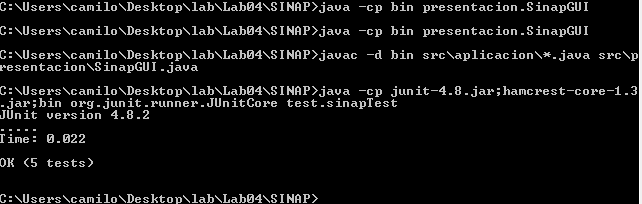
**¿Qué métodos deberían propagarla?**

* **El método accionAdicionar de la clase SinapGUI:** ya que fue el método el cual fue llamado para que se añadiera, por lo que debe decirle a actionPerformed si se pudo completar la operación o no.

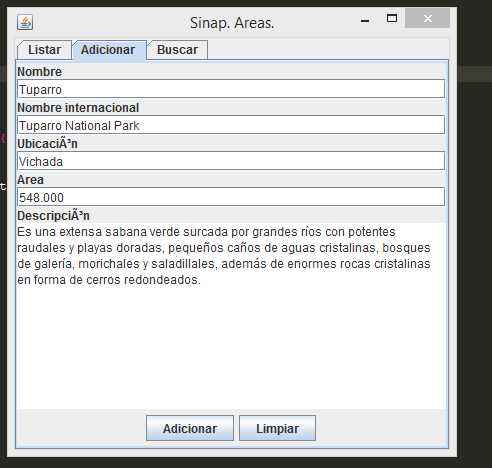
**¿Qué método debería atenderla? Explique claramente.**

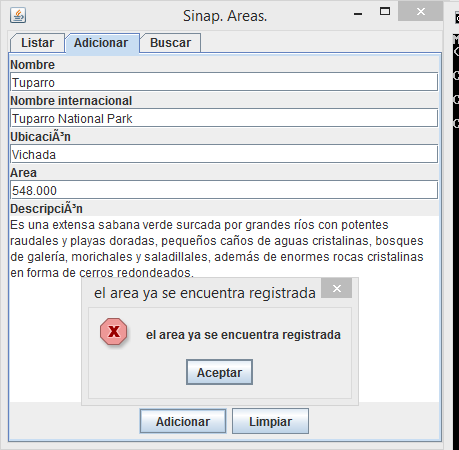
* **Metodo actionPerformed de la clase SinapGUI:** Ya que este será el que decida si el método debe ejecutarse o no.

1. **Construya la solución propuesta. Capturen los resultados de las pruebas**



1. **Ejecuten nuevamente la aplicación con el caso de prueba propuesto en 1., ¿Qué sucede ahora? Capture la pantalla.**





**Adicionar un área. ¿Otras condiciones?** [En lab04.doc, SINAP.asta y \*.java]

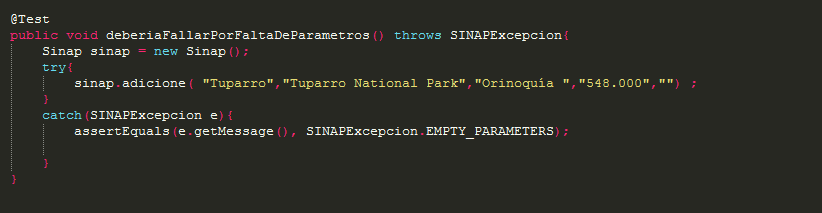
(NO OLVIDEN BDD – MDD)

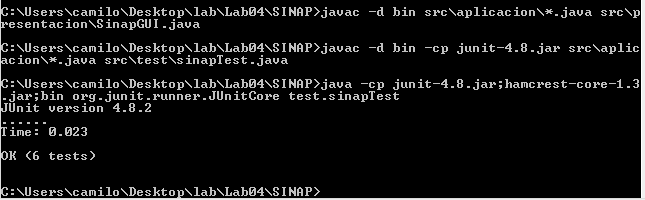
**El objetivo es perfeccionar la funcionalidad de adicionar un área.**

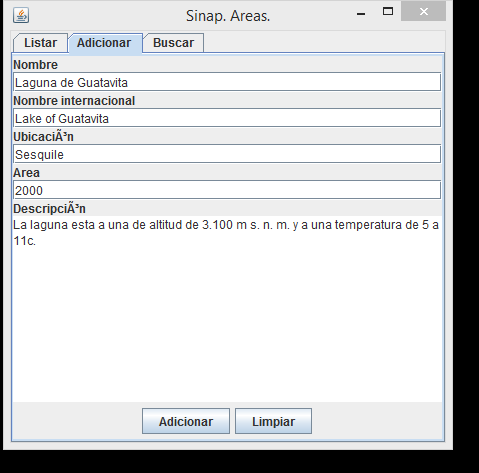
1. **Propongan nuevas condiciones para que la adición de un área sea más robusta.**[**2**](#_bookmark1)

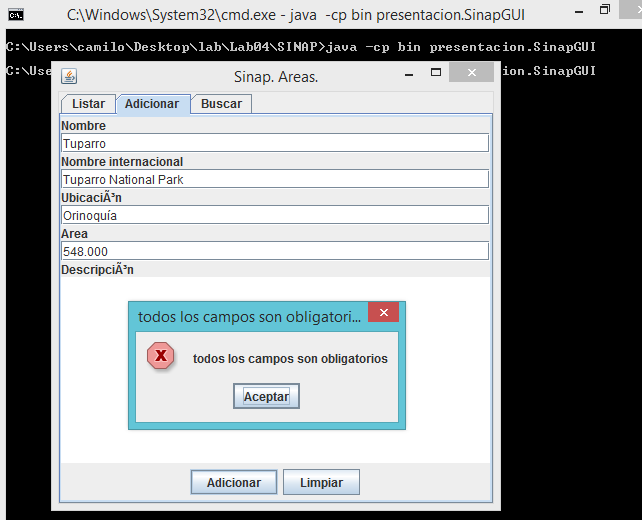
Para que un área sea más, añadimos la excepción de que es necesario que todos los campos sobre esta deben ser completados obligatoriamente.

1. **Construya la solución propuesta. (diseño, prueba de unidad, código) Capturen los resultados de las pruebas.**

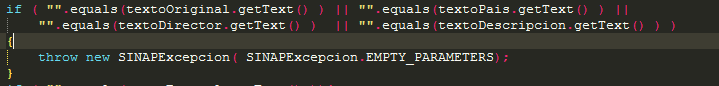
**Prueba:**







**Codigo:**



### Consultando por patrones. ¡ No funciona y queda sin funcionar!

[EnSINAP.asta,SINAP.log, lab04.java y \*.java]

(NO OLVIDEN BDD - MDD)

 **Consulten un área especial que inicie con T.**

**¿Qué sucede?**

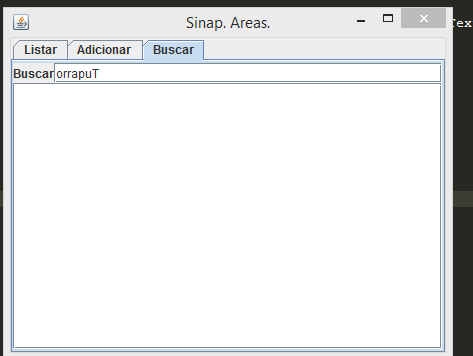
no busca el nombre introducido y además en la barra de búsqueda guarda la entrada al revés **(En consola salen muchos erros)**.

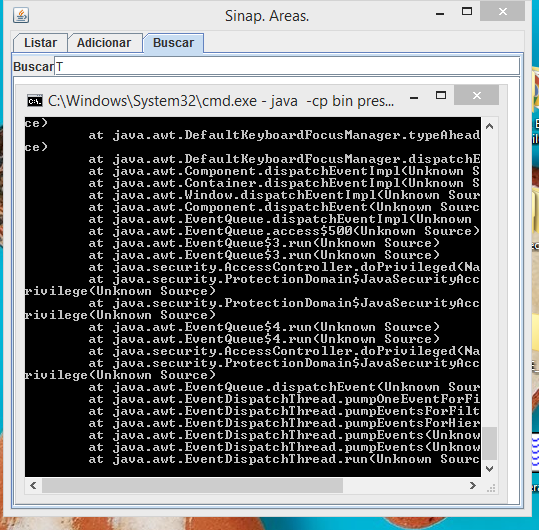
**¿Qué creen que pasó?**

No debe estar registrando todas las excepciones.

**Capturen el resultado. ¿Quién debe conocer y quien NO debe conocer esta información?**

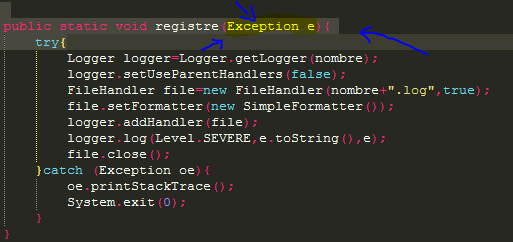
Los Programadores debe ver los errores que deben almacenarce en el logger, el usuario no debe saber de su existencia.





** Exploren el método registre de la clase Registro ¿Qué servicio presta?**

* En la clase Registre Se crea un nuevo objeto tipo **Logger (utilizado para registrar mensajes para un sistema o componente de aplicación específico)** en este caso para registrar las excepciones del sistema, ya que el único parámetro que recibe es de tipo Excepción, como se logra apreciar en la imagen.



** Analicen el punto adecuado para que SIEMPRE, al sufrir en cualquier punto el sistema un incidente como este, se presente un mensaje especial de alerta al usuario, se guarde la información del error en el registro de error y termine la ejecución. Expliquen y construyan la solución.**

* Para evitar que en cualquier punto el sistema sufra un incidente se debe realizar un try and catch en cada acción que pueda realizar SinapGUI **(Adicionar, Buscar, Listar )** y para registrar se debe poner Registro.registre(exception), donde excepción fue la excepción atrapada.

** Ejecuten nuevamente la aplicación con el caso propuesto en 1.**

**¿Qué mensaje salió en pantalla?**

Sale el mensaje establecido por nosotros con el JOptionPane **Opss ocurrió un error al momento de buscar**

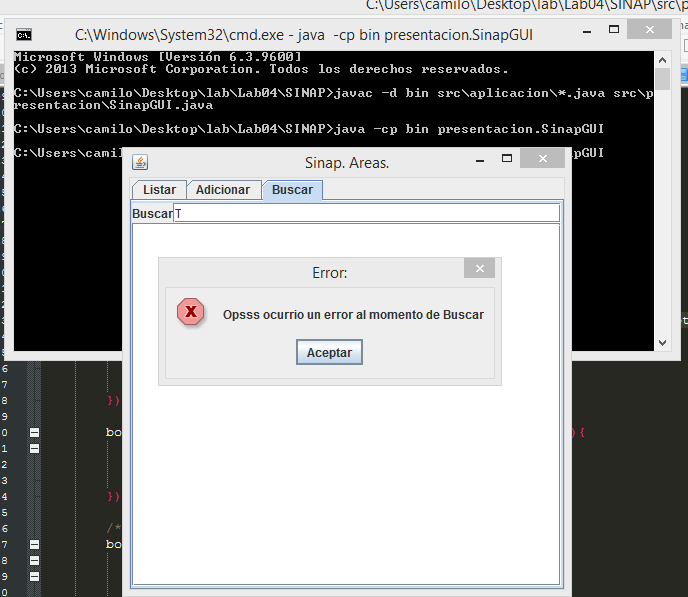
**¿La aplicación termina?**

la aplicación se cierra apenas se da aceptar del mensaje ya que se **solicitó** **guarde la información del error en el registro de error y termine la ejecución.**

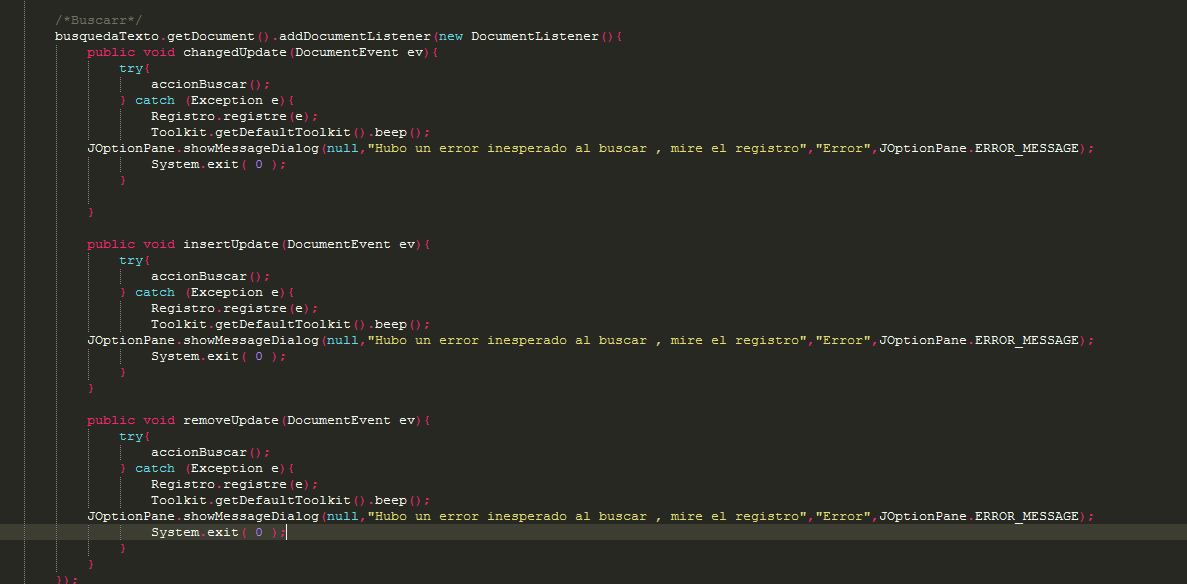
**¿Qué información tiene el archivo de errores?**

“Al llamar el método **log** del **Logger** se le indica el nivel del log y el mensaje que deseamos registrar, esto automáticamente reportara la fecha, hora, el nombre completo de la clase y el método y en el caso de mensajes de nivel grave la línea de código donde se generó el reporte.

**Sacado de :”**[**https://medium.com/el-acordeon-del-programador/logs-en-java-con-java-util-logging-d344ae2ba7bc**](https://medium.com/el-acordeon-del-programador/logs-en-java-con-java-util-logging-d344ae2ba7bc)**”**



**Implementación**



** ¿Es adecuado que la aplicación continúe su ejecución después de sufrir un incidente como este? ¿de qué dependería continuar o parar?**

No necesariamente el programa o el sistema debe parar de ejecutarse, para Búsqueda no sería necesario, dependería de la propagación de nuevas excepciones que este ocasione

1 Para presentar los mensajes de error al usuario use el méodo de clase de JOptionPane public static void **showMessageDialog**(Component parentComponent,

Object message, String title, int messageType)

throws HeadlessException

Con componente padre:este mensaje: la cadena correspondiente al mensaje de error de la excepcion correspondiente, titulo: ERROR y tipo de mensaje: JOptionPane.ERROR\_MESSAGE

2Robustez o solidez. Se refiere a la capacidad del software de defenderse de las acciones anormales que llevan al sistema a un estado no deseado o por lo menos no previsto, causando un comportamiento inesperado, indeseado y posiblemente erróneo

 **Analicen el punto adecuado para que EN ESTE CASO se presente un mensaje especial de alerta al usuario, se guarde la información del error en el registro y continúe la ejecución. Expliquen y construyan la solución. No eliminen la solución de 3.**

En la implementación anterior de acción de buscar simplemente documentamos el comando **System.exit(0);**

** Ejecuten nuevamente la aplicación con el caso propuesto en 1.**

** Ejecuten nuevamente la aplicación con el caso propuesto en 1.**

**¿Qué mensaje salió en pantalla?**

Sale el mensaje establecido por nosotros con el JOptionPane **Opss ocurrió un error al momento de buscar**

**¿La aplicación termina?**

la aplicación **NO** se cierra apenas se da aceptar del mensaje ya que se **no es necesario ejecutar el comando System.exit(0);**

**¿Qué información tiene el archivo de errores?**

Automáticamente reportara la fecha, hora, el nombre completo de la clase y el método y en el caso de mensajes de nivel grave la línea de código donde se generó el reporte, junto al error.

### Consultando por patrones. ¡Ahora si funciona!

(NO OLVIDEN BDD - MDD)

1. **Revisen el código asociado a buscar en la capa de presentación y la capa de aplicación.**

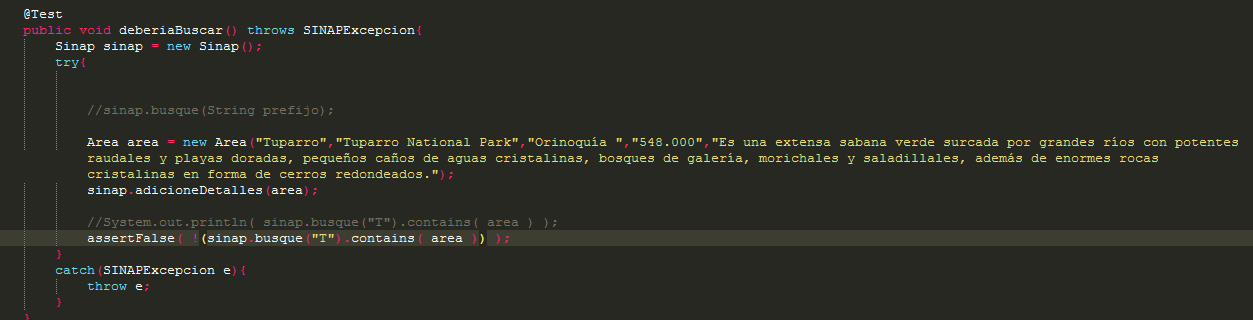
**¿Qué método es responsable en la capa de presentación?**

**Presentación:** El método accionBuscar

**¿Qué método es responsable en la capa de aplicación?**

**Aplicación:** El método busque de la clase Sinap

1. **Realicen ingeniería reversa para la capa de aplicación para buscar. Capturen los resultados de las pruebas. Deben fallar.**



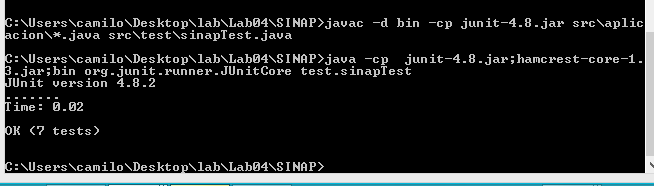


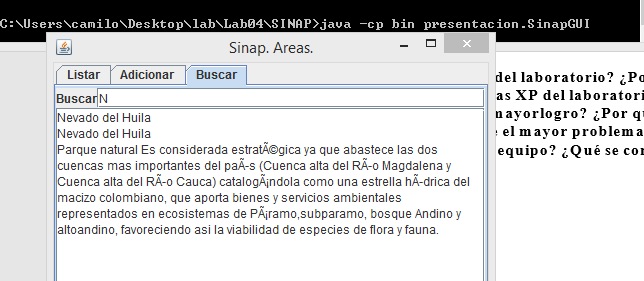
1. **¿Cuál es el error? Soluciónenlo. Capturen los resultados de las pruebas.**

El error es esta línea de código que se encuentra en la función Sinap

**Corrección:**

1. **Ejecuten la aplicación nuevamente con el caso propuesto.**



**¿Qué tenemos en pantalla?**

**¿Qué información tiene el archivo de errores?**

La fecha, hora, el nombre completo de la clase y el método y en el caso de mensajes de nivel grave la línea de código donde se generó el reporte, junto al error.

**RETROSPECTIVA**

**1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes?**

**(Horas/Hombre)**

**Iván Camilo Rincón :12 horas.**

**Miguel Ángel Fúquene Arias: 12 horas.**

**2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?**

Terminado, puesto que le invertimos el tiempo y la disposición requerida durante toda la semana.

**3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?**

La programación a pares, ya que con esto ambos integrantes del grupo aprenden y se practican los temas que se vieron durante la clase, adicionalmente también nos sirvió la práctica “Solo un par integra código a la vez” puesto que podemos hacerle mejoras o trabajarle al laboratorio desde cada uno de nuestros computadores teniendo en cuenta lo que el otro compañero ya había adicionado para no modificar u omitir lo que ya se había hecho.

**4. ¿Cuál consideran fue el mayor logro? ¿Por qué?**

Pudimos aplicar los conceptos vistos en clase satisfactoriamente puesto que pudimos realizar todas las funcionalidades y requerimientos que se pedían.

**5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para**

**resolverlo?**

Conocimiento de algunos comandos nuevos que aparecía, sin embargo, se investigó y se hicieron varias pruebas para determinar la funcionalidad de cada uno.

**6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los**

**resultados?**

Distribuimos bien el tiempo y utilizamos espacios de estudio fuera de clase para fortalecer los conocimientos y poder realizar todas las funcionalidades que proponía el laboratorio. Nos comprometemos a reforzar los temas vistos.