# PROGRAMACIÓN ORIENTADA A OBJETOS

**I/O 2020-01**

**Laboratorio 6/6 [ :) ]**

## OBJETIVOS

1. Completar el código de un proyecto considerando requisitos funcionales.
2. Diseñar y construir los métodos básicos de manejo de archivos: abrir, guardar como, importar y exportar.
3. Controlar las excepciones generadas al trabajar con archivos.
4. Experimentar las prácticas XP : *Refactor whenever and wherever possible.*



All code must pass all [unit tests](http://www.extremeprogramming.org/rules/unittests.html) before it can be released. -

# 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.

 En el foro de entrega de avance deben indicar los logros y los problemas pendientes por resolver.

## DESARROLLO

**Preparando**

En este laboratorio vamos a extender el proyecto automata adicionando un menú barra con las opciones básicas de entrada-salida y las opciones estándar nuevo y salir.

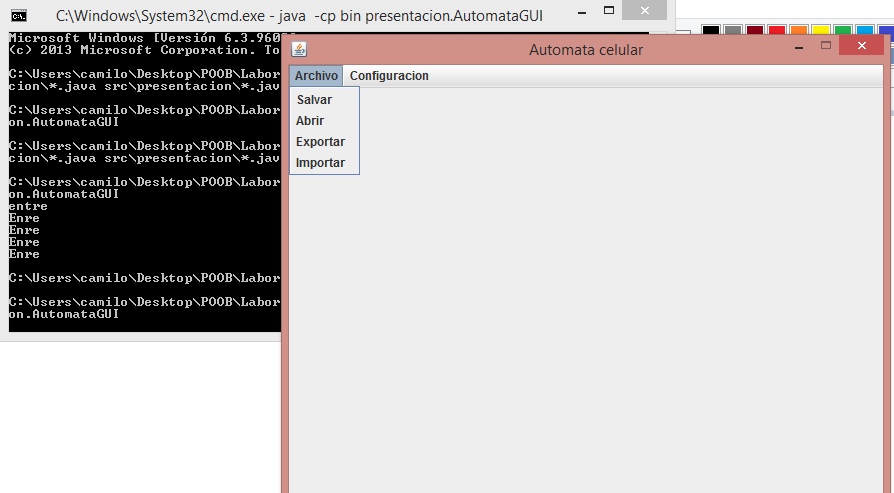
1. En su directorio descarguen la versión del proyecto realizado por ustedes para el laboratorio 03 y preparen el ambiente para trabajar desde **CONSOLA**
2. Ejecuten el programa, revisen la funcionalidad.

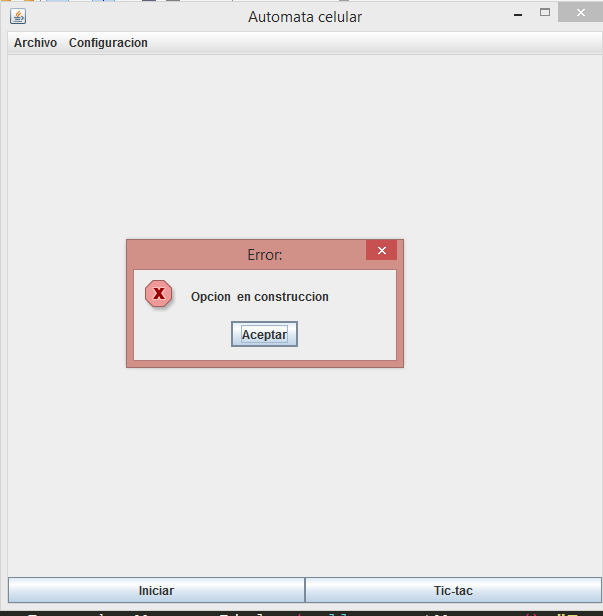
## Creando la maqueta

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

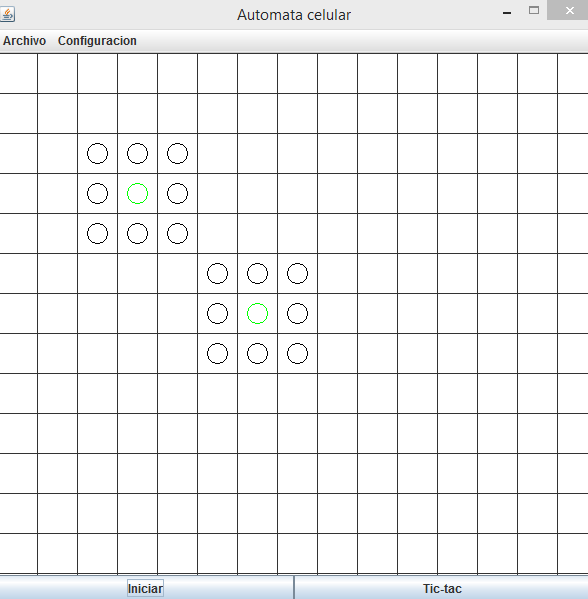
En este punto vamos a construir la maqueta correspondiente a esta extensión siguiendo el patrón MVC.

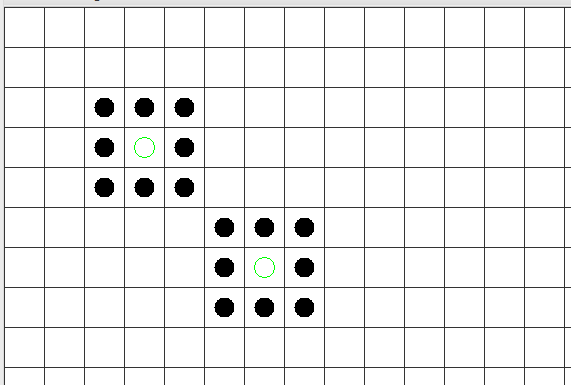
1. **MODELO:** Preparen los métodos correspondientes a **reiniciar** y a las cuatro opciones básicas de entrada-salida (salve,abra,exporte,importe). Los métodos deben simplemente propagar una automataExcepcion con el mensaje: “Opción … en construcción”. Los métodos de entrada salida deben tener un parámetro File.

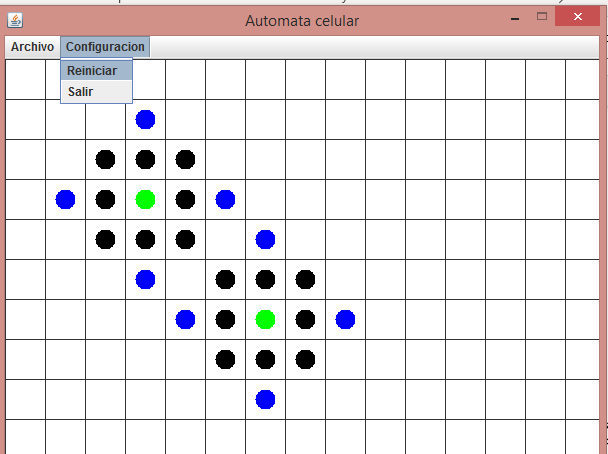


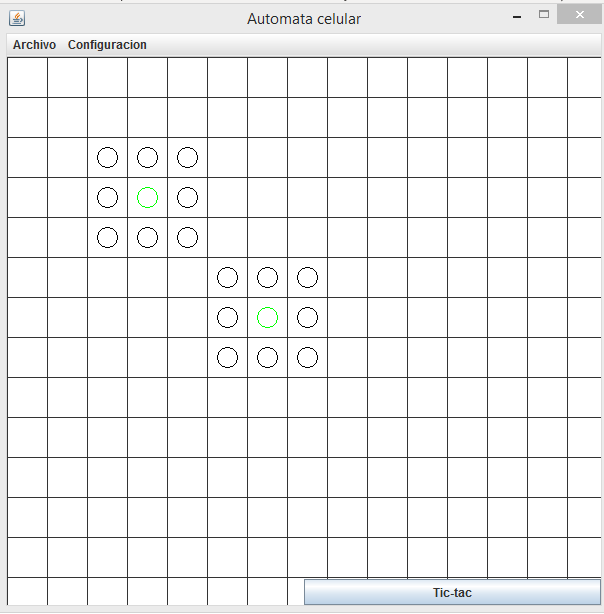


**Reiniciar**

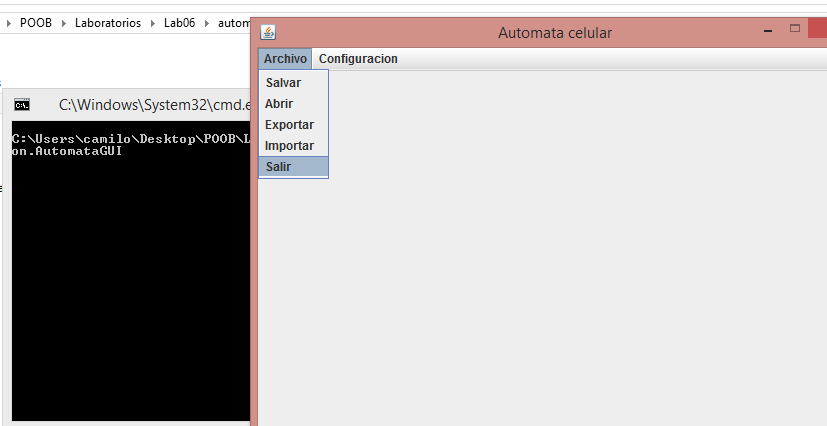


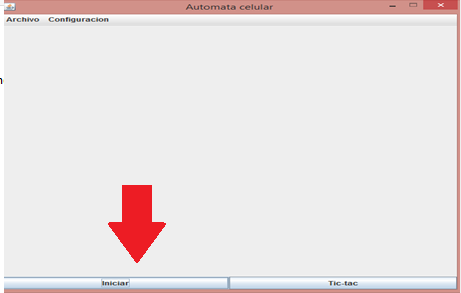


****



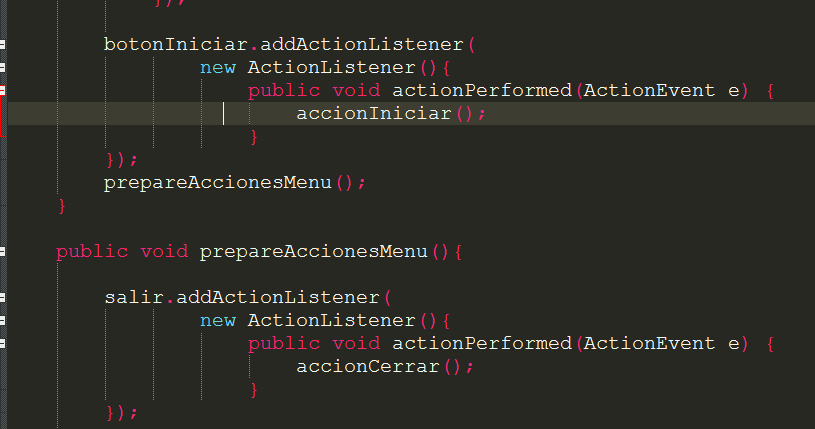
1. **VISTA :** Construyan un menú barra que ofrezca, además de las opciones básicas de entrada-salida, las opciones estándar de iniciar y salir. Para esto creen el método prepareElementosMenu. Capturen la pantalla correspondiente.





1. **CONTROLADOR:** Construyan los controladores correspondientes a estas acciones. Para esto creen el método prepareAccionesMenu y los métodos base del controlador

( opcionSalver,opcionAbir, opcionExportar, opcionImportar, opcionReiniciar, opcionCerrar), Estos métodos, por ahora, **llaman directamente el método correspondiente de la capa de aplicación**. Capturen una pantalla significativa.

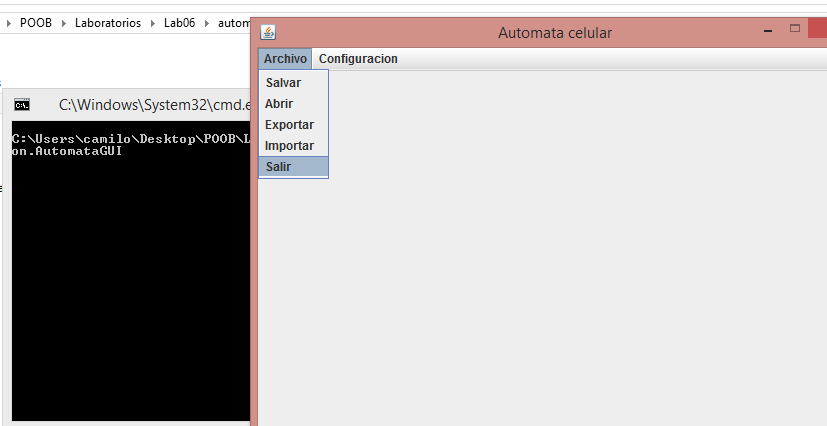


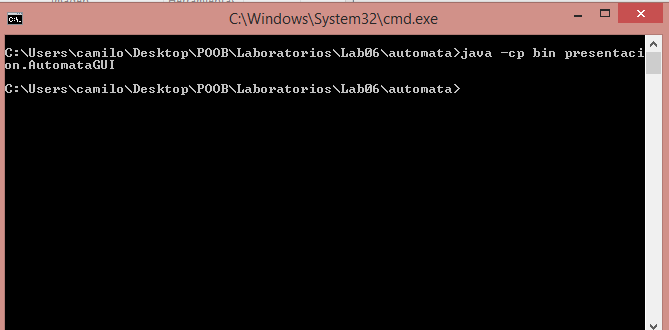
## Implementando salir e iniciar

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

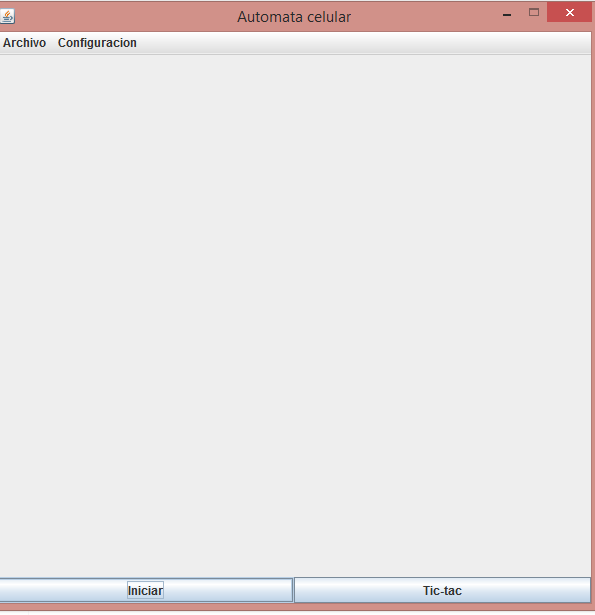
Las opciones salir e iniciar van a ofrecer los dos servicios estándar de las aplicaciones. El primero no requiere ir a capa de aplicación y el segundo sí.

1. Construyan el método opcionSalir que hace que se termine la aplicación. No es necesario incluir confirmación.

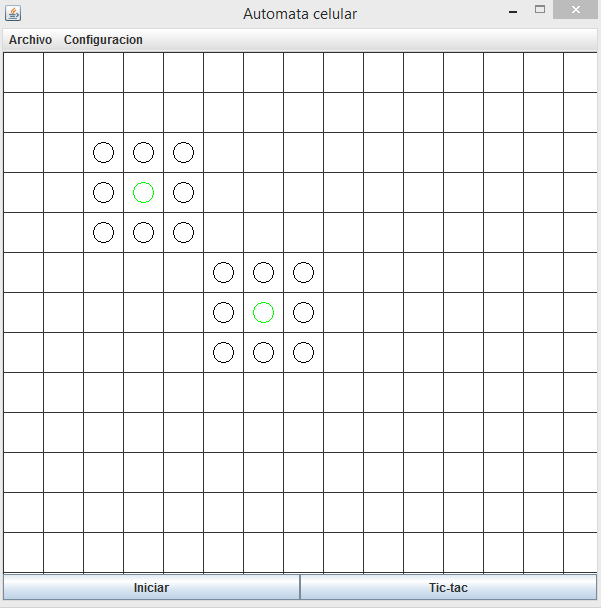




1. Construyan el método opcionIniciar que crea un nuevo automata. Capturen una pantalla significativa.



**Aparecerá con una configuración inicial**



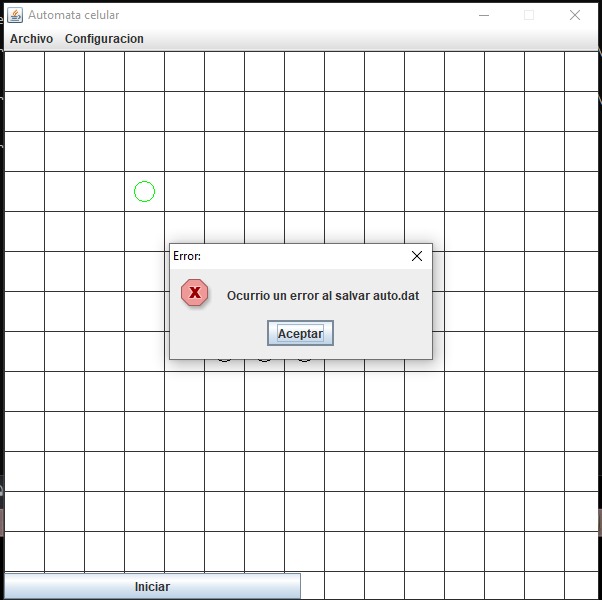
## Implementando salvar y abrir

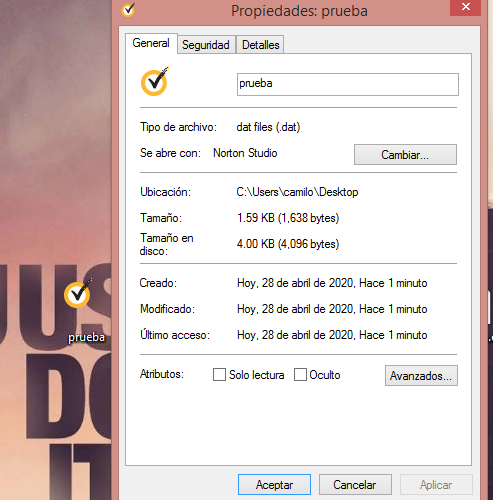
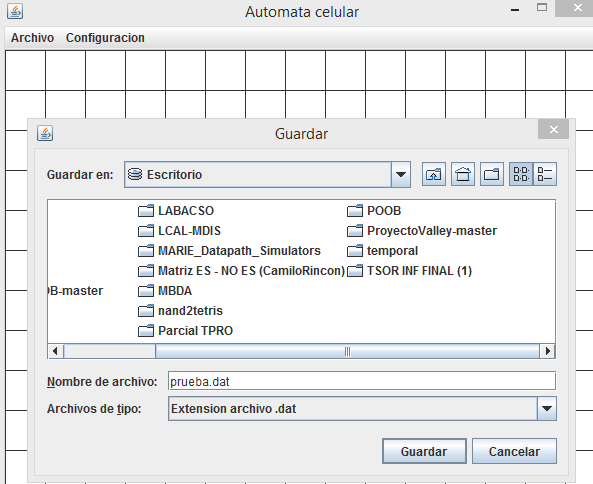
**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

Las opciones salvar y abrir van a ofrecer servicios de persistencia de un automata como objeto. Los nombres de los archivos deben tener como apellido .dat.

**Para filtrar al abrir o salvar usar** [**FileNameExtensionFilter**](https://docs.oracle.com/javase/7/docs/api/javax/swing/filechooser/FileNameExtensionFilter.html#FileNameExtensionFilter(java.lang.String,%20java.lang.String...))**(**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**description,**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**... extensions)**

1. Construyan el método opcionSalvar que une de forma adecuada la capa de presentación con la capa de aplicación. Usen un FileChooser y atiendan la excepción. Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.



1. Construyan el método salve que ofrece el servicio de guardar en un archivo el estado
2. Validen este método guardando el autómata inicial después de dos clics como

unautomata.dat.

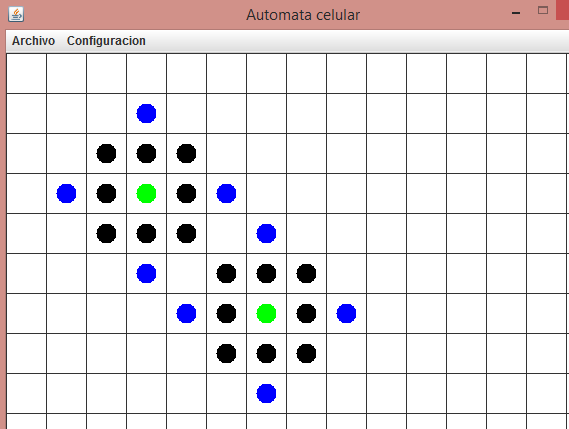
**¿El archivo se creó en el disco?**

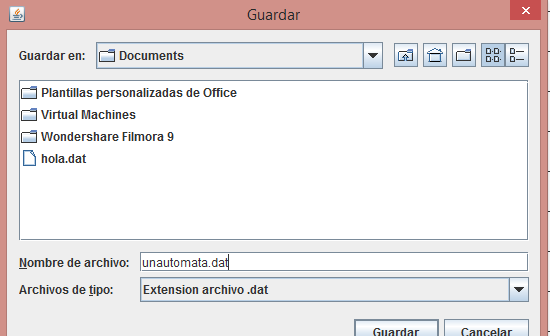
Si se crea en disco

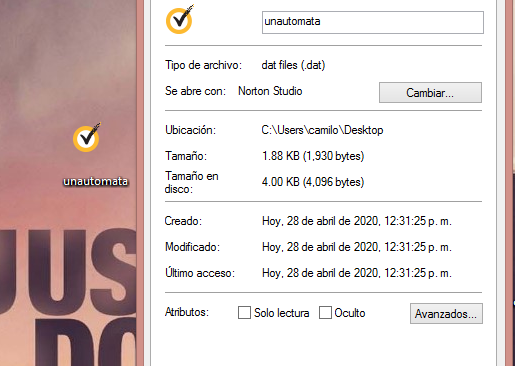
**¿Cuánto espacio ocupa?**

El tamaño es de 1.88 kb y en disco es de 4.00 kb

Autómata a guardar





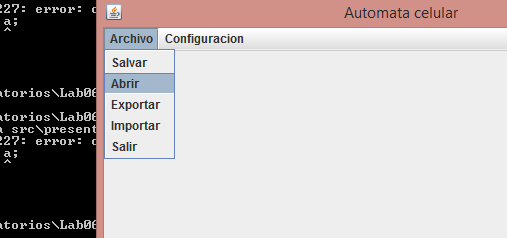


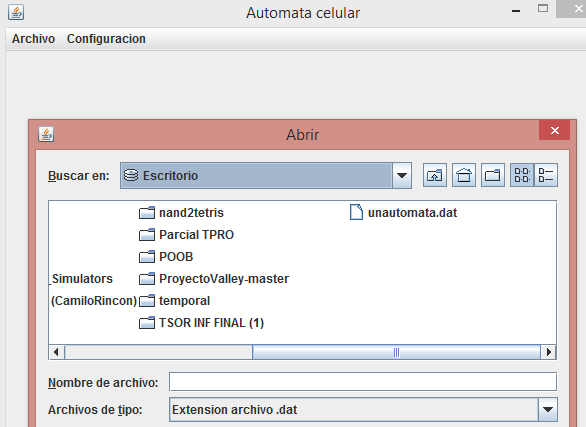
**IMPORTANTE**

“Para que un programa [*java*](http://chuwiki.chuidiang.org/index.php?title=Categor%C3%ADa:Java) pueda convertir un objeto en un montón de *bytes* y pueda luego recuperarlo, el objeto necesita ser **Serializable**. Al poder convertir el objeto a *bytes*, ese objeto se puede [enviar a través de red](http://www.chuidiang.com/java/sockets/socket.php), [guardarlo en un fichero](http://www.chuidiang.com/java/ficheros/ObjetosFichero.php), y después reconstruirlo al otra lado de la red, leerlo del fichero,....

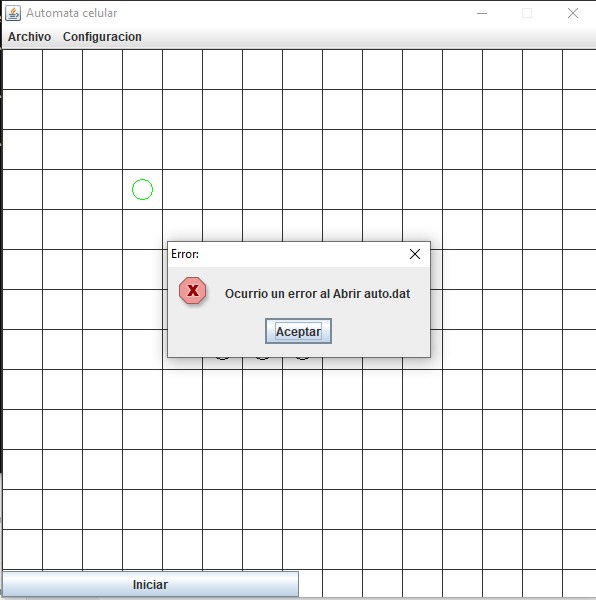
Para que un objeto sea serializable basta con que implemente la interfaz **Serializable**. Como la interfaz **Serializable** no tiene métodos, es muy sencillo implementarla, basta con un *implements Serializable* y nada más. Por ejemplo, la clase *Datos* siguiente es **Serializable** y java sabe perfectamente enviarla o recibirla por red, a través de [*socket*](http://www.chuidiang.com/java/sockets/socket.php) o de [rmi](http://www.chuidiang.com/java/rmi/rmi.php). También *java* sabe escribirla en un fichero o reconstruirla a partir del fichero.” Tomado de wikipedia

1. Construyan el método opcionAbrir que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.



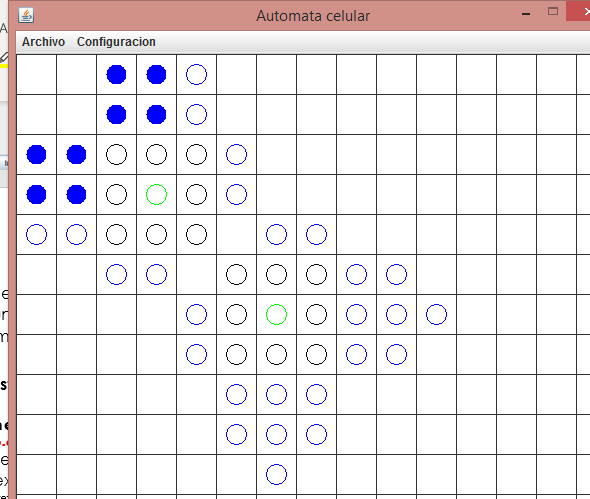


1. Construyan el método abra que ofrece el servicio de leer un automata de un archivo. Por ahora para las excepciones sólo consideren un mensaje de error general.

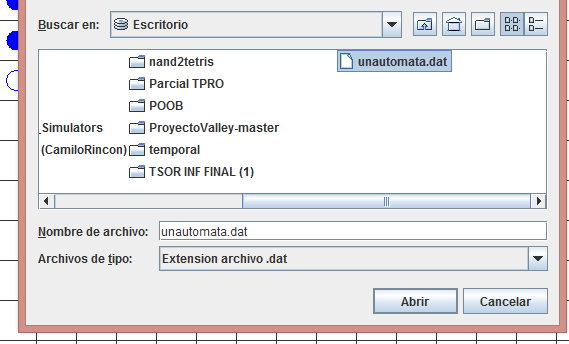
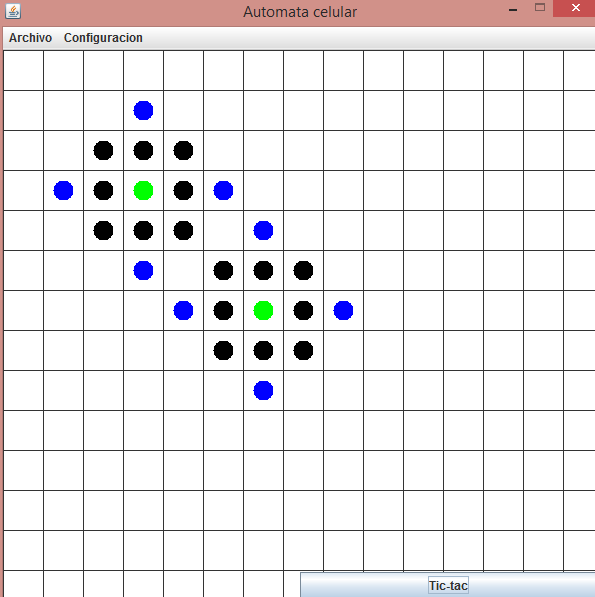


1. Realicen una prueba de aceptación para este método iniciando la aplicación, creando una nueva situación en el automata y abriendo el archivo unautomata.dat. Capturen imágenes significativas de estos resultados.

**Estado inicial antes de abrir el archivo.**



**Ahora abriremos el archivo que anteriormente guardamos**



## Implementando importar y exportar

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

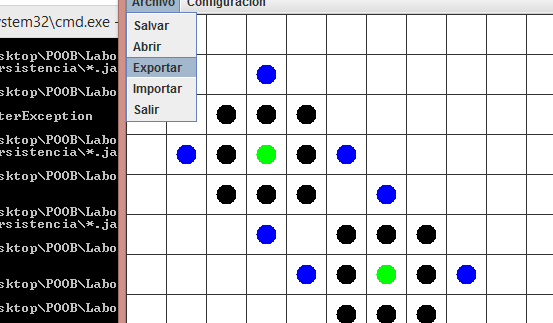
Estas operaciones nos van a permitir importar información de un automata desde un archivo de texto y exportarlo. Los nombres de los archivos de texto deben tener como apellido .txt

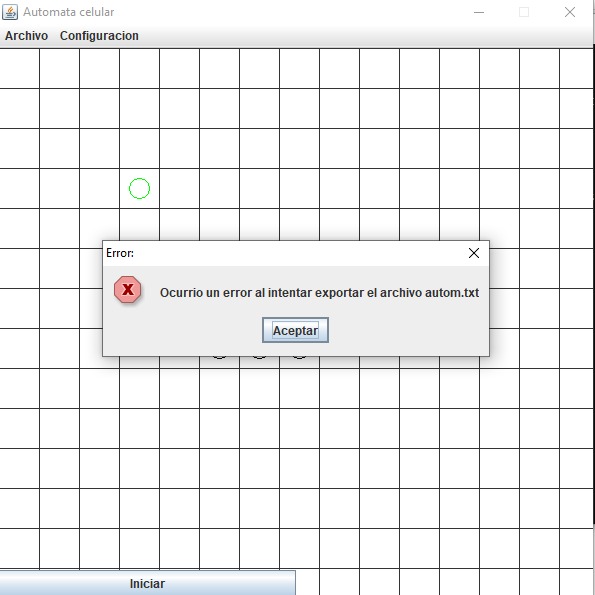
Los archivos texto tienen una línea de texto por cada elemento

En cada línea asociada un elemento se especifica el tipo y la posición. Cedula 20 20

Barrera 50 50

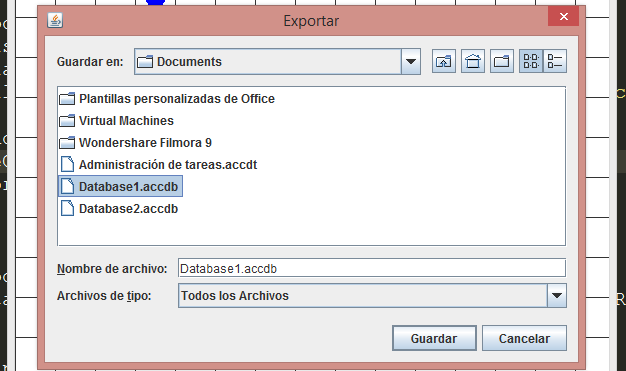
1. Construyan el método opcionExportar que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación y capturen una pantalla significativa.

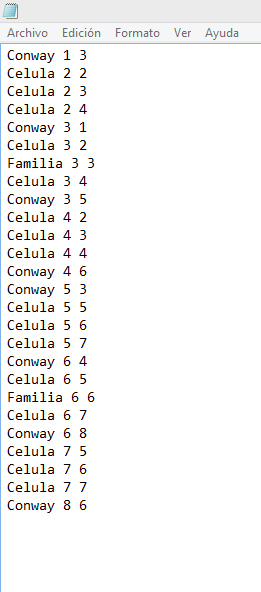


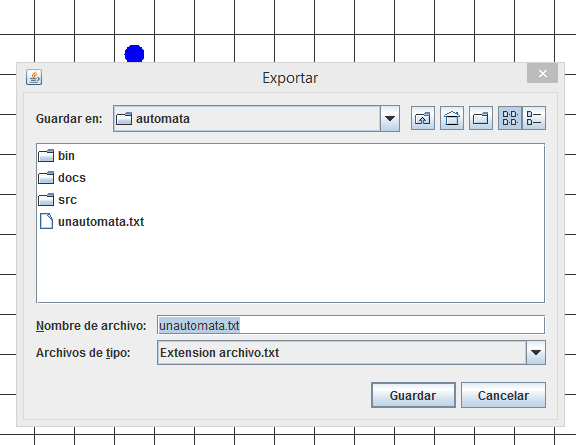


1. Construyan el método exporte que ofrece el servicio de exportar a un archivo texto, con el formato definido, el estado actual.



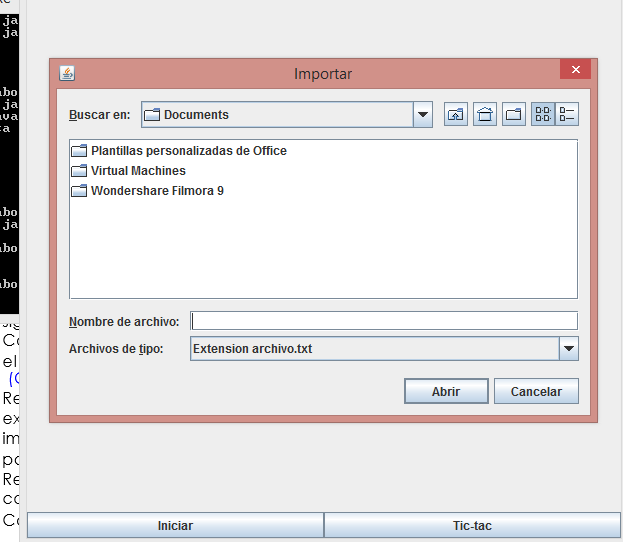


1. Realicen una prueba de aceptación de este método: iniciando la aplicación y exportando como unautomata.txt. Editen el archivo y analicen los resultados. ¿Qué pasó?



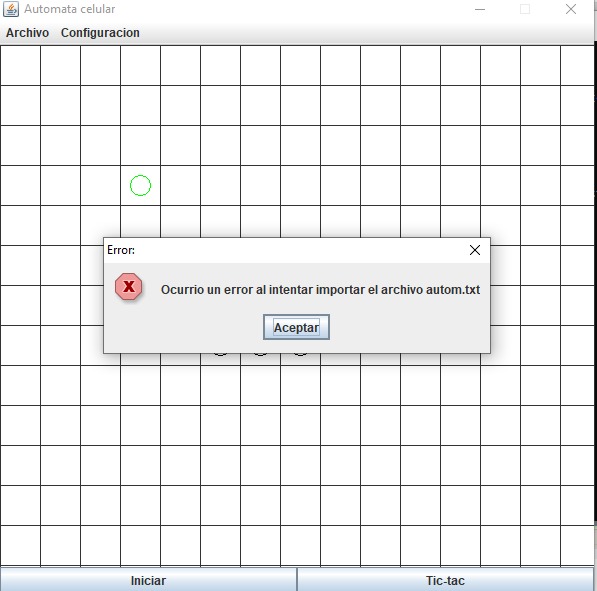
Si se vuelve a cambiar el autómata y se da exportar el archivo **txt** solo quedara con la configuración actual de la autómata celular

1. Construyan el método opcionImportar que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación y capturen una pantalla significativa.

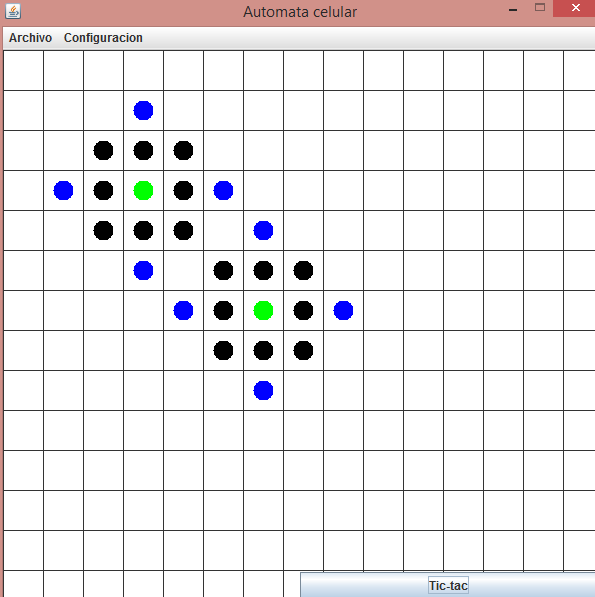
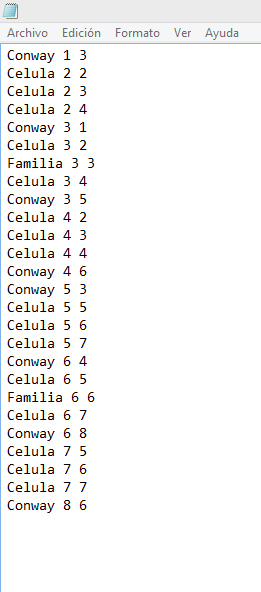


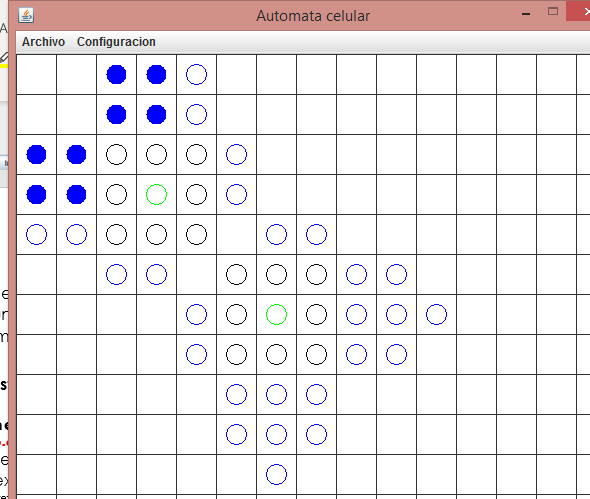
1. Construyan el método importe que ofrece el servicio de importar de un archivo texto con el formato definido. Por ahora sólo considere un mensaje de error general.

(Consulten en la clase String los métodos trim y split)

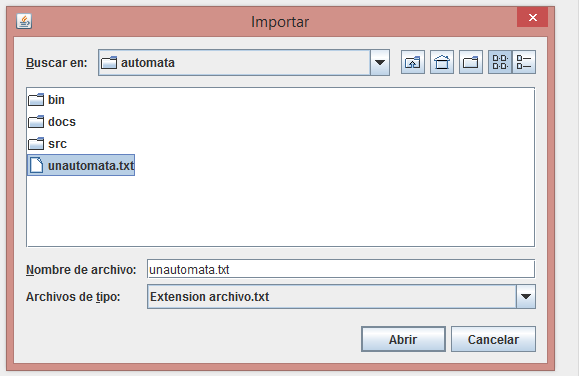
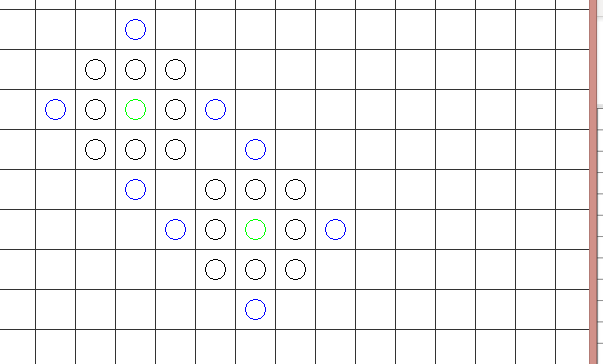


1. Realicen una prueba de aceptación de este par de métodos: iniciando la aplicación exportando a unautomata.txt. saliendo, entrando, creando un nuevo automata e importando el archivo otroautomata.txt.. ¿Qué resultado obtuvieron? Capturen la pantalla final.

Exportamos

Creamos una nueva celula

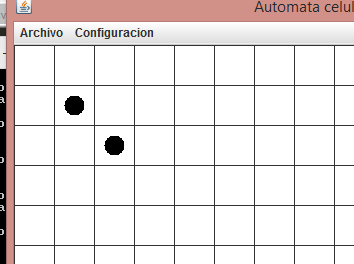
Luego importamos



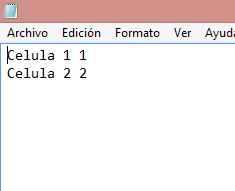
Al exportar solo el tipo y su ubicación, cuando se recupere la información la automata tendrá todos sus elementos que sean células sin ningún estado inicial

1. Realicen otra prueba de aceptación de este método escribiendo un archivo de texto correcto en unautomata.txt. e importe este archivo. ¿Qué resultado obtuvieron? Capturen la pantalla.

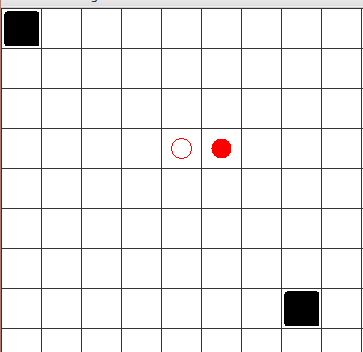
Creamos una nueva celula



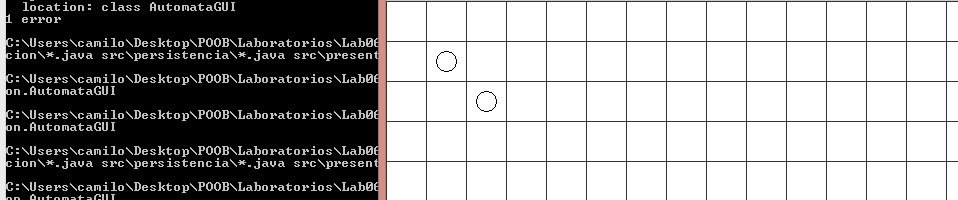
La exportamos al archivo txt y el resultado final es :



Creamos una nueva celula



Ahora importamos la primer celula que guardamos y el resultado es el siguiente.



## Analizando comportamiento

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

1. **Ejecuten la aplicación, den tres clics, salven a un archivo cualquiera y ábranlo. Describan el comportamiento**

Se guardó un archivo dat con los objetos, y luego se generó el autómata como fue salvada en el anterior archivo.

1. **Ejecuten la aplicación, tres clics, exporten a un archivo cualquiera e importen. Describan el comportamiento**

Se guardó un archivo txt con los atributos mínimos necesarios, y luego con esos datos mínimos necesarios se trata de recrear el objeto exportado anteriormente.

1. **¿Qué diferencias ven el comportamiento 1? y 2.? Expliquen los resultados.**

El primero quedara exactamente con las mismas características que tenía cuando salvo, el segundo no ya que puede que las células tomaran la misma posición, mas no significa que tomen el mismo estado.

## Perfeccionando salvar y abrir

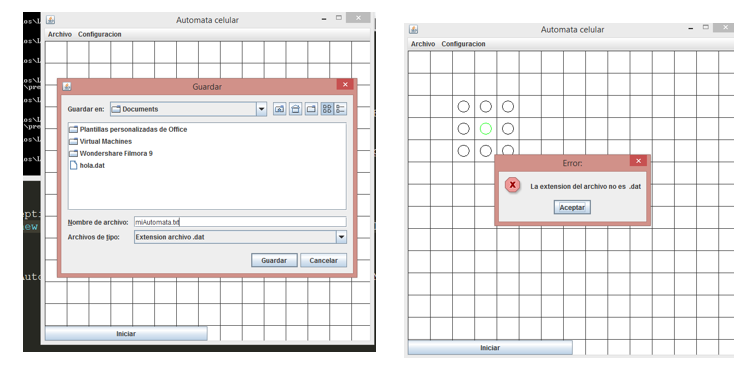
**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de abra y salve y renómbrenlos como abra01 y salve01
2. Perfeccionen el manejo de excepciones de los métodos abra y salve detallando los errores.

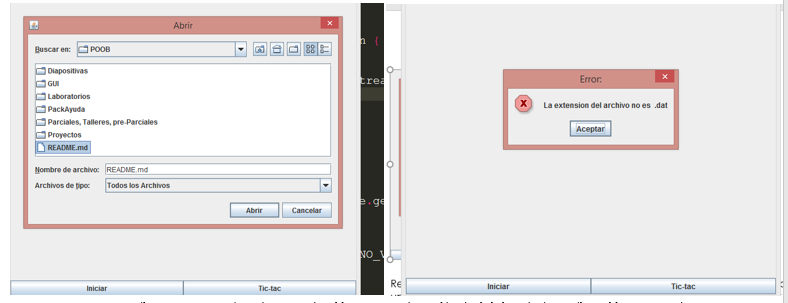
**Solo dejara salvar y abrir los archivos con extensiones .dat**

1. Realicen una prueba de aceptación para validar cada uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

**Salvar**



**abrir**



## Perfeccionando importar y exportar.

**[En lab06.doc, \*.asta , automataErr.txt \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe01

y exporte01

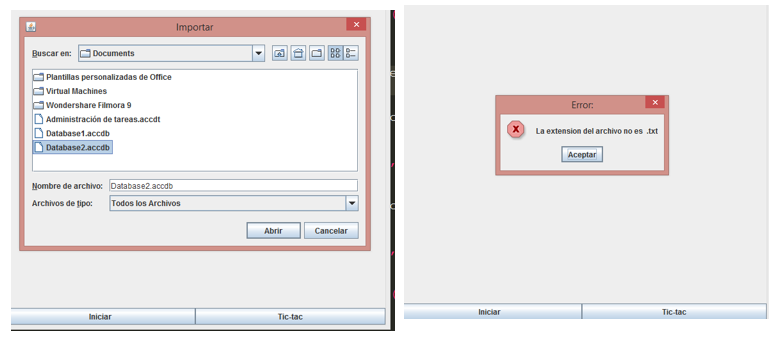
1. Perfeccionen el manejo de excepciones de los métodos importe y exporte detallando los errores.

**Solo dejara importar y exportar los archivos conextensiones .txt**

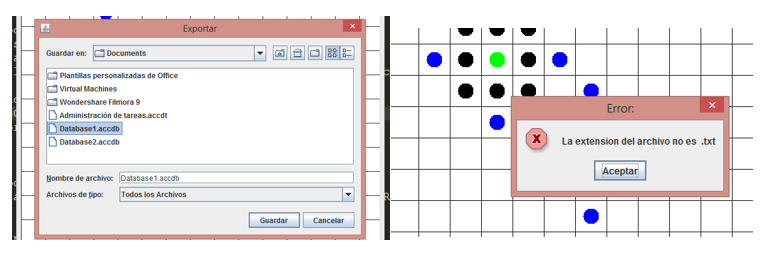
1. Realicen una prueba de aceptación para validar cada una de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

**Solo los deja manejar los archivos .txt**

**Importar**



**Exportar**



## Perfeccionando importar. Hacia un minicompilador.

**[En lab06.doc, \*.asta , automataErr.txt \*.java] [NO OLVIDEN BDD y MDD]**

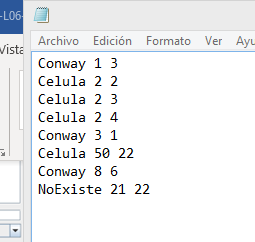
1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe02 y

exporte02

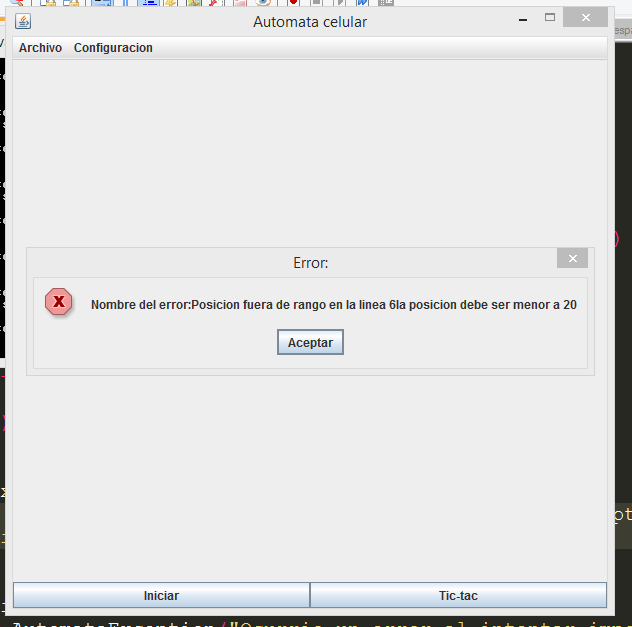
1. Perfeccionen el método **importe** para que, además de los errores generales, en las excepciones indique el detalle de los errores encontrados en el archivo (como un compilador) : número de línea donde se encontró el error, palabra que tiene el error y causa de error.
2. Escriban otro archivo con errores, llámelo automataErr.txt, para ir arreglándolo con ayuda de su “importador”. Presente las pantallas que contengan los errores.

**IMPORTAR**

**Para Importar, note que los índices se salen del tamaño de la automata , ya que la automata tiene un tamaño fijo de 20**



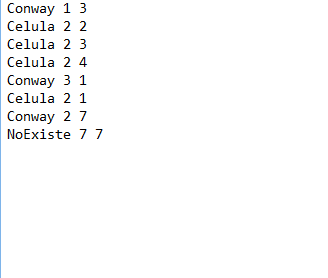
Al importar debe salir el siguiente mensaje:

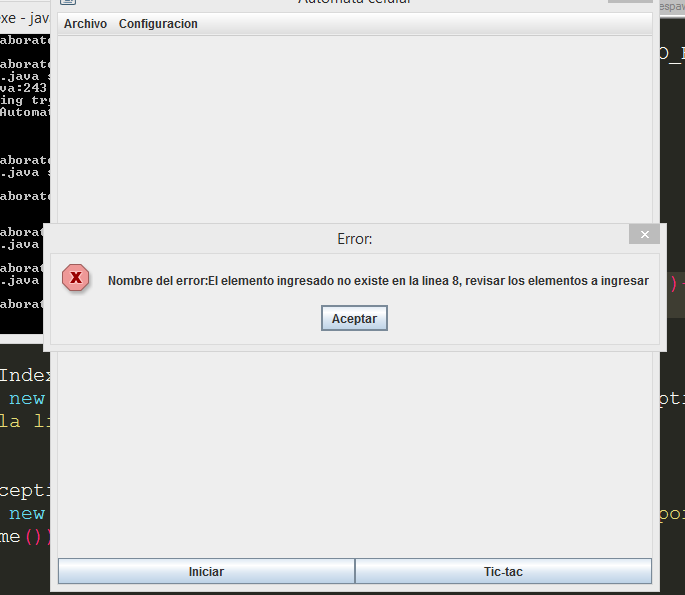


El objeto ingresado no existe

**Importar**

**Note que “NoExiste “es un elemento no definido en el programa**





## Perfeccionando importar. Hacia un minicompilador flexible.

**[En lab06.doc, \*.asta , automataFlex.txt \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe03 y

exporte03

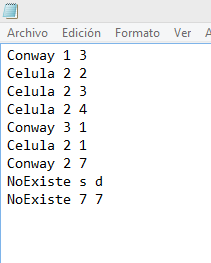
1. Perfeccionen los métodos importe y exporte para que pueda servir para cualquier tipo de elementos creados en el futuro

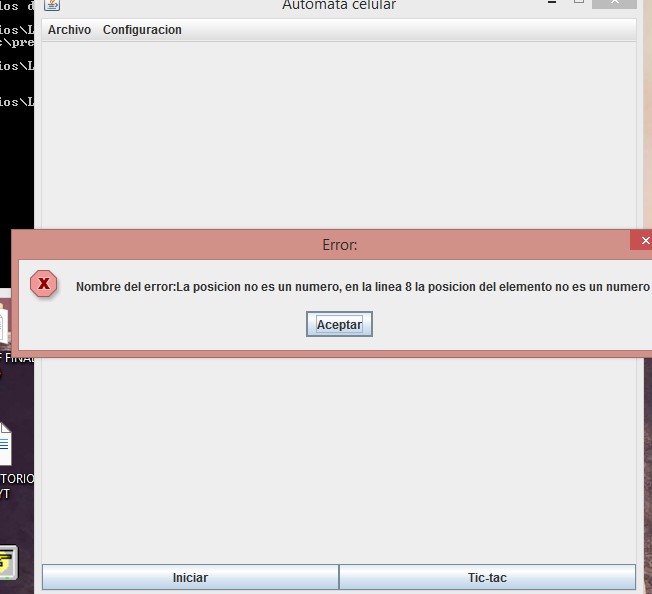
(Investiguen cómo crear un objeto de una clase dado su nombre)

1. Escriban otro archivo de pruebas, llámelo automataErrG.txt, para probar la flexibilidad. Presente las pantallas que contenga un error significativo.

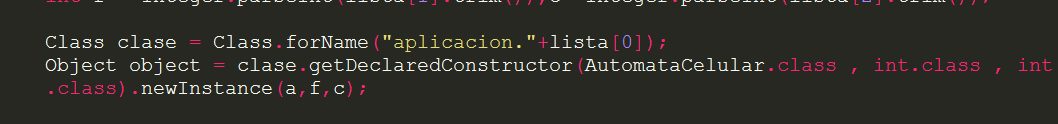
**Ingresar una posición que no puede convertirse a un numero**

**Cuando exportemos el siguiente archivo**





**cómo crear un objeto de una clase dado su nombre**



Con las anteriores líneas de código, se busca la clase que pertenezca a **aplicación**

y como” un **modelo lógico de bases de datos” ,** decimos la clase (Tabla), sus atributos y después que le vamos a ingresar esos atributos. Especificando su paquete especifico

# 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?**

Pensar en como ir mejorando cada una de las 4 funcionalidades de abrir, guardar, exportar, importar.

**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.