**Laboratorio 6**

**Santiago Gualdron- Sofia Gil**

**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 en la clase fachada del modelo los métodos correspondientes a las cuatro optiones básicas de entrada-salida (open, save, import y export). Los métodos deben simplemente propagar una gardenException con el mensaje de “Opción nombreOpción en construcción. Archivo nombreArchivo”. Los métodos deben tener un parámetro File.

Imagen que contiene Tabla

Descripción generada automáticamente

**VISTA :** Construyan un menú barra que ofrezca, además de las opciones básicas de entrada-salida, las opciones estándar de nuevo y salir (Nuevo, Abrir, Guardar como, Importar, Exportar como, Salir). No olviden incluir los separadores. Para esto creen el método prepareElementsMenu. Capturen la pantalla del menú.

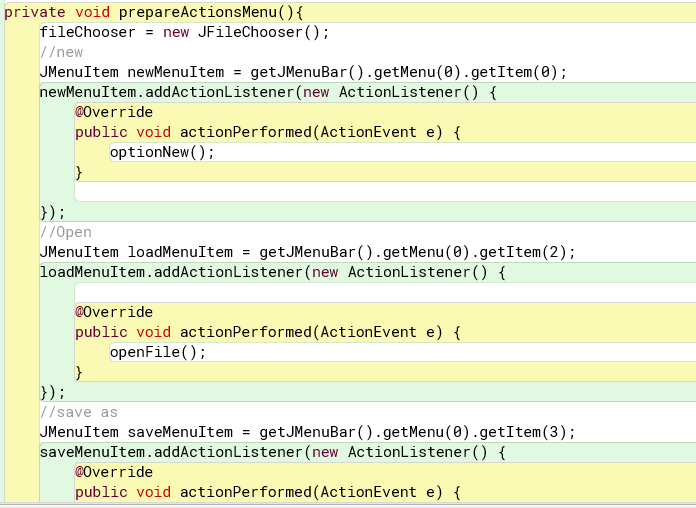
Texto

Descripción generada automáticamente

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

**CONTROLADOR**: Construyan los oyentes correspondientes a las seis opciones. Para esto creen el método prepareActionsMenu y los métodos base del controlador (optionOpen, optionSave, optionImport, optionImport, optionNew, optionExit). En las opciones que lo requieran usen un FileChooser y atiendan la excepción. Estos métodos llaman el método correspondiente de la capa de dominio que por ahora sólo lanza una excepción. Ejecuten las diferentes acciones del menú y para cada una de ellas capture una pantalla significativa.



Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Gráfico

Descripción generada automáticamente con confianza mediaGráfico

Descripción generada automáticamente con confianza baja

Gráfico

Descripción generada automáticamente con confianza mediaGráfico

Descripción generada automáticamente

Gráfico, Gráfico de barras

Descripción generada automáticamente

**Implementando salir y nuevo**

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

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

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

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

2. Construyan el método optionNew que crea un nuevo garden. Capturen una pantalla significativa.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**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 del garden como objeto. Los nombres de los archivos deben tener como extensión .dat.

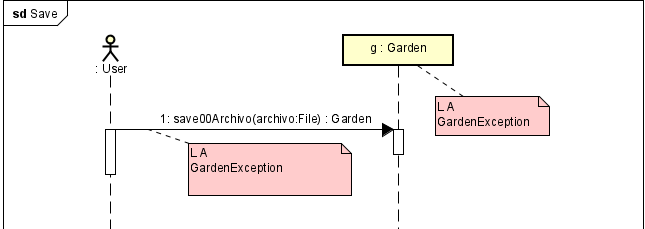
1. Copien las versiones actuales de open y save y renómbrenlos como open00 y save00

Tabla

Descripción generada automáticamente con confianza baja

2. Construyan el método save que ofrece el servicio de guardar en un archivo el estado actual de garden. Por ahora para las excepciones sólo consideren un mensaje de error general. No olviden diseño y pruebas de unidad.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente 

Interfaz de usuario gráfica, Texto, Aplicación, Sitio web

Descripción generada automáticamente

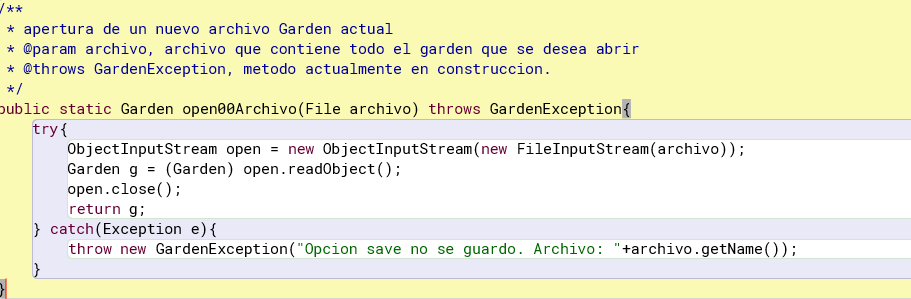
3. Validen este método guardando el estado obtenido después de dos clics como onegarden.dat. ¿El archivo se creó en el disco? ¿Cuánto espacio ocupa?

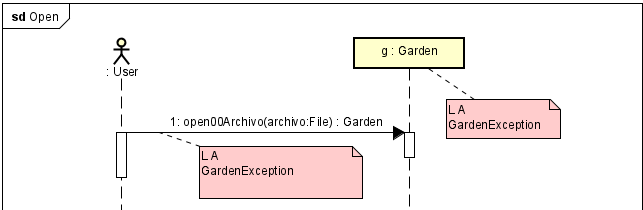
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Se guardo en la carpeta donde tenemos los archivos del laboratorio y pesa 4 KB.

4. Construyan el método open que ofrece el servicio de leer un garden de un archivo. Por ahora para las excepciones sólo consideren un mensaje de error general. No olviden diseño y pruebas de unidad.

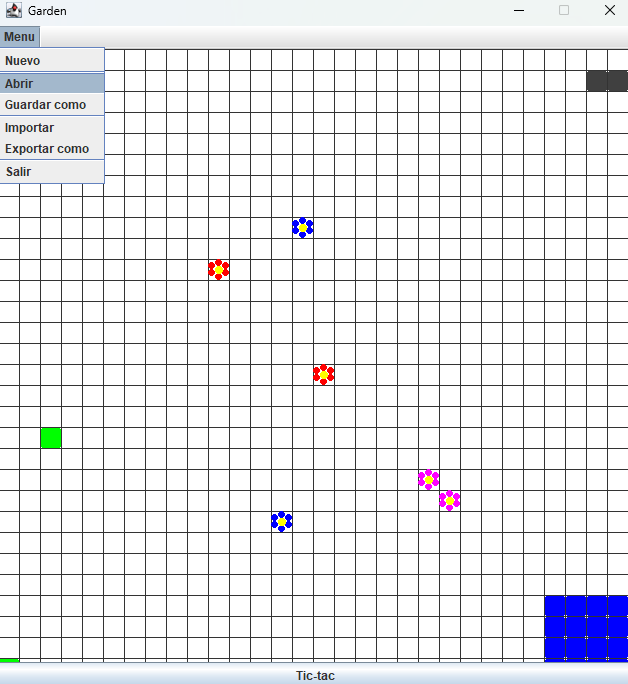






5. Realicen una prueba de aceptación para este método iniciando la aplicación, creando un nuevo estado y abriendo el archivo onegarden.dat. Capturen imágenes significativas de estos resultados.

1. Abrimos la aplicación



2. Creamos un nuevo estado

Gráfico, Gráfico de barras

Descripción generada automáticamente

3. Abrimos el archivo onegarden.dat que teníamos guardado

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

**Implementando importar y exportar**

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

Estas operaciones nos van a permitir importar información del garden desde un archivo de texto y exportarlo. Los nombres de los archivos de texto deben tener como extensión .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.

Flower 10 10

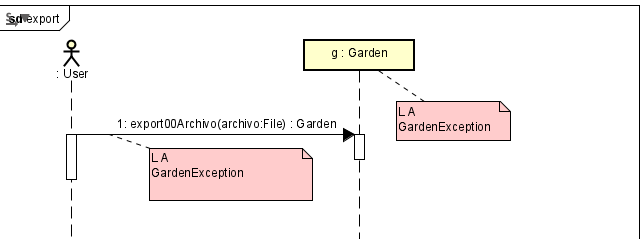
Sand 20 20

1. Copien las versiones actuales de import y export y renómbrenlos como import00 y export00.



2. Construyan el método export que ofrece el servicio de exportar a un archivo texto, con el formato definido, el estado actual. Por ahora para las excepciones sólo consideren un mensaje de error general. No olviden diseño y pruebas de unidad.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente



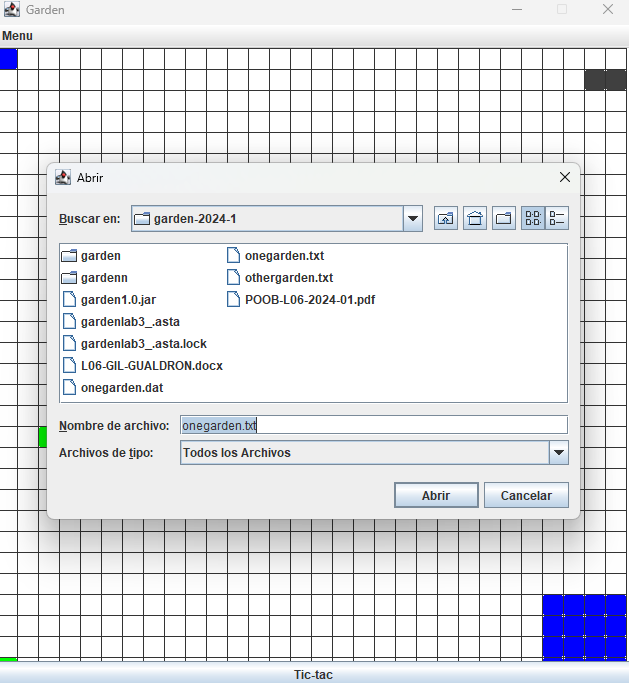
3. Realicen una prueba de aceptación de este método: iniciando la aplicación y exportando como onegarden.txt. Editen el archivo y analicen los resultados. ¿Qué pasó?

1. Abrimos la aplicación

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

2. hacemos un click y exportamos

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

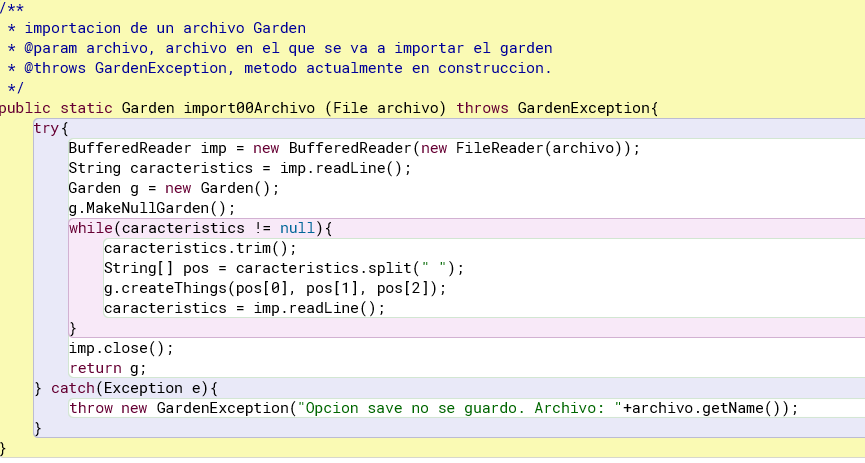
Descripción generada automáticamente

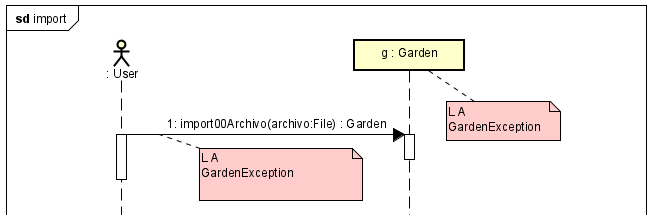
Lo que pasa que el archivo que se generó nos dice las posiciones de donde se habia exportado el Garden anterior para seguir con la contunacion de la aplicación.

4. Construyan el método import que ofrece el servicio de importar de un archivo texto con el formato definido. Por ahora sólo considere un mensaje de error general. No olviden diseño y pruebas de unidad. (Consulten en la clase String los métodos trim y split)

Split: Divide una cadena de texto en subcadenas basándose en uno o más caracteres separadores.

Trim: Este método elimina caracteres en blanco iniciales y finales de una cadena de texto.

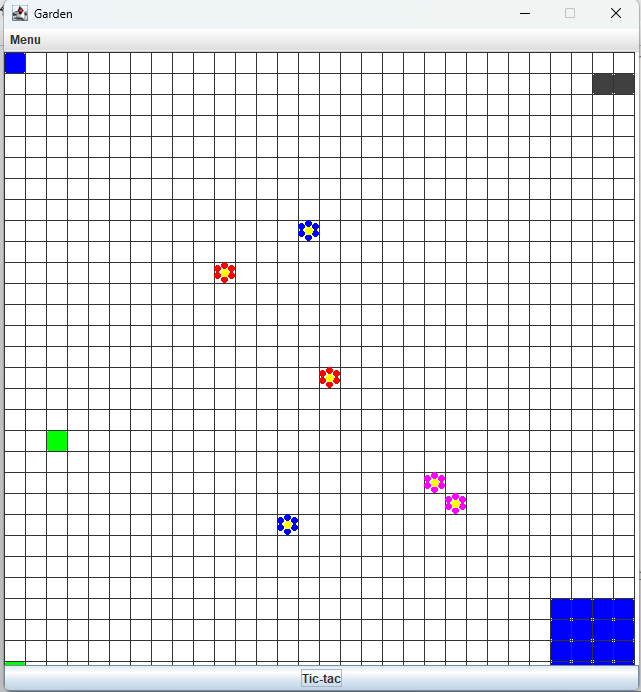






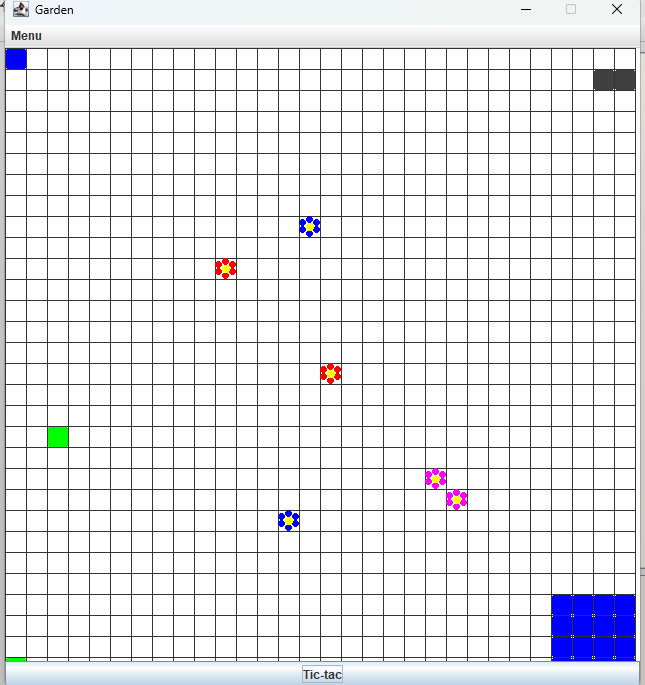
5. Realicen una prueba de aceptación de este par de métodos: iniciando la aplicación exportando a onegarden.txt. saliendo, entrando, creando una nueva e importando el archivo othergarden.txt. ¿Qué resultado obtuvieron? Capturen la pantalla final.

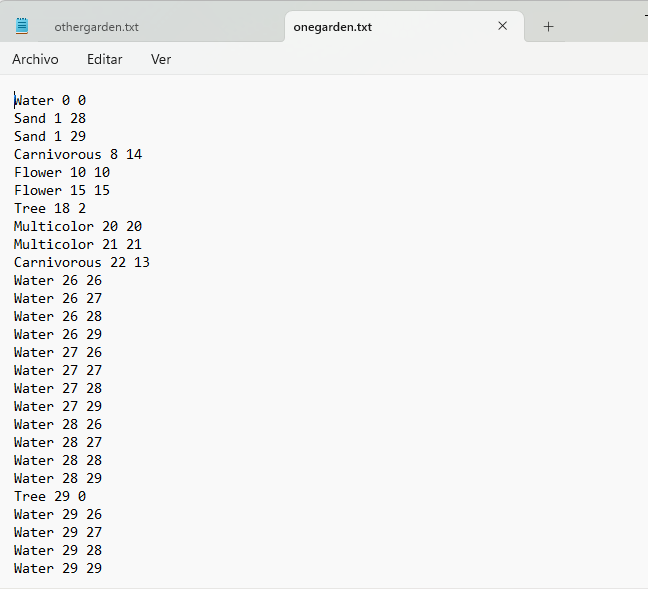
1. Abrimos la aplicación



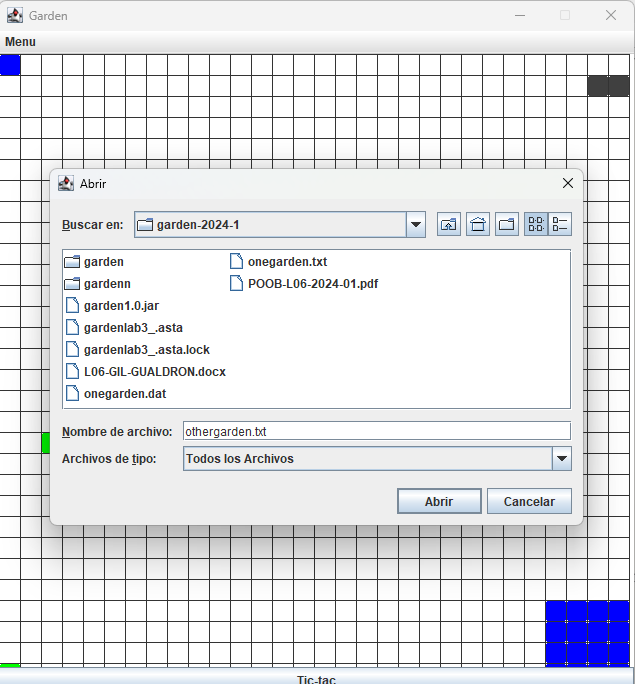
2. exportamos onegarden.txt sin cambiar el estado

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

Descripción generada automáticamente 



3. creamos un nuevo estado e importamos el archivo

Gráfico, Gráfico de dispersión

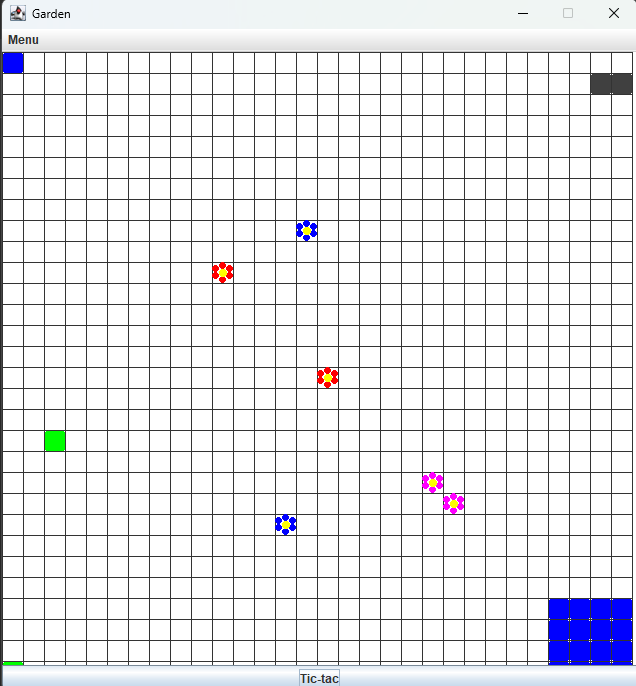
Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

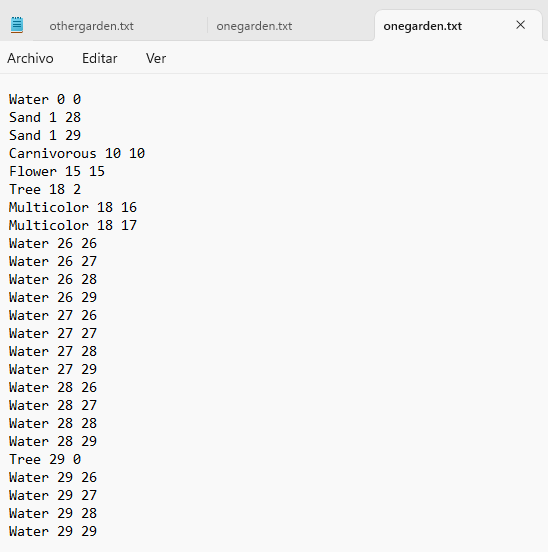
Descripción generada automáticamente

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

1. Abrimos la aplicación



2. Escribimos un archivo txt correcto, donde obtuvimos las posiciones de las flores y los objectos en los lugares establecidos.

Gráfico

Descripción generada automáticamente

**Analizando comportamiento**

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

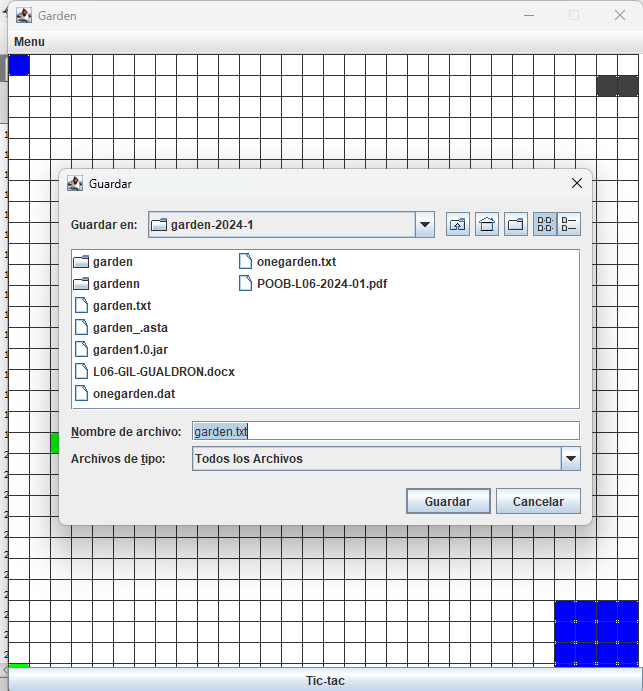
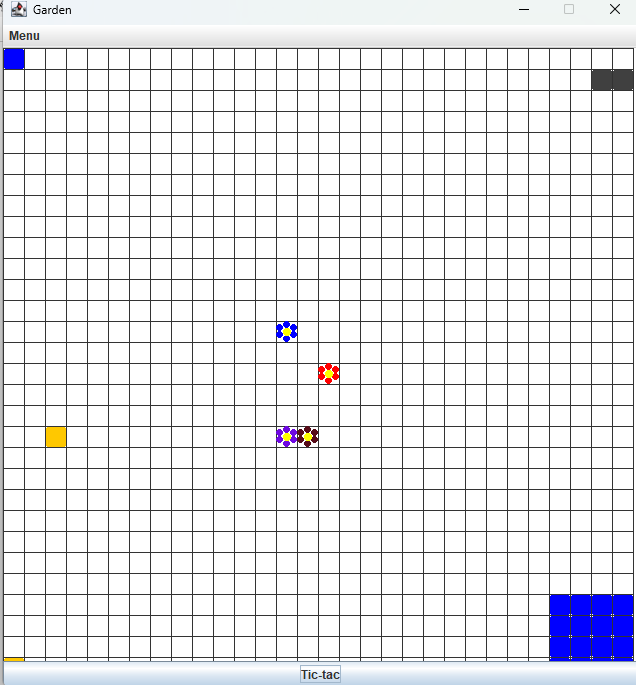
1. Ejecuten la aplicación, den tres clics, salven a un archivo cualquiera y ábranlo. Describan el comportamiento.

Ejecutamos la aplicación.

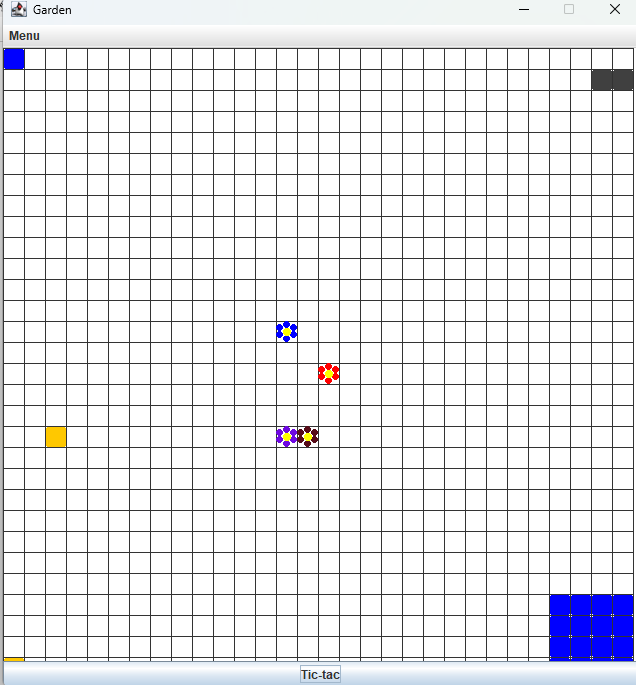
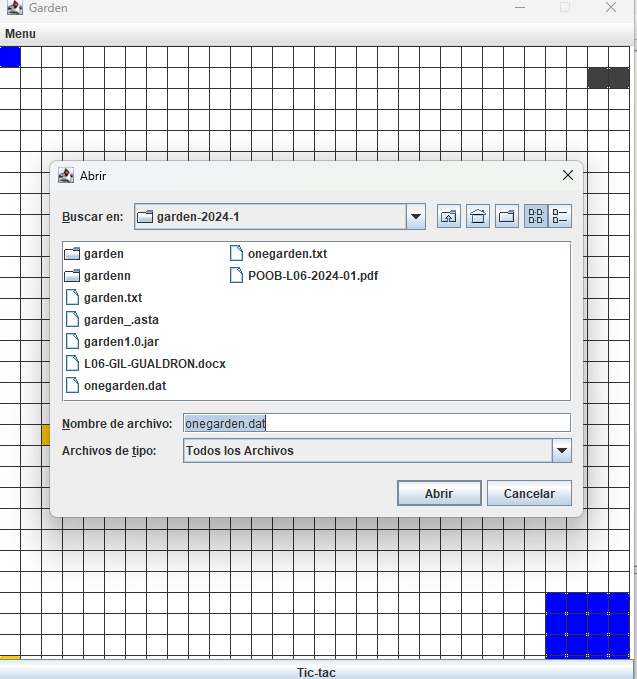
Gráfico

Descripción generada automáticamente

Después de 3 clics. Salvamos el archivo garden.txt

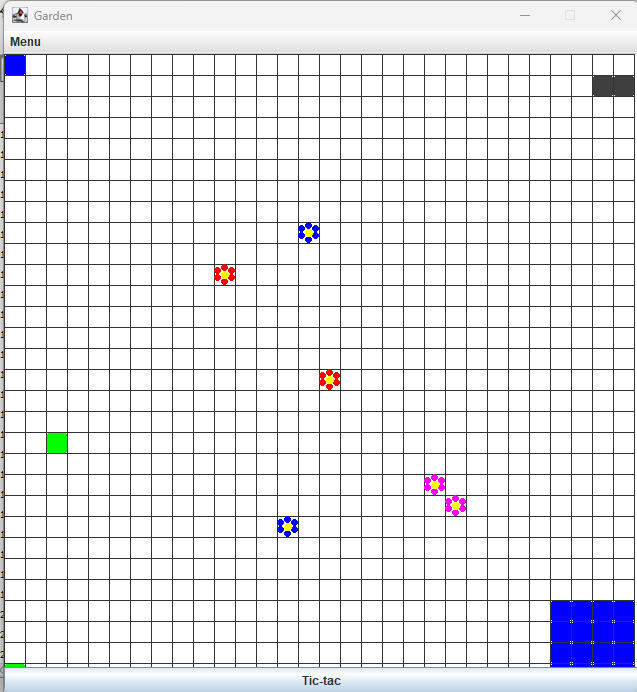


Abrimos el archivo de onegarden.dat

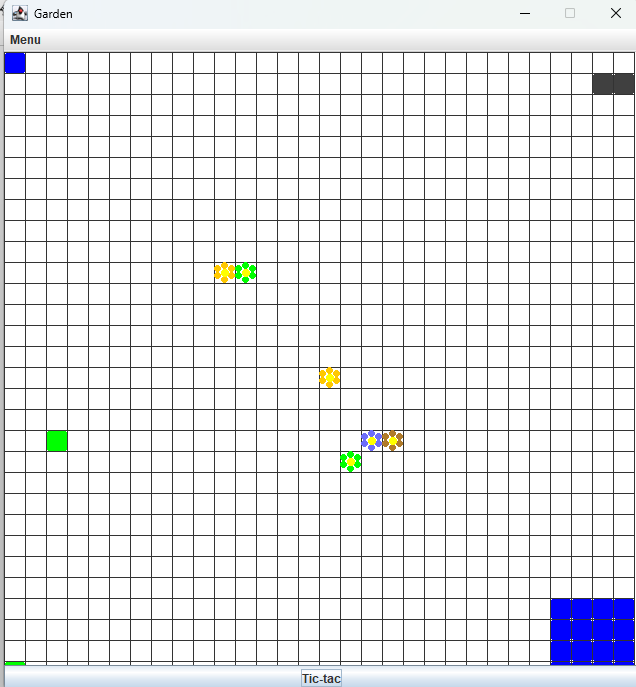


2. Ejecuten la aplicación, tres clics, exporten a un archivo cualquiera e importen. Describan el comportamiento

Ejecutamos la aplicación

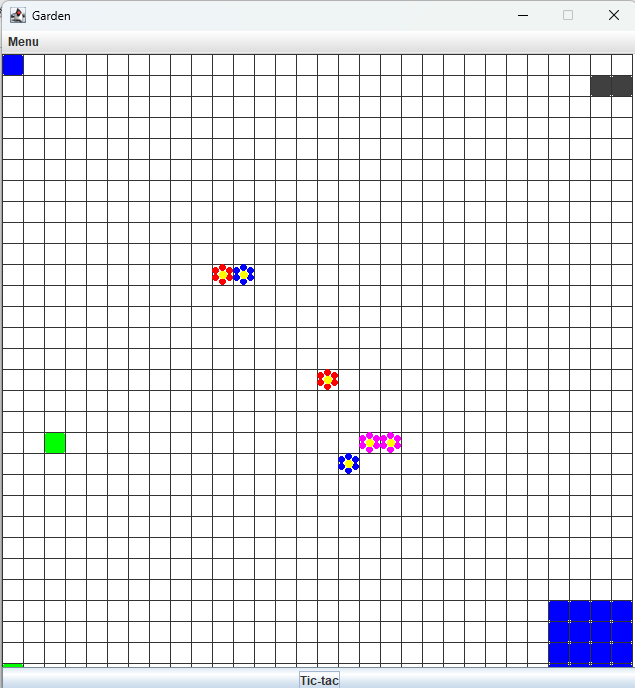
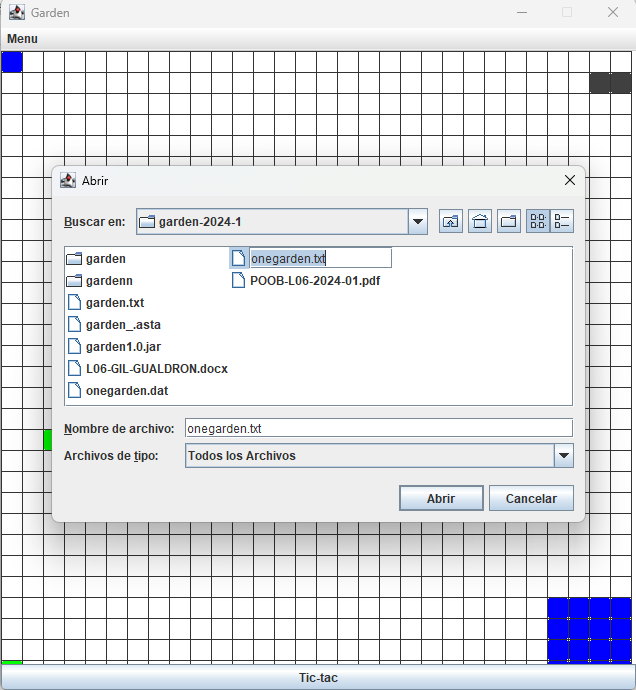


Después de 3 clics Exporten un archivo

Tabla

Descripción generada automáticamente

Importamos el archivo



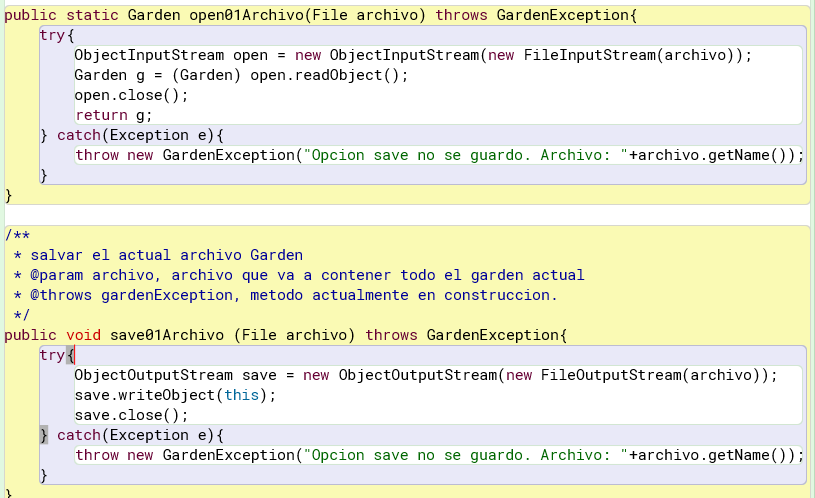
3. ¿Qué diferencias ven el comportamiento 1? y 2.? Expliquen los resultados.

Guardar y abrir están más relacionados con la acción de almacenar y acceder a datos dentro de un sistema modo (.dat), mientras que exportar e importar se refieren a mover datos entre diferentes sistemas modo (.txt).

**Perfeccionando salvar y abrir**

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

1. Copien las versiones actuales de open y save y renómbrenlos como open01 y save01



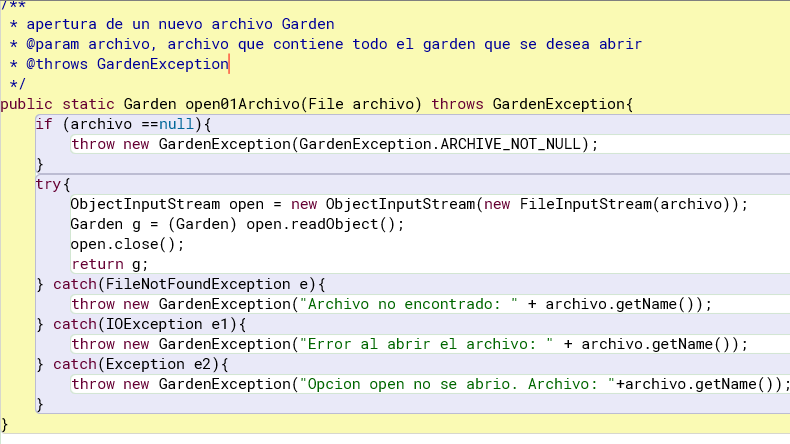
2. Perfeccionen el manejo de excepciones de los métodos open y save detallando los errores. No olviden pruebas de unidad.

1.Si el archivo no es encontrado para Open

2. Si hay un error al abrir un archivo para Open

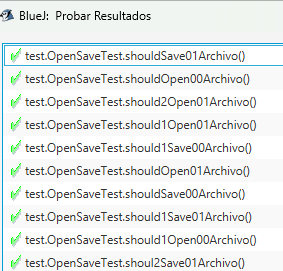
3. Si no se puede guardar un archivo para Save

4. Si hay un error al salvar el archivo para Save



Tabla

Descripción generada automáticamente con confianza media



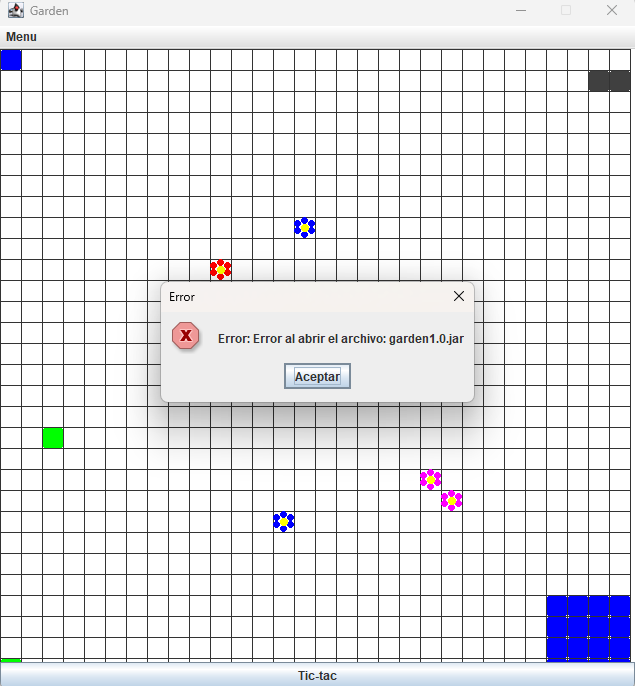
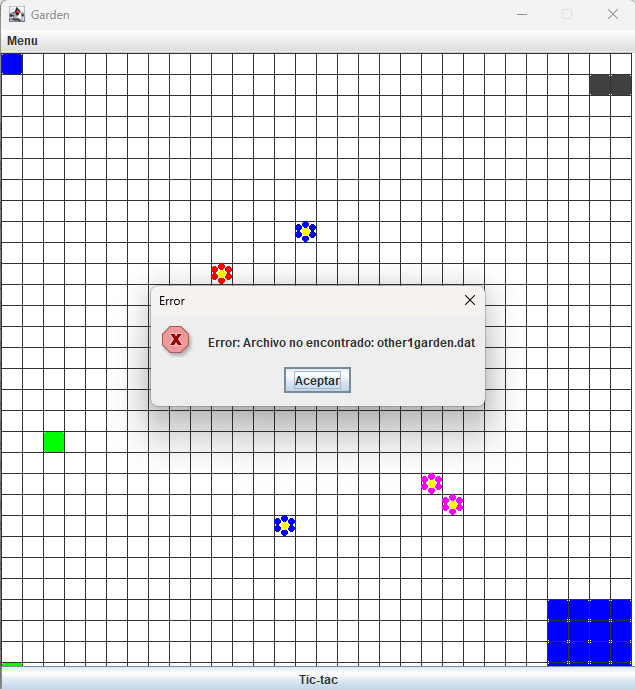
3. Realicen una prueba de aceptación para validar uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

1. Abrimos la aplicación

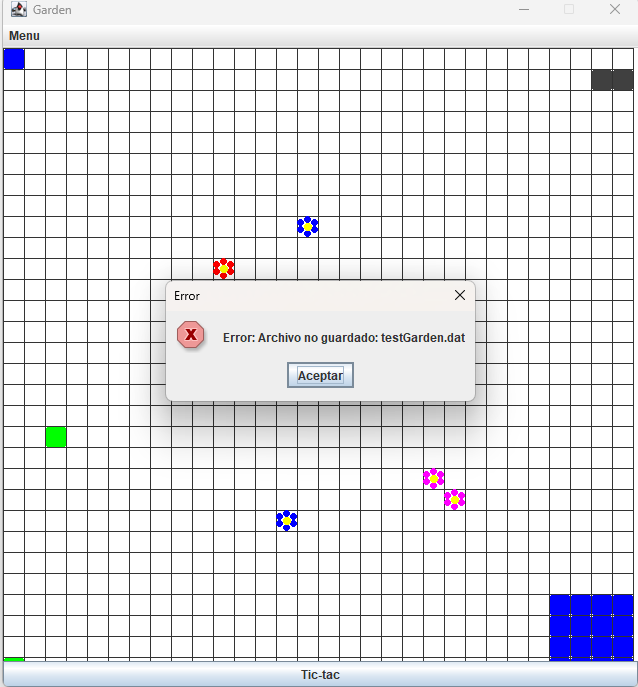
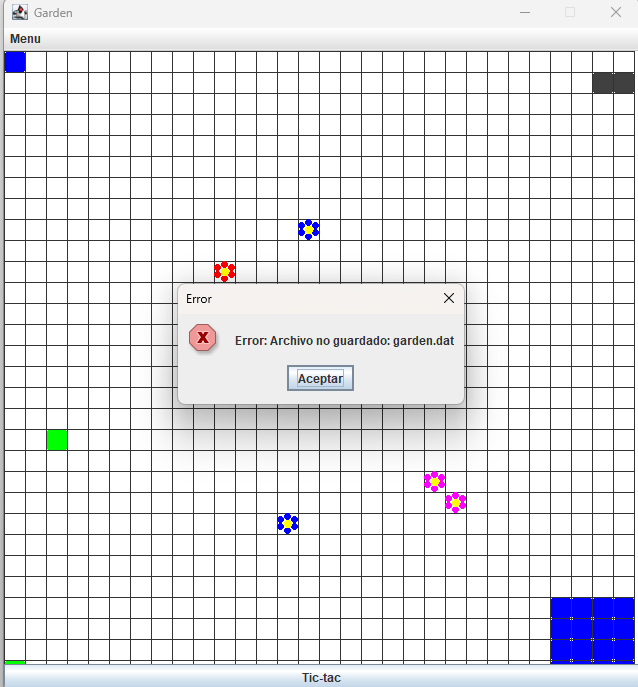
Gráfico, Gráfico de dispersión

Descripción generada automáticamente

2. Probamos las nuevas excepciones creadas para Open, donde abrimos un archivo que no exista para el primer caso y para el segundo caso abrimos un archivo de tipo diferente.



3. Probamos las nuevas excepciones creadas para Save donde salvamos un archivo que no se pueda guardar para el primer caso y para el segundo caso abrimos un archivo de tipo diferente.



**Perfeccionando importar y exportar.**

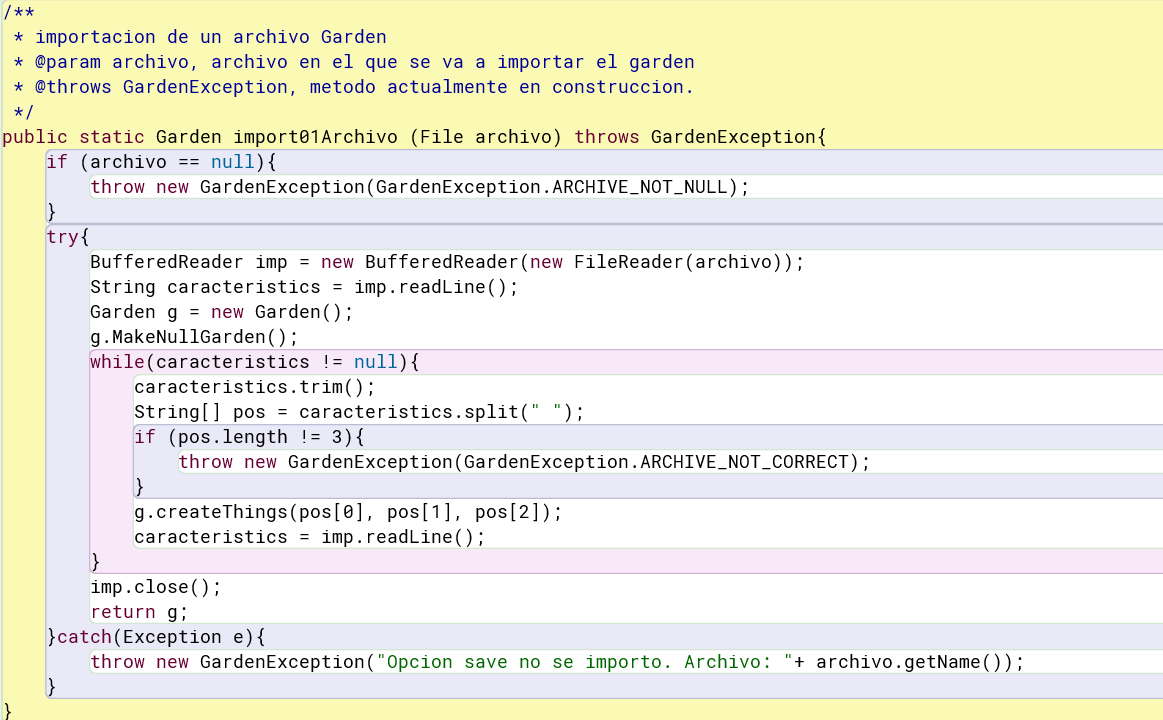
**[En lab06.doc, \*.asta , gardenErr.txt \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de import y export y renómbrenlos como import01 y export01

2. Perfeccionen el manejo de excepciones de los métodos import y export detallando los errores. No olviden pruebas de unidad.

Para perfeccionar el manejo de estas excepciones, consideramos 6 casos:

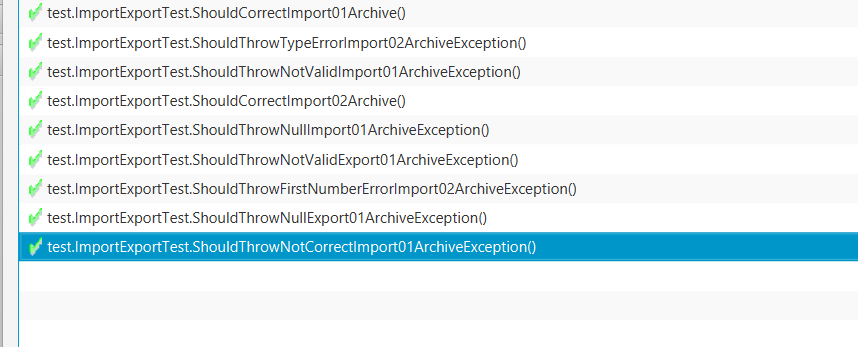
1. Si no se selecciona un archivo
2. Si el archivo seleccionado no es valido
3. Si en el archivo no esta la información correcta
4. Si en la información las posiciones no se pueden pasar a entero
5. Si hay clases que no existen dentro del proyecto
6. Si los números encontrados se salen de la matriz



Interfaz de usuario gráfica, Aplicación

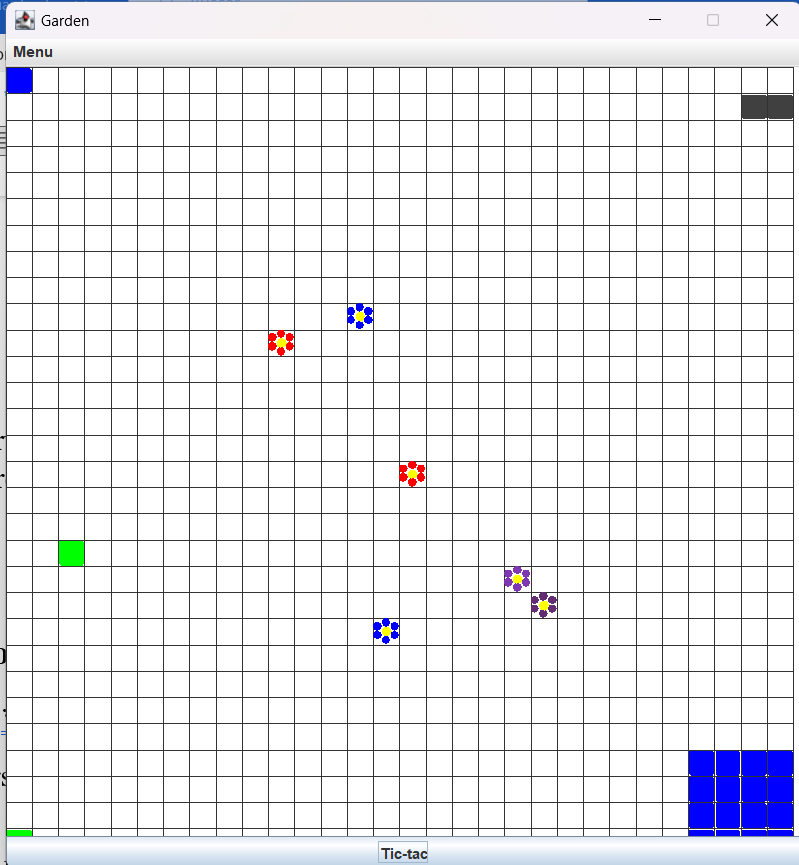
Descripción generada automáticamenteInterfaz de usuario gráfica, Texto

Descripción generada automáticamente

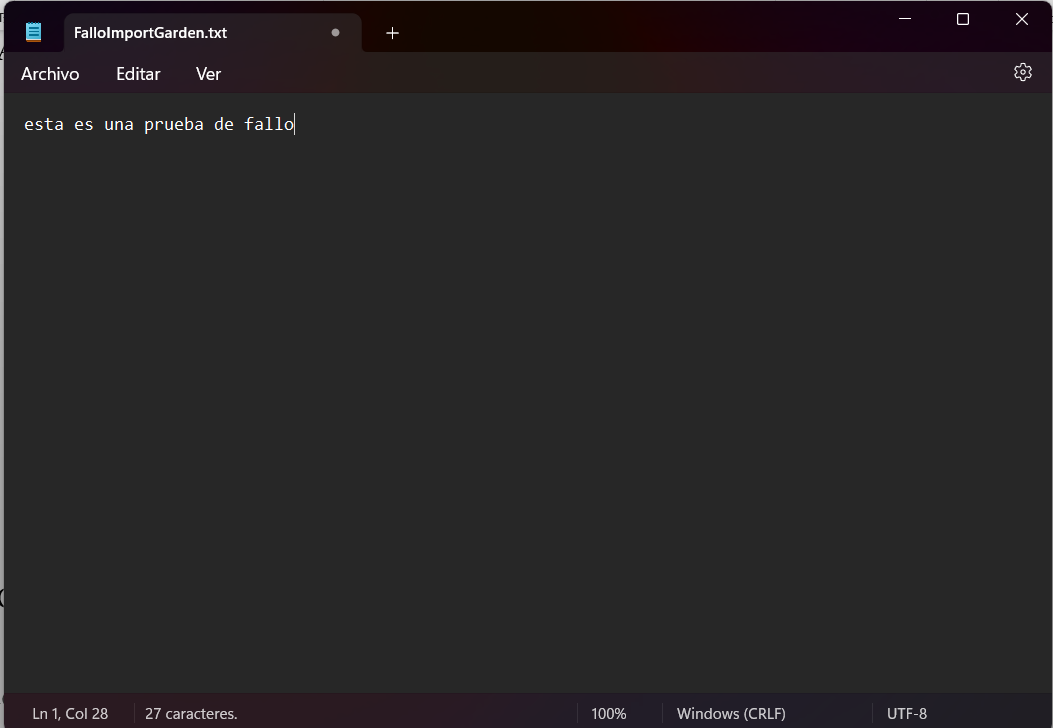


3. Realicen una prueba de aceptación para validar uno de los nuevos mensajes diseñados, ejecútenla y capturen la pantalla final.

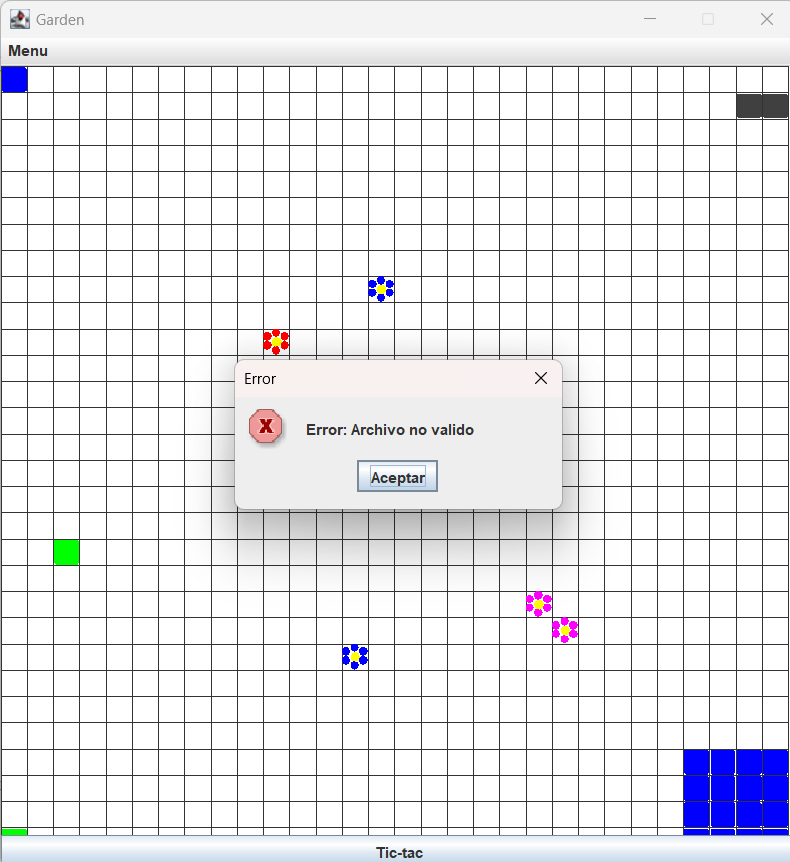
1. Abrimos la aplicación y le damos 1 tic-Tac



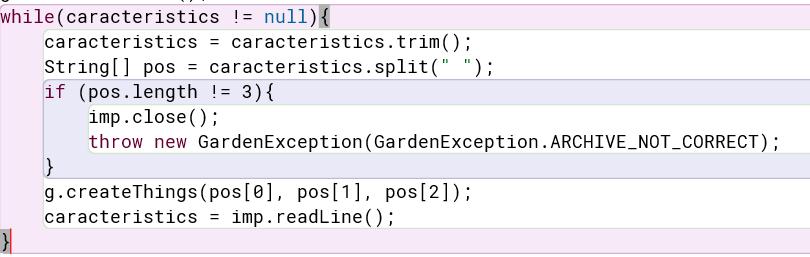
1. Generamos un archivo texto aleatorio con el mensaje “esta es una prueba de fallo”



1. Intentamos abrir el archivo en la aplicación:



Con esto se puede demostrar que si funciona el throws “archivo no valido” en el método import:

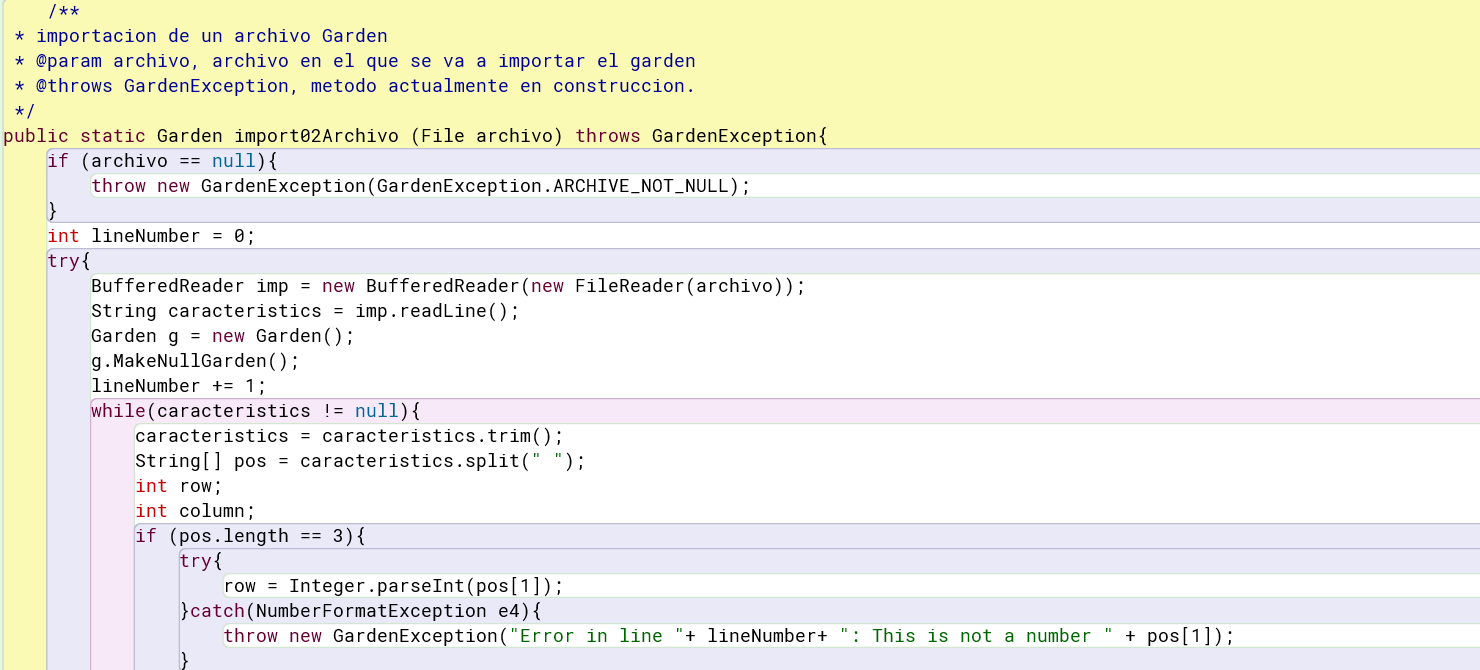


**Perfeccionando importar. Hacia un minicompilador.**

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

1. Copien las versiones actuales de import y export y renómbrenlos como import02 y export02

2. Perfeccionen el método import 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.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

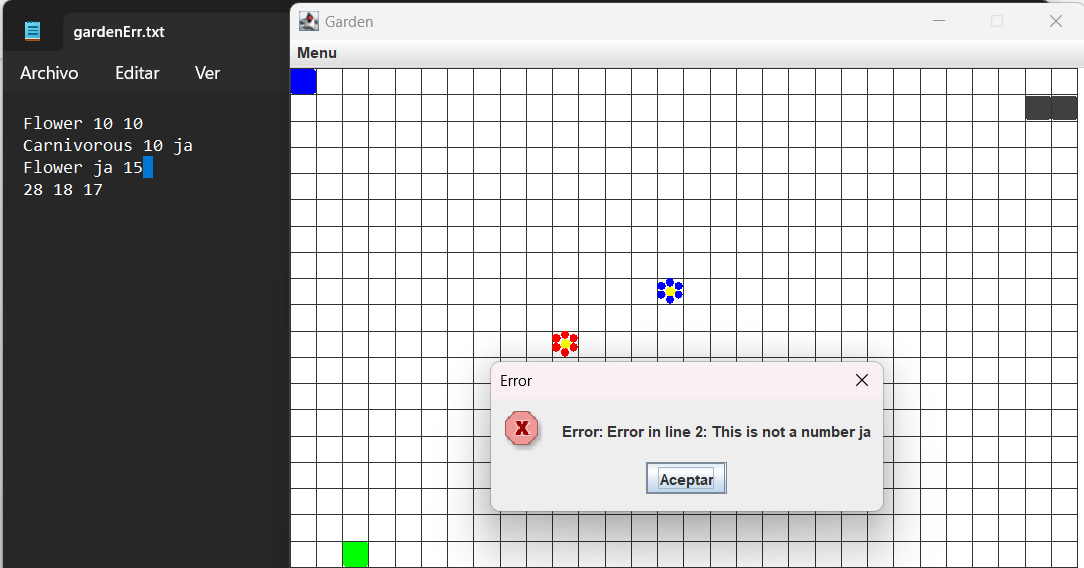
Descripción generada automáticamente

1. Escriban otro archivo con errores, llámelo gardenErr.txt, para ir arreglándolo con ayuda de su “importador”. Presente las pantallas que contengan los errores.

Patrón de fondo

Descripción generada automáticamente

Primer error:



Segundo error:

Imagen que contiene Diagrama

Descripción generada automáticamente

Tercer error:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Archivo de forma completa:

Gráfico

Descripción generada automáticamente

**RETROSPECTIVA**

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes?(Horas/Hombre)

El total fueron 13 horas invertidas en este laboratorio (7 horas por persona aproximadamente)

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

El laboratorio esta completado a excepción del Bono, puesto que preferimos invertir el tiempo al proyecto actual que al bono

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

Utilizamos la practica xp “ONLY ONE PAIR INTEGRATES CODE AT A TIME”, puesto que nos quedó más fácil entender los códigos y el progreso que llevaba el laboratorio cada vez que cambiábamos de ciclo

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

Terminar los 4 métodos “open”,”save as”,”import”,”export”; puesto que es algo nuevo para los 2 lo cual nos generó bastantes inconvenientes al inicio

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

Terminar con el último ciclo de Import, puesto que no entendíamos bien a que se refería y como desarrollarlo.

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

Hicimos bien a comparación de los laboratorios anteriores, el manejo del tiempo ya que este laboratorio logramos completarlo antes de la finalización de la semana. Nos comprometemos a terminar los bonos.