**PROGRAMACIÓN ORIENTADA A OBJETOS**

**2020-01**

**Laboratorio 6/6**

**DESARROLLO**

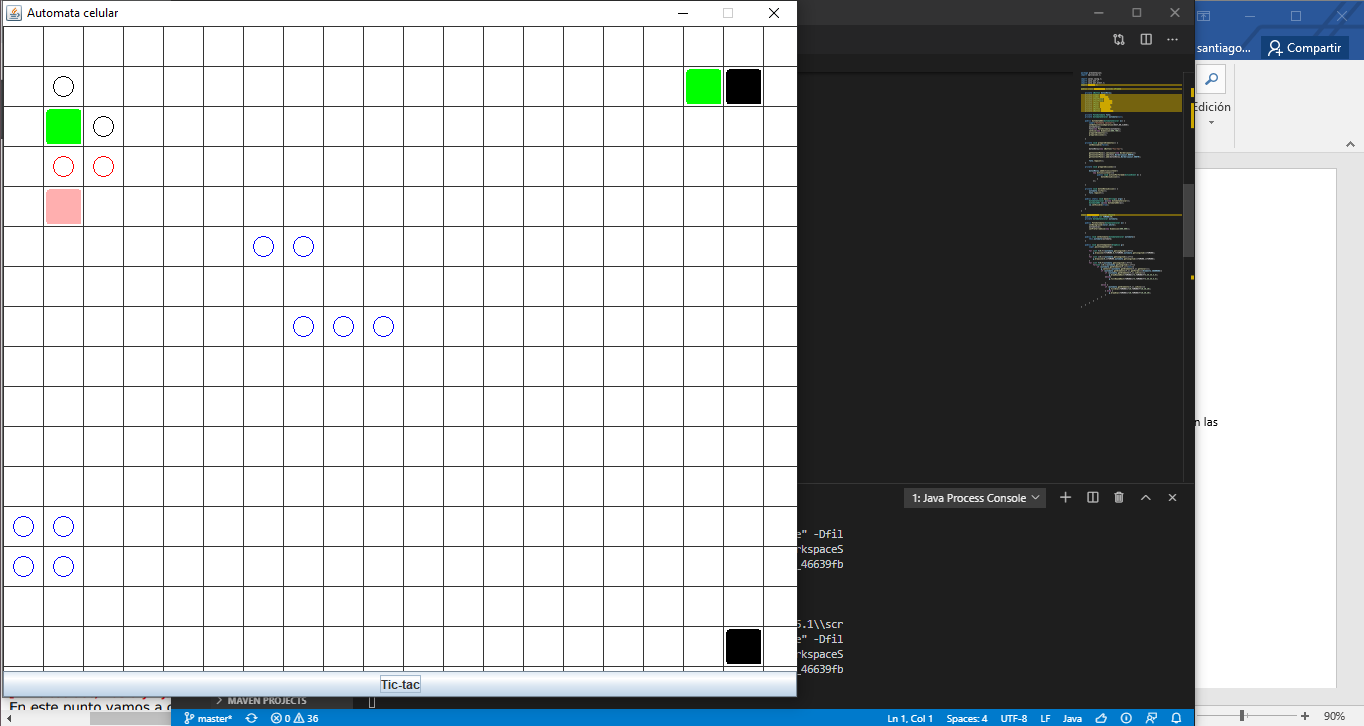
**Preparando**

En este laboratorio vamos a extender el proyecto **autómata** 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 o3 y preparen el ambiente para trabajar desde **CONSOLA**

**Carpeta Lab06**

1. Ejecuten el programa, revisen la funcionalidad



**Creando la maquina**

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

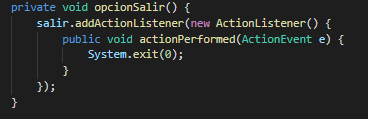
1. **MODELO:** Preparen los métodos correspondientes a reiniciar y a las cuatro opciones básicas de entreda-salida (salve, abra, exporte, importe). Los métodos deben simplemente propagar una automataExcepcion con el mensaje: “Opción … en construcción”. Los métodos de entrada salida deben tener un parámetro Fila.
2. **VISTA**: Construyan un menú barra que ofrezca, además de las opciones básicas de entreda-salida, las opciones estándar de iniciar y salir. Para esto creen el método preprareElementosMenu. Capturen la pantalla correspondiente.

1. **CONTROLDAOR:** Constritan los controladores correspondientes a estas acciones. Para esto creen el método **prepareAccionesMenu** y los métodos base del controlador (opcionSalver, opcionAbrir, opcionExportar, opcionImportar, opcionReiniciar, opcionCerrar).

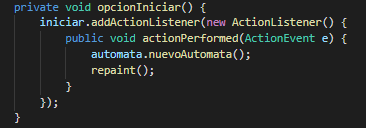
**Implementando salir e iniciar**

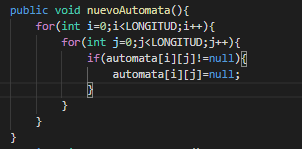
Las opciones salir e iniciar van a ofrecer los dos servicios estándar de las aplicaciones. El primero n requiere ir a capa de aplicación y el segundo sí.

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 autómata. Capturen una pantalla significativa



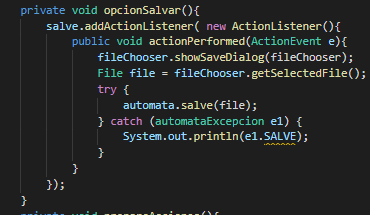


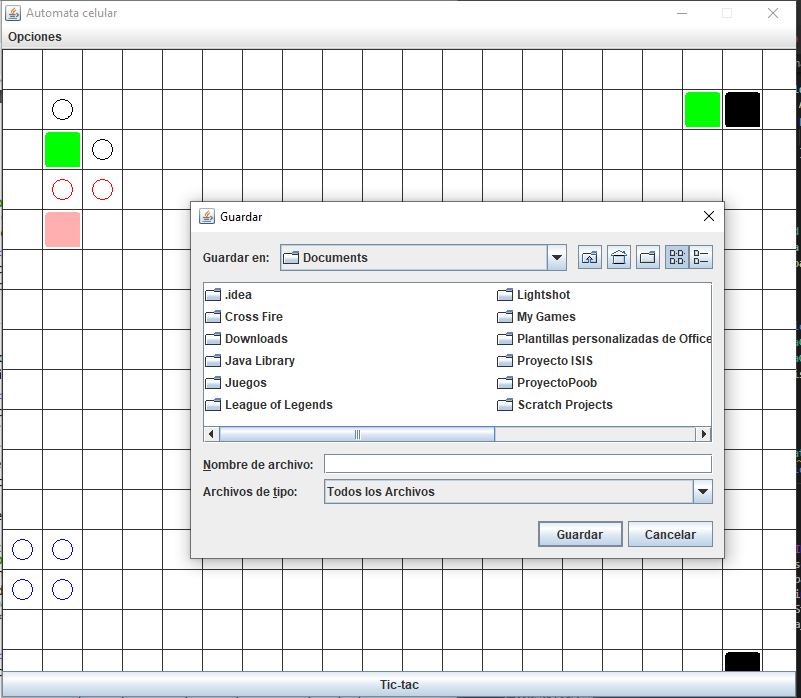
**Implementando salvar y abrir**

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

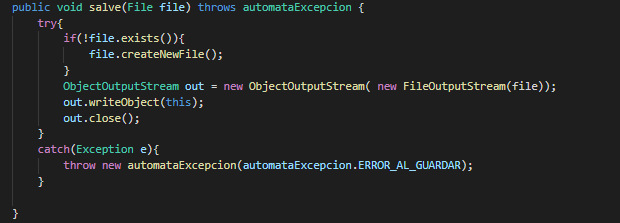
1. Construyan el método opcionSalvar que une de forma adecuada la capa de presentación con la capa de aplicación. Usen un FileChooser y atiendan la excepción.

Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.

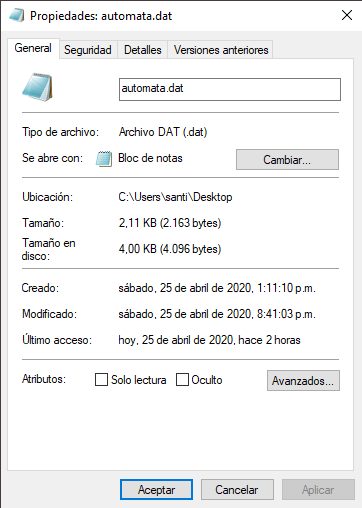




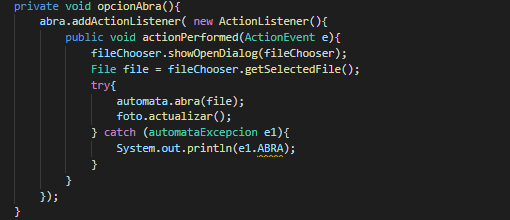
1. Construyan el método salve que ofrece el servicio de guardar en un archivo el estado actual del autómata

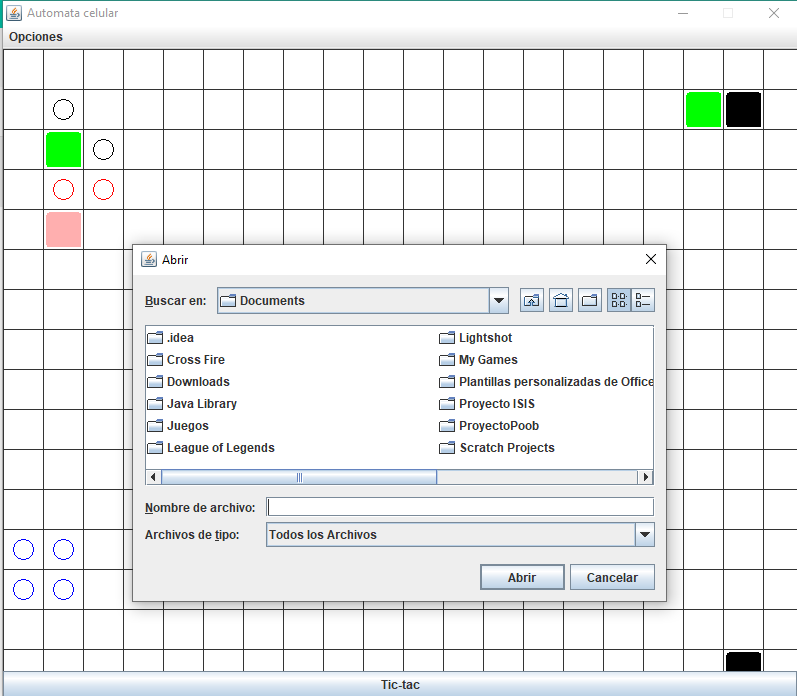


1. Validen este método opcionAbrir guardando el autómata inicial después de dos clics como un autómata.dat. ¿El archivo se creó en el disco? ¿Cuánto espacio ocupa?

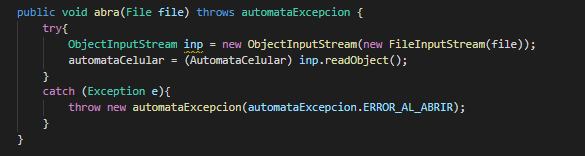


1. Construyan el método opcionAbrir que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.



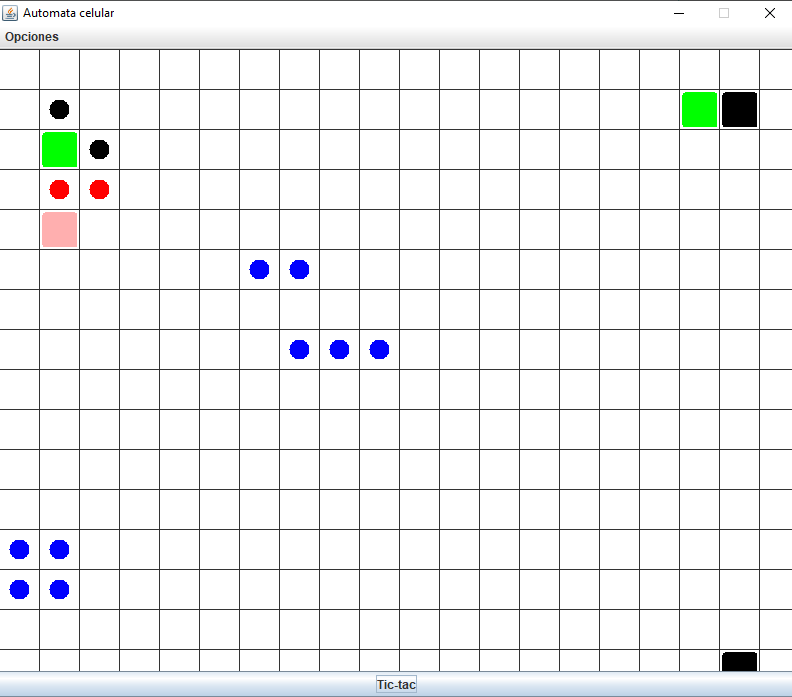


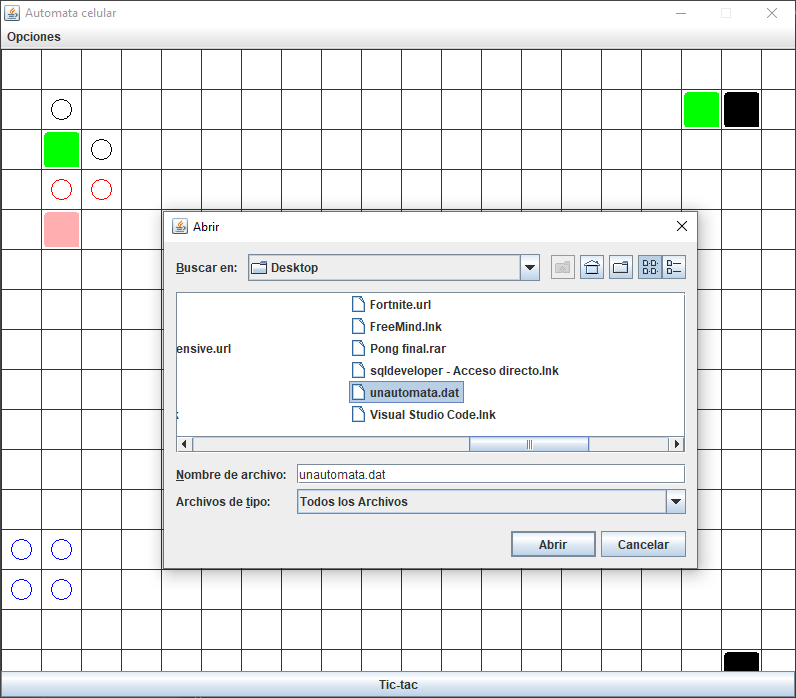
1. Construyan el método abra que ofrece el servicio de leer un autómata de un archivo. Por ahora para las excepciones sólo considere un mensaje de error general.



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

Situación de prueba:

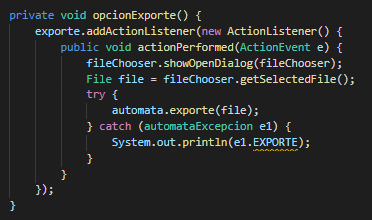




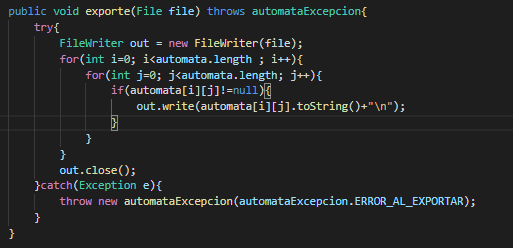
**Implementando importa y exporta**

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

1. Construya 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.

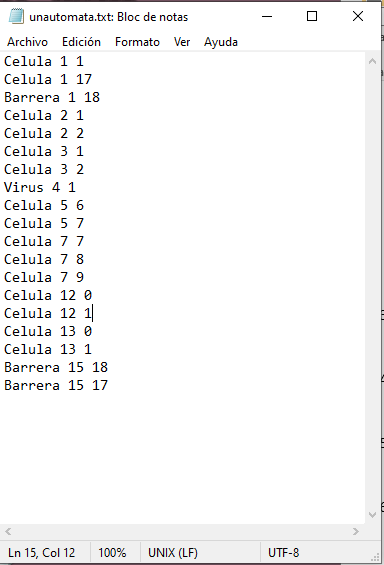


1. Construyan el método **exporte** que ofrece el servicio de exportar a un archivo texto, con el formato definido, el estado actual.

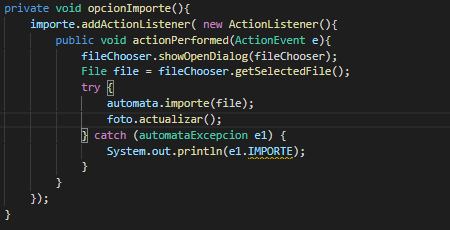


1. Realicen una prueba de aceptación de este método: iniciando la aplicación y exportando como **unautomata.txt**. Editen el archivo y analicen los resultados. ¿Qué pasó?

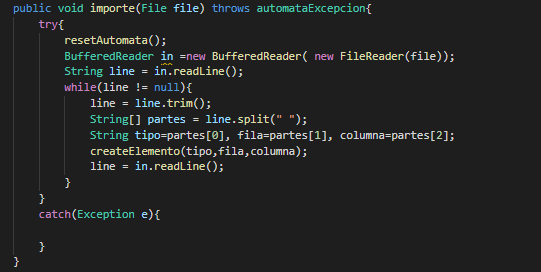
El archivo permite ser editar, nos muestra el tipo con su respectiva fila y columna

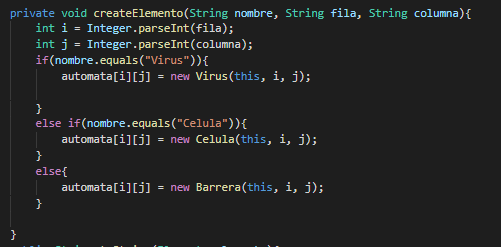


1. Construyan el método **opcionImportar** que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación y capturen una pantalla significativa.



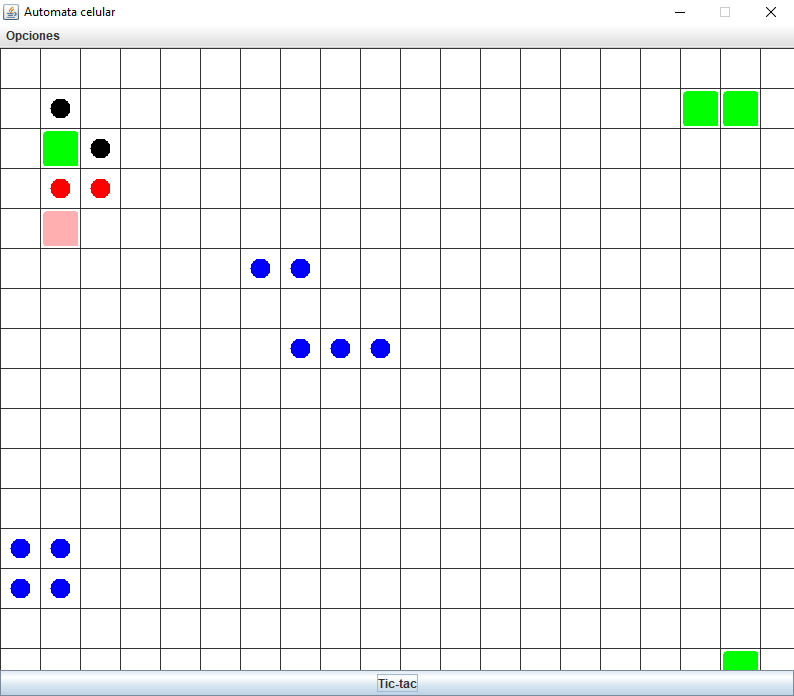
1. Construyan el método **importe** que ofrece el servicio de importar de un archivo texto con el formato definido. Por ahora sólo considere un mensaje de error general.

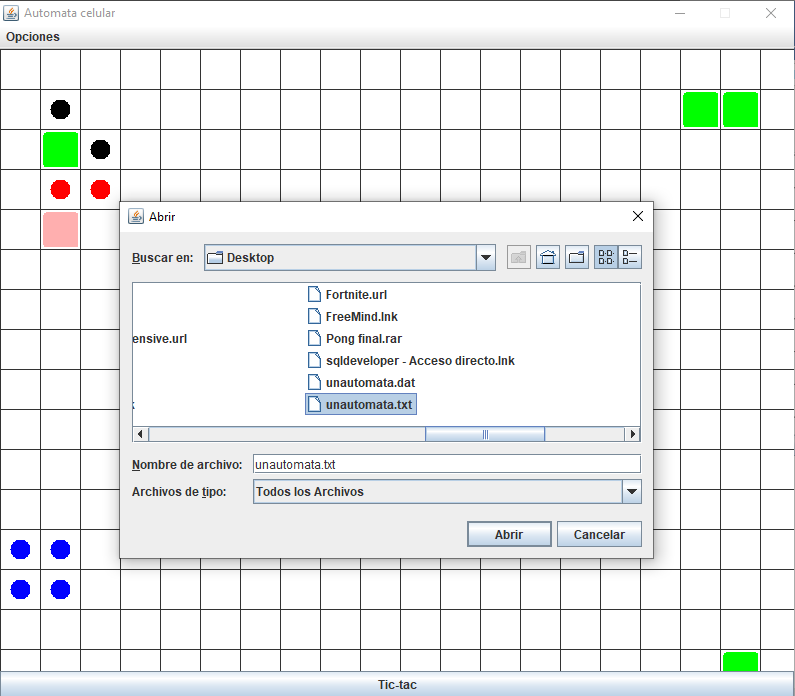




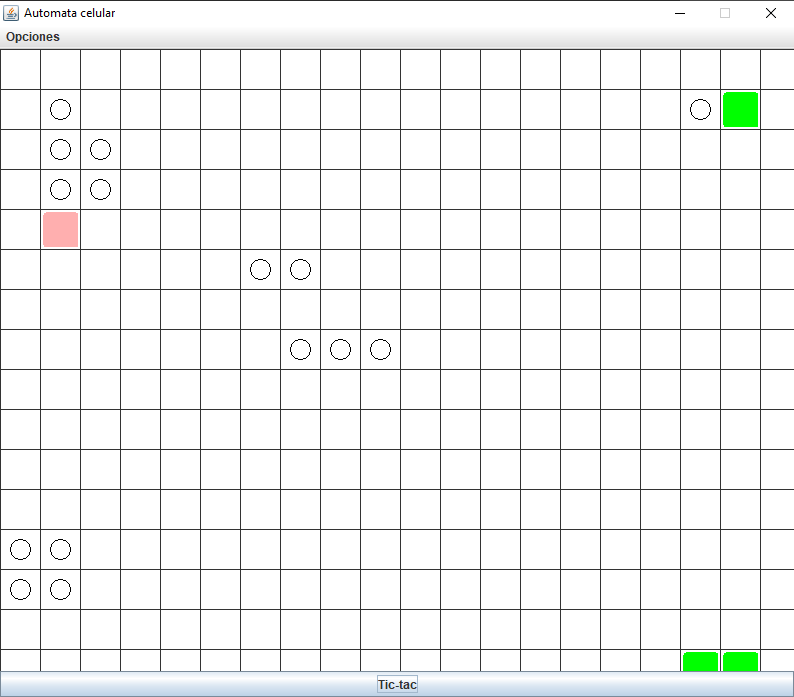
1. Realicen una prueba de aceptación de este par de métodos: iniciando la aplicación exportando a **unautomata.txt**. ¿Qué resultado obtuvieron? Capturen pantalla.

Caso inicio:

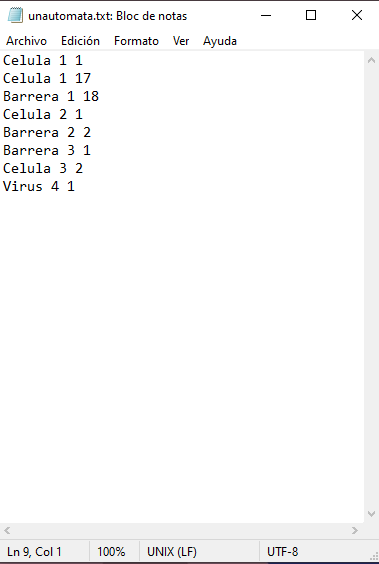


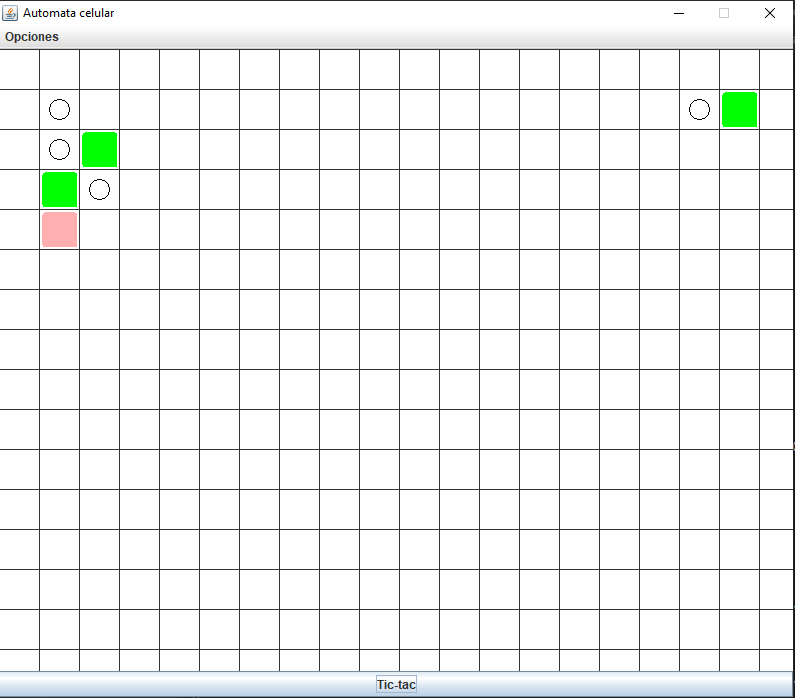


Al abrir:



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





**Analizando comportamiento**

1. Ejecuten la aplicación, den tres clics, salven a un archivo cualquiera y ábranlo. Describan el comportamiento.
2. Ejecuten la aplicación, tres clics, exporten a un archivo cualquiera e importen. Describan el comportamiento.
3. ¿Qué diferencias ven el comportamiento 1 y 2? Expliquen los resultados.

**Perfeccionando salvar y abrir**

1. Copien las versiones actuales de abra y salve y renómbrenlos como abrea01 y salva 01
2. Perfeccionen el manejo de excepciones de los métodos abra y salve detallando los errores.
3. Realicen una prueba de aceptación para validar cada uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

**Perfeccionando importar y exportar.**

1. Copien las versiones actuales de **importe** y **exporte** y renómbrenlos como **importe01** y **exporte01**
2. Perfeccionen el manejo de excepciones de los métodos **importe** y **exporte** detallando los errores.
3. Realicen una prueba de aceptación para validad cada uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

**Perfeccionando importar. Hacia un minicompilardor.**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe02 y exporte02
2. Perfeccionen el manejo de excepciones de los métodos importe y exporte detallando los errores.
3. Realicen una prueba de aceptación para validad cada uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

**Perfeccionando importar. Hacia un minicompilador flexible.**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe03 y exporte03
2. Perfeccionen los métodos importe y exporte para que pueda servir para cualquier tipo de elemento creados en el futuro
3. Escriban otro archivo de pruebas, llámelo automataErrG.txt para probar la flexibilidad. Presente las pantallas que contenga un error significativo.

**RETROSPECTIVA**

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

Santiago – 50 horas

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

Creo que está completo el laboratorio.

1. Considerando la práctica XP del laboratorio ¿por qué consideran que es importante?

Repartir bien la carga en el tiempo que se nos da.

1. ¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue si mayor problema? ¿Qué hicieron para resolverlo?

El mayor logro que tuve fue poder implementar de una manera correcta el ObjectImputStream. Este también fue el mayor problema dado que no sabia como implementarlo y tarde mucho hasta poder lograr que abriera un archivo.

1. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?

Estoy solo en el grupo.